# Foundations of machine learning interpretability

Gianluigi Lopardo


# Fondements de l'Interprétabilité de l'Apprentissage Automatique

**Gianluigi LOPARDO**

Laboratoire Jean-Alexandre Dieudonné (LJAD), Centre Inria d'Université Côte d'Azur (Maasai)

**Présentée en vue de l'obtention du grade de docteur en** Mathématiques **d'**Université Côte d'Azur

**Dirigée par :** Damien GARREAU
**Co-dirigée par :** Frédéric PRECIOSO
**Soutenue le :** 14 Octobre 2024

**Devant le jury, composé de :**
Céline HUDELOT, Professeure, CentraleSupélec, Paris
Jean-Michel LOUBES, Professeur, Université Toulouse Paul Sabatier
Ulrike VON LUXBURG, Professeure, University of Tübingen
Tim VAN ERVEN, Professeur Associé, University of Amsterdam

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité
anr agence nationale de la recherche
AU SERVICE DE LA SCIENCE

# FONDEMENTS DE L'INTERPRÉTABILITÉ DE L'APPRENTISSAGE AUTOMATIQUE

---

***Foundations of Machine Learning Interpretability***

**Gianluigi LOPARDO**

## Jury :

**Rapporteurs**

Céline HUDELOT, Professeure, CentraleSupélec, Paris
Jean-Michel LOUBES, Professeur, Université Toulouse Paul Sabatier

**Examinateurs**

Ulrike VON LUXBURG, Professeure, University of Tübingen
Tim VAN ERVEN, Professeur Associé, University of Amsterdam

**Directeur de thèse**

Damien GARREAU, Professeur, Julius-Maximilians-Universität Würzburg

**Co-directeur de thèse**

Frédéric PRECIOSO, Professeur, Université Côte d'Azur

---

Université Côte d'Azur

Gianluigi LOPARDO

***Fondements de l'Interprétabilité de l'Apprentissage Automatique***

xviii+196 p.

"I checked it very thoroughly," said the computer, "and that quite definitely is the answer. I think the problem, to be quite honest with you, is that you've never actually known what the question is."

*Douglas Adams,*
*The Hitchhiker's Guide to the Galaxy*

# Fondements de l'Interprétabilité de l'Apprentissage Automatique


## Résumé

L'utilisation croissante de modèles complexes d'apprentissage automatique (ML), en particulier dans des applications critiques, a souligné le besoin urgent de méthodes d'interprétabilité. Malgré la variété de solutions proposées pour expliquer les décisions algorithmiques automatisées, comprendre leur processus de prise de décision reste un défi. Ce manuscrit examine l'interprétabilité des modèles ML, utilisant une analyse mathématique et une évaluation empirique pour comparer les méthodes existantes et proposer de nouvelles solutions. Notre principal objectif est sur les méthodes d'interprétabilité post-hoc, qui fournissent des informations sur le processus de prise de décision des modèles de ML après l'entraînement, indépendamment des architectures de modèles spécifiques. Nous nous interessons plus particuliermenet du langage naturel, explorant des techniques pour expliquer les modèles de texte. Nous abordons un défi clé : les méthodes d'interprétabilité peuvent produire des explications variées même pour des modèles apparemment simples. Cela met en évidence un problème critique : l'absence d'une base théorique solide pour ces méthodes. Pour tenter de resoudre ce problème , nous utilisons un cadre théorique rigoureux pour analyser formellement les techniques d'interprétabilité existantes, évaluant leur comportement et leurs limites. Sur cette base, nous proposons un nouvel explicateur pour fournir une approche plus fidèle et robuste pour interpréter les modèles de données textuelles. Nous nous engageons également dans le débat sur l'efficacité des poids d'attention comme outils explicatifs au sein des architectures de transformateurs puissants. Grâce à cette analyse, nous éclairons les forces et les limites des méthodes d'interprétabilité existantes et ouvrons la voie à des approches plus fiables et théoriquement fondées. Cela conduira à une compréhension plus profonde de la façon dont les modèles prennent des décisions, favorisant la confiance et le déploiement responsable dans les applications ML critiques.



**Mots-clés :** Interprétabilité de l'apprentissage automatique, IA Explicable, Traitement du langage


# Foundations of Machine Learning Interpretability


## Abstract

The rising use of complex Machine Learning (ML) models, especially in critical applications, has highlighted the urgent need for interpretability methods. Despite the variety of solutions proposed to explain automated algorithmic decisions, understanding their decision-making process remains a challenge. This manuscript investigates the interpretability of ML models, using mathematical analysis and empirical evaluation to compare existing methods and propose novel solutions. Our main focus is on post-hoc interpretability methods, which provide insights into the decision-making process of ML models post-training, independent of specific model architectures. We delve into Natural Language Processing (NLP), exploring techniques for explaining text models. We address a key challenge: interpretability methods can yield varied explanations even for simple models. This highlights a critical issue: the absence of a robust theoretical foundation for these methods. To address this issue, we use a rigorous theoretical framework to formally analyze existing interpretability techniques, assessing their behavior and limitations. Building on this, we propose a novel explainer to provide a more faithful and robust approach to interpreting text data models. We also engage with the debate on the effectiveness of attention weights as explanatory tools within powerful transformer architectures. Through this analysis, we expose the strengths and limitations of existing interpretability methods and pave the way for more reliable, theoretically grounded approaches. This will lead to a deeper understanding of how complex models make decisions, fostering trust and responsible deployment in critical ML applications.



**Keywords:** Machine Learning Interpretability, Explainable AI, Natural Language Processing.

# Acknowledgments

Firstly, I would like to thank my supervisors. Fred, thank you for the trust you placed in me from the very beginning and for all the time you dedicated to me, especially in the early days, despite your countless commitments. I place immense value on everything I have learned thanks to your advice and all our discussions. Nonetheless, I am grateful to you for making every moment together, whether in a meeting, at lunch or over a coffee, a pleasure. I truly believe that a large part of the great atmosphere I found at Maasai is thanks to you. Your passion has always been engaging and motivating.

Damien, thank you for your constant support throughout these years. Much of the value of this thesis is due to your precise and insightful feedbacks. Your rigor and your desire to really delve deep have inspired me and taught me a lot about how to approach science, and much more beyond that. I will really miss all our conversations, not just the scientific ones. I think we talked about just about everything during this time, and whether it was politics, cooking, economics, wine, cinema, or where to go on vacation, you always offered an interesting and unique perspective. I truly believe that every single conversation we had enriched me in some way. You have been an important mentor to me, in the highest sense of the word. Thank you so much.

I also want to express my sincere gratitude to the committee members. Professors Céline Hudelot, Jean-Michel Lobes, Tim van Erven, and Ulrike von Luxburg: I am incredibly honored to have you on my PhD jury. Thank you for accepting and for the patience and time you have dedicated to the entire process and to my emails. Special thanks to the reviewers, Céline and Jean-Michel: your reports are full of insightful comments, which I sincerely appreciated.

I am grateful to the fantastic colleagues and officemates, both past and present, that I've had at Maasai and LJAD. Thank you to the *Valrose PhD* gang: my life in Nice would have been empty without you. Thank you to those who managed to stay close despite the distance; I constantly felt the support of my friends and would not have been able to get through so much without you. Thank you to my entire family for your unconditional support.

# Table of contents

## Appendix

# List of figures

# List of tables

# CHAPTER 1

# Introduction

In the last fifteen years, Artificial Intelligence (AI) has transitioned from a futuristic and almost science-fiction concept, mainly confined to academia, to a tangible tool successfully applied across nearly every sector. More recently, AI chatbots like ChatGPT [Bahrini et al., 2023], have been developed and are used daily by millions of people (see Figure 1.1), radically transforming not only our digital experience but also the way we learn and work (Smith [2021], Schmelzer [2024], Marr [2024]). This rapid evolution has been primarily driven by Machine Learning (ML): the study and development of algorithms that learn from data and generalize to unseen instances (UC Berkeley [2020]). Machine Learning has harnessed the vast availability of data on the internet [Boucher, 2020], the increase in computational power [Ajani et al., 2024], and advancements in technology and methodology, such as improved network architectures and optimization techniques. Additionally, the boost from commercial applications has made large-scale AI applications possible. Thanks to machine learning, **automated systems can perform tasks without explicit instructions by learning recurrent patterns from large amounts of available data**.

The widespread adoption of AI in various sectors, especially critical ones like healthcare, finance, and transportation brings significant economic, political, and social implications (Roser [2022]). AI has been leveraged from developing new healthcare diagnosis based on patient data [Bohr and Memarzadeh, 2020] to optimizing financial decisions in real-time [Nazareth and Reddy, 2023], from improving transportation efficiency through predictive maintenance [Bharadiya, 2023] to personalizing education by understanding individual learning patterns [Chen et al., 2020]. The integration of AI in these fields has the potential to improve efficiency and optimize decision-making processes, but it also presents challenges related to privacy [Manheim and Kaplan, 2019, Murdoch, 2021], ethics [Cath, 2018, Gerke et al., 2020, Akgun and Greenhow, 2022], and security [Hoadley and Lucas, 2018, Li, 2018, Akgun and Greenhow, 2022]. These profound implications have led AI to emerge as a crucial topic in public debate (The Economist [2023], Kaminski [2024], The Economist [2024a]), attracting the attention and participation of various stakeholders, from the scientific community to industry professionals, domain experts, politicians and civil society. Today, **there is a strong demand for reliable and manageable AI systems** that can meet the needs and expectations of a wide range of users and stakeholders, respecting societal norms and values. **Interpretability is among the most urgent requirements.**

## 1.1 The need for interpretability

As AI continues to evolve at a rapid pace, the need to ensure the accountability of these systems becomes increasingly important. In certain domains, such as music recommendation systems, the risks associated with AI might be negligible, with little to no potential harm for users. However,

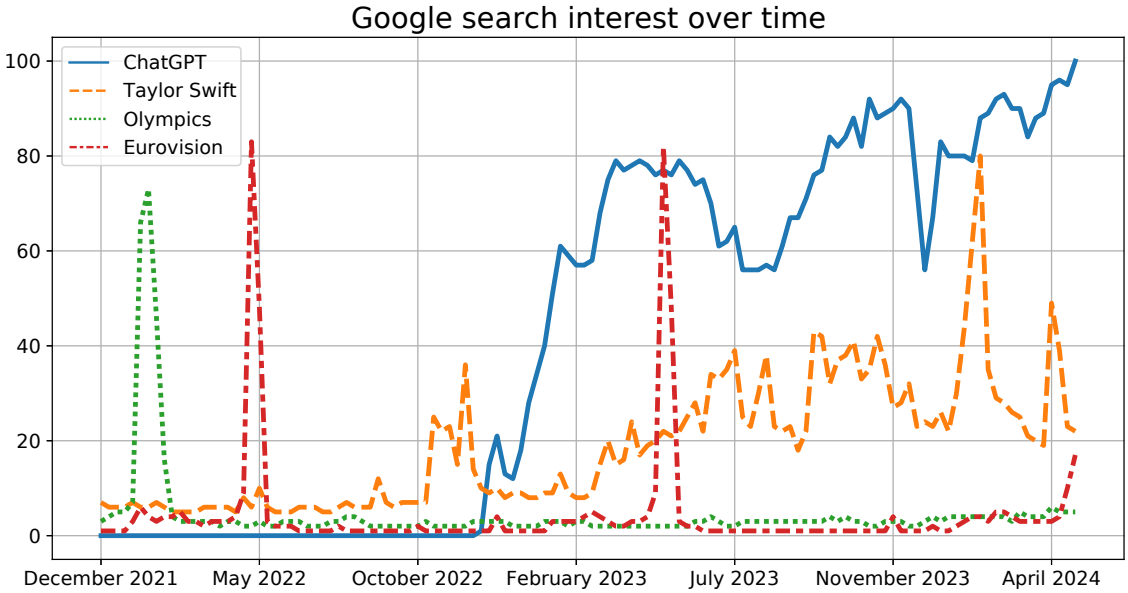


Figure 1.1 – **Comparison of worldwide Google search trends for ChatGPT, Taylor Swift, the Olympics, and Eurovision Song Contest.**. Following its debut, ChatGPT has seen a steady rise in search interest, outpacing the 2023 Person of the Year (TIME [2023]), and eclipsing major global events in sports and music. This trend underscores the immense public interest in ChatGPT, growing since its release.

**when AI is employed in high-stakes areas, such as when making medical decisions, understanding the underlying principles that govern these models becomes imperative** [Jan et al., 2020]. This is where the field of eXplainable AI (XAI) comes into play [Molnar, 2020], aiming to create techniques that produce more understandable and interpretable AI predictions, thereby bridging the gap between AI decision-making and human understanding.

### 1.1.1 Use cases: *when do we need interpretability?*

This section illustrates specific situations where interpretability is particularly beneficial for citizens and customers by providing a few concrete examples. These examples will demonstrate how transparency in AI systems can enhance trust, ensure fairness, and improve outcomes across various domains.

*Example 1.1.1* – **Loan application.** Imagine applying for a loan to purchase your dream house. Figure 1.2 shows a schematic representation of this process. Traditionally, obtaining a mortgage involves a loan officer evaluating your financial situation and the property's worthiness. This process includes analyzing documents, verifying income, job security, creditworthiness, and details about the property such as location, size, and ownership history. After gathering all the required documentation, you visit the bank, providing personal information such as your place of birth, age, gender, and family situation. The bank officer diligently completes the necessary forms. However, at the end of the process, your loan application is rejected.

At the very least, you will want an explanation for the denial. If a human operator made the decision, you might receive an explanation (though potentially dubious) that you could contest and


Figure 1.2 – **An illustration of an automated loan application process as in Example 1.1.1.** A customer arrives at the bank with personal documents (income, gender, age, nationality, neighborhood). The bank leverages artificial intelligence to streamline the loan application process: the bank operator inputs the customer's information into a sophisticated model. This *black-box* model, while complex and seemingly mysterious, analyzes the data and produces the outcome: the loan application has been rejected.

discuss. However, the situation becomes more complicated if the bank uses complex algorithms to analyze your data and make automated decisions. While efficient, such models can be opaque, making it difficult, if not impossible, to understand the reasoning behind their rejections.

There might be legitimate reasons for the denial. Perhaps the AI considered the house's location and factored in data suggesting upcoming construction that could decrease its value. It might have concluded the house is too expensive for your income or simply too large for your needs. In the worst case, one might suspect discriminatory biases, such as racism or sexism, influencing the outcome, especially if the provided reasons seem arbitrary. Even the loan officer might not understand why your application was denied. **When the only possible explanation given is a vague "our super accurate automated system decided so", it becomes nearly impossible to challenge such biases or understand the factors that led to the denial.**

*Example 1.1.2* – **U.S. Welfare eligibility.** The United States Government frequently employs AI systems to determine individuals' eligibility for welfare assistance (Gilman [2020]). However, these systems, designed to detect welfare fraud, often end up disproportionately penalizing those in need. Despite intentions to streamline processes and minimize fraudulent claims, **these algorithms can inadvertently perpetuate racist and sexist biases**, leading to erroneous denials of aid for deserving applicants. Clear **explanations can promptly detect these biases and help understand why a decision has been made**, emphasizing the need for interpretability in AI systems used for such critical purposes.

*Example 1.1.3* – **Spurious correlations.** Consider an image classifier for animals that boasts high overall accuracy. However, a detailed examination of the model reveals some disconcerting biases. For instance, the model consistently predicted "waterbird" for any image containing a bird with a sea background (as in Figure 1.3). Conversely, it classified any dog with snow in the background as a "wolf" (as in Ribeiro et al. [2016, Figure 11]).

The model failed to recognize the key features that define a waterbird, such as the shape of the beak or the pattern of the wings, or a wolf, such as the texture of the fur or specific facial features. Instead, it relied on the background of the image, which can often be misleading. Detecting this

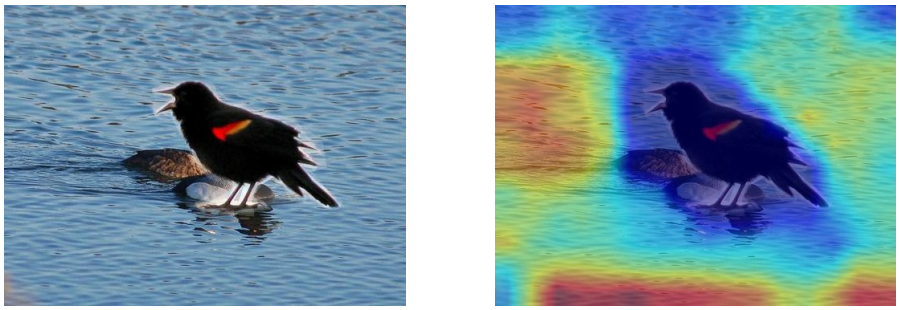

Figure 1.3 – **Interpretability helps detecting spurious correlations** (Example 1.1.3). On the left, an image of a waterbird is shown [Sagawa et al., 2020], and on the right, the output of an interpretability method that produces saliency maps. The saliency map reveals a critical insight: the model is primarily focusing on the water in the image rather than the bird itself. This suggests that while the model is correctly classifying the bird as a waterbird, it is doing so for the wrong reason, relying on the presence of water in the image to make the prediction. The model is relying on a spurious correlation, using the presence of water in the image to make the prediction. This example clearly demonstrates that a **model's accuracy is not the sole determinant of its usefulness or reliability; understanding *why* and *how* it makes its predictions is equally important.**

problem is challenging. Explainable AI tools can identify these biases early, allowing them to be addressed before the model is deployed to production.

While this example might not have serious risks, imagine the case where the model is a brain scanner classifier used to detect the presence of a tumor. In such high-stakes scenarios, spurious correlations could lead to devastating misdiagnoses. Therefore, ensuring the interpretability of AI models is crucial to understand and mitigate potential biases and errors.

*Example 1.1.4* – **Amazon automated recruitment process.** In 2014, Amazon attempted to automate its recruitment process using a machine learning model (Dastin [2018]). This model was trained on the resumes of existing employees to identify the most promising candidates. However, it soon became apparent that the system was not evaluating candidates for software developer roles and other technical positions in a gender-neutral manner. Essentially, Amazon's system had learned to favor male candidates over female candidates. This bias was not due to the model's design but rather a reflection of the training data it was fed. The models were trained on resumes submitted to the company over a decade, a period that mirrored the male predominance within the broader tech industry. Consequently, the model learned to replicate this imbalance, underscoring the critical importance of careful data selection.

*Example 1.1.5* – **Prediction of Criminal Recidivism.** In the criminal justice system, decisions about bail, sentencing, and parole have profound implications for individuals' lives. To assist judges in making fast and accurate decisions, AI models are increasingly being used (Hao [2019]). These models analyze vast amounts of data to predict the likelihood of an individual committing a future crime. By considering factors such as criminal history, demographic information, and socioeconomic background, these AI systems aim to provide judges with additional insights into a defendant's risk level. However, **concerns have been raised about the fairness and transparency of these models.** Critics argue that they may perpetuate biases present in historical data, leading to discriminatory outcomes, particularly against marginalized communities.

*Example 1.1.6 –* **Automated medical prescriptions.** Healthcare systems worldwide, especially in the wake of the pandemic, are under mounting pressure. Doctors often find themselves overwhelmed by paperwork, leading to exhaustion and dissatisfaction. AI can alleviate this burden by assisting with treatment prescriptions, particularly in less severe cases. By analyzing extensive patient data, including medical records, lab results, and medication profiles, AI models can propose fast and accurate treatment plans. These models draw from a large pool of patient data, enabling them to leverage insights from a wider spectrum of cases compared to individual doctors. This potential for learning from a broader range of scenarios theoretically allows for more informed recommendations, thereby enhancing patient outcomes. However, the question remains: **can a computer-generated medical prescription, lacking human-readable explanation, be fully trusted?**

### 1.1.2 Motivation: *why do we need interpretability?*

As demonstrated with previous examples, **machine learning interpretability is vital for fostering trust and transparency in AI systems** across various domains [Doshi-Velez and Kim, 2017, Gilpin et al., 2018, Tolmeijer et al., 2022, Verma et al., 2023]. The lack of trust is indeed one of the main deterrents to the use of artificial intelligence systems [Lee and Rich, 2021, Ma et al., 2023], despite their effectiveness. In their research, Jan et al. [2020] identify the lack of explainability in AI models as a major obstacle to their adoption in business settings. Business users, often domain experts but not necessarily skilled in data science, struggle to trust model predictions without a clear understanding of their automated process. This **skepticism arises from the inherent lack of transparency in decision-making models**, hindering the full utilization of AI's potential in improving business operations [Lopardo, 2021].

This is exemplified in Example 1.1.1. While automated systems hold promise for enhancing the efficiency of loan application processes and potentially improving overall accuracy, their reliance on opaque decision-making poses significant challenges. Banks face difficulties in trusting a system that makes specific and vital decisions for their operation without providing explanations. The lack of transparency restricts the bank's ability to intervene and validate or reject automatic decisions. **Interpretability is fundamental for effective human-AI collaboration** [Zhang et al., 2022], and responsible AI development. Doshi-Velez and Kim [2017] argue that explanations are essential for trusting automated decisions, facilitating human oversight, and ensuring accountability.

In Example 1.1.6 and in the healthcare sector in general [van de Sande et al., 2021, Burgess et al., 2023], the interpretability of machine learning models is of fundamental importance [Vellido, 2020, Lee and Rich, 2021, Verma et al., 2023]: clinicians confidence is necessary for adoption [Charachon et al., 2021]. Miotto et al. [2018] suggests that machine learning could be instrumental in translating large biomedical data into an improvement in human health, but recognizes the importance of making these models more understandable at least for domain experts and scientists, and supports the development of interpretable systems to guarantee human understanding. Li et al. [2019] highlights the potential of AI in bioinformatics, in tasks ranging from processing DNA sequences to classifying biomedical images, but emphasizes the need for transparency in the decision-making process of these models. Stiglic et al. [2020] criticizes the absence of explainability, especially in light of the rapid spread of advanced AI applications in sensitive applications. Abdullah et al. [2021] argues that it would be advantageous if ML models could provide explanations that allow doctors to make data-based decisions. This could reveal specific

patient characteristics, such as age, medical history, and test results, which significantly influence the prescription. In this way, the model can be understood and therefore trusted by healthcare professionals. **Interpretability can be crucial in life-or-death scenarios, potentially enabling the detection of anomalous decisions.**

However, providing a form of explanations to support a decision does not automatically ensure trust in an automated system. In a user-based study, Ahn et al. [2024] analyzed how interpretability influences users' task performance and trust in AI, concluding that it had modest effects on participants. Specifically, in a clinical context, Yang et al. [2023] found that when the clinician's hypothesis was wrong and the AI advice was correct, the explanations rarely persuaded clinicians to take the advice. Ehsan et al. [2021] states that **for interpretability to effectively contribute to transparency and social acceptance, explanations must be designed for being user-centered.**

Beyond ensuring trust, XAI offers significant advantages for developers. Machine learning models can sometimes establish spurious correlations [Hermann et al., 2020, Izmailov et al., 2022], leading to incorrect outputs due to unforeseen reasons. In other instances, they may arrive at correct predictions, but the underlying justifications may be flawed or misleading. This is exemplified in Example 1.1.3, where the waterbird picture on the left panel of Figure 1.3 is correctly classified. However, the saliency map on the right panel reveals a critical insight: the model is primarily focusing on the water in the image rather than the bird itself. While these correlations may result in high accuracy metrics, they are misleading and demonstrate misunderstanding by the model. **Interpretability techniques aid in debugging machine learning models** by examining internal mechanisms, enhancing robustness against adversarial perturbations, and ensuring meaningful variables drive the output, thereby validating the model's reasoning [Arrieta et al., 2020].

Machine learning algorithms are designed to generalize to unseen instances by learning from large amounts of existing data. **An intrinsic problem of machine learning, due to its very definition, is that it can result in models perpetuating existing biases.** In Example 1.1.4, Amazon's hiring model was trained on applications from the previous decade, in an industry heavily dominated by men. Consequently, the model learned that, statistically, a successful candidate is male and perpetuated this bias, effectively adopting sexist behavior (Dastin [2018]). By interpreting model's predictions, it became evident that it paid significant attention to gender, enabling prompt intervention. Mehrabi et al. [2021] analyzes various real-world applications that have shown bias and identifies several sources of bias that can influence AI applications. Similar issues arise in Example 1.1.5, where concerns have been raised about the fairness of recidivism prediction algorithms (Angwin et al. [2016]), which exhibit racist behaviors [Dressel and Farid, 2018]. Efforts are underway to enhance the interpretability and fairness of AI systems in the criminal justice domain [Berk et al., 2021].

Researchers are developing methods to detect and mitigate bias in algorithmic decision-making, as well as techniques to provide judges with understandable explanations for AI-generated recommendations [Barocas et al., 2023]. Similarly, Schwartz et al. [2022] discusses how bias can emerge in AI systems: bias can be linked to the design and creation of the system, the data used to train models, or the biases of the individuals who create and use AI systems. Caton and Haas [2024] also delves into the issue of fairness in machine learning, **identifying interpretability as a central tool for evaluating the fairness of automatic decisions.** Interpretability can be leveraged to detect and mitigate biases, ensuring non-discriminatory behaviors. Nevertheless, while interpretability and fairness are deeply connected, they must be approached with careful consideration of their limitations. Deck et al. [2024] critically examines the relationship between explainable AI

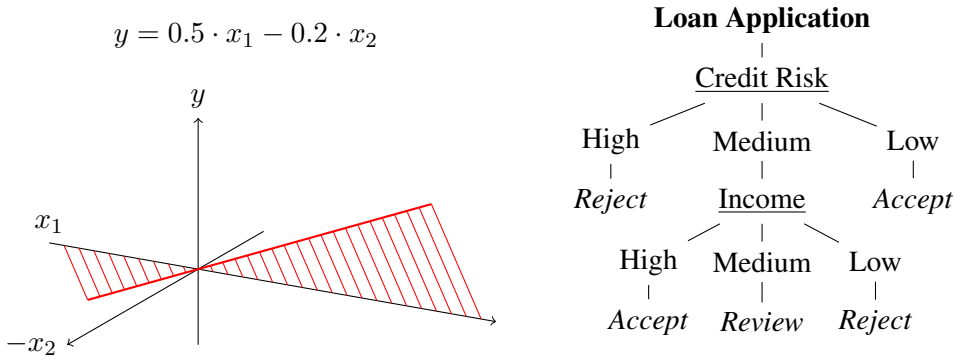


Figure 1.4 – **Illustration of two interpretable models.** On the left panel, a small linear model predicting an outcome $y$ as a linear combination of two parameters: $x_1$ and $x_2$. On the right panel, a small decision tree is used for loan applications. This tree operates on a simple risk assessment: applicants with high credit risk are immediately rejected, while those with low credit risk undergo further scrutiny based on their income. These two models are intrinsically interpretable; **their simple structure allows humans to predict precisely how any variation in the input parameters will influence the output**, providing clear and understandable results. This transparency is crucial in many fields where understanding the decision-making process is as important as the decision itself.

and fairness, highlighting that many claims about XAI's fairness benefits often lack grounding, or are poorly aligned with the actual capabilities of existing interpretability methods.

### 1.1.3 Interpretable models

It is now clear that interpretability is fundamental, therefore it is important to understand how to achieve it. However, as explored in Section 1.4.1, defining interpretability in a clear and unambiguous manner proves to be a complex challenge. **Interpretability is ensured by using intrinsically interpretable models**, such as linear models and decision trees, which are intrinsically interpretable (or *self-explainable*) by design [Doshi-Velez and Kim, 2017] due to their simple underlying structure. As emphasized by Molnar [2020], such models provide transparency by design, allowing for straightforward interpretation of their predictions.

**Linear models.** In the regression setting, linear models predict outcomes by computing a weighted sum of input features. Mathematically, this can be expressed as:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_b x_b + \varepsilon\,, \tag{1.1}$$

where $y$ is the predicted outcome for the instance point $x$, while $x_1, x_2, \ldots, x_b$ are the input features, $\beta_0$ is the intercept, $\beta_1, \beta_2, \ldots, \beta_b$ are the weights (coefficients) associated with each feature, and $\varepsilon$ is the error term. These weights and the intercept are learned from the data in such a way as to minimize the error between the actual prediction and the model outcome. These parameters are typically determined using methods like LASSO regression [Tibshirani, 1996] or Ridge regression [Hoerl and Kennard, 1970], which **minimize the differences between the actual and estimated**

**outcomes** by solving the following optimization problems, respectively:

$$\min_{\beta} \left\{ \sum_{i=1}^{n} \left( y^{(i)} - \beta_0 - \sum_{j=1}^{b} \beta_j x_j^{(i)} \right)^2 + \lambda \sum_{j=1}^{b} \beta_j^2 \right\}$$

and

$$\min_{\beta} \left\{ \sum_{i=1}^{n} \left( y^{(i)} - \beta_0 - \sum_{j=1}^{b} \beta_j x_j^{(i)} \right)^2 + \lambda \sum_{j=1}^{b} |\beta_j| \right\},$$

where $x_j^{(i)}$ is the $j$-th feature of the $i$-th sample, $y^{(i)}$ its actual outcome, while $\lambda$ is a regularization parameter that controls the trade-off between fitting the data and keeping the model coefficients small to prevent overfitting (see Hastie et al. [2009], Section 3.4.3).

One of the main advantages of linear models is their interpretability. Each coefficient $\beta_j$ directly represents the change in the outcome variable $y^{(i)}$ for a one-unit change in the corresponding predictor $x_j^{(i)}$, holding all other predictors constant. **This straightforward relationship allows for an easy understanding and explanation of the effect of each feature.**

Consider the left panel of Figure 1.4. It shows that, all else being equal, increasing the value of $x_1$ by one unit will increase the outcome $y$ by $0.5$, while an augment of one unit of $x_2$ will results in a decrease of $0.2$. The simplicity and interpretability of the linear relationships have made linear models a popular tool not only in statistics and computer science but also in fields such as medicine, sociology, psychology, and other quantitative research fields.

**Logistic regression.** Logistic regression extends linear regression to binary classification problems by using the logistic function to ensure outputs fall between 0 and 1, making them interpretable as (*pseudo*) probabilities. This model is particularly useful for two-class problems, providing a meaningful threshold for class separation and avoiding extrapolation outside the $0 - 1$ range. Logistic regression works by modeling the *log-odds* (or *logit*) of the probability of an event occurring:

$$\text{logit}(p) := \log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_b x_b\,, \tag{1.2}$$

where $p$ is the probability of the event occurring, and, as before, $x_1, x_2, \ldots, x_b$ are the input features, $\beta_0$ is the intercept, and $\beta_1, \beta_2, \ldots, \beta_b$ are the coefficients.

The coefficients in a logistic regression model represent the change in the log-odds of the outcome for a one-unit change in the predictor variable, holding all other predictors constant. This relationship helps to understand and explain how each feature impacts the likelihood of the event occurring. To convert the log-odds back to a probability, the logistic function is used:

$$p = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_b x_b)}}\,.$$

Understanding the log-odds is particularly insightful when considering the decision boundary. Indeed, close to the decision boundary, where the log-odds are near zero, small changes in input

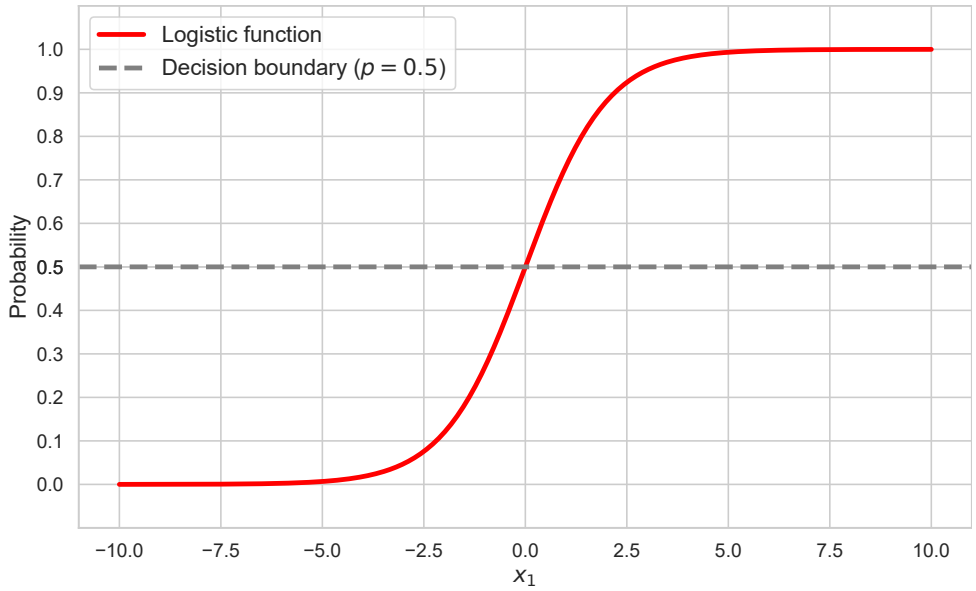


Figure 1.5 – **Probability curve in logistic regression** with the decision boundary at $p = 0.5$. Here, there is only one input feature ($x_1$, *i.e.*, $b = 1$). Points above this threshold are classified as *True*, while points below are classified as *False*. Close to the boundary, where $p$ is *near* 0.5, the classification is sensitive: a small variation in input features can lead to a change in prediction. Far from the boundary, the model shows stronger confidence in the predicted class.

features can significantly affect the predicted probability, reflecting high uncertainty in classification (see Figure 1.5). Far from the decision boundary, the log-odds are larger in magnitude, indicating stronger confidence in the predicted class and smaller impacts of input changes on the probability.

**Decision rules.** Decision rules are often used in machine learning [Holte, 1993]. These consist of simple `IF-THEN` statements of the form:

`IF` condition 1 `AND` condition 2 `AND` $\cdots$ `THEN` outcome.

The effectiveness of decision rules is measured by *support* (the proportion of instances in the dataset where the rule applies) and *accuracy* (the proportion of correct predictions made by the rule). This has a very **straightforward interpretation: one simply needs to check whether a series of conditions are satisfied or not.**

Various approaches have been proposed to identify an optimal trade-off between support and accuracy [Borgelt, 2005, Letham et al., 2015]. RuleFit [Friedman and Popescu, 2008] combines decision rules with linear regression to create a model that is simple and interpretable like linear regression, but also incorporates interaction effects. The model is learned as linear combinations of compact decision rules derived from the data.

**Decision trees.** Decision trees [Breiman et al., 1983] make predictions based on a series of nested if-else conditions on the input features. Each condition is made at a node of the tree, and each outcome determines the path to the next node. The final prediction is made at the leaf nodes.

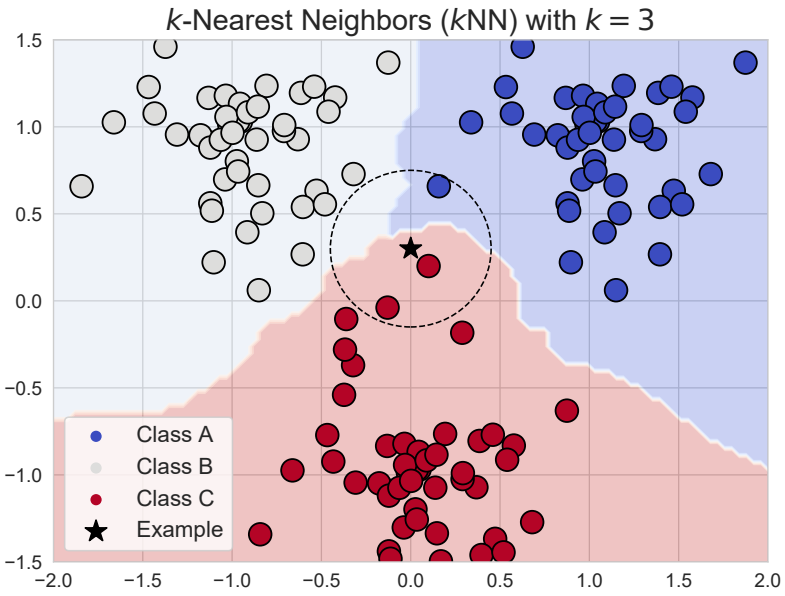


Figure 1.6 – **Illustration of a $K$-Nearest Neighbors ($k$NN) classification model** with $K = 3$. The plot shows decision boundaries separating three classes $A$, $B$, and $C$, in a two-dimensional feature space. Points represent data samples, with colors indicating their respective class. The black star marks the example point to predict. The dashed circle outlines an area around it, encompass three points: two belonging to the class $C$ and one of class $A$. By majority voting, the example is therefore classified as $C$.

They are intrinsically interpretable due to their simple structure, especially when they have a small depth.

In the case of orthogonal splits to the axes, decision trees split the feature space into rectangular regions by making decisions based on a single feature at a time, in the form $x_j \leq \tau_j$ (where $x_j$ is a feature value and $\tau_j$ is the corresponding threshold). This means that each decision rule in the tree corresponds to an axis-aligned cut in the feature space, partitioning it into subspaces where the decision boundaries are parallel to the feature axes.

This method allows for a clear and interpretable structure where the criteria for each decision are straightforward, making it easy to understand how the final predictions are reached. The interpretation of a decision tree is straightforward: starting from the root node, one follows the path determined by the decisions at each node, leading to the prediction at the leaf. All splits along the path are connected by logical `AND` operations. Consider the right panel of Figure 1.4, where a small decision tree is employed to process the loan application of Example 1.1.1. For instance, the loan is always accepted if the credit risk is low (credit risk $\leq \tau_1$) and the income is high (income $> \tau_2$). For any new loan application, the decision-making path can be precisely followed, allowing for an unambiguous determination of how changes in the inputs will impact the output.

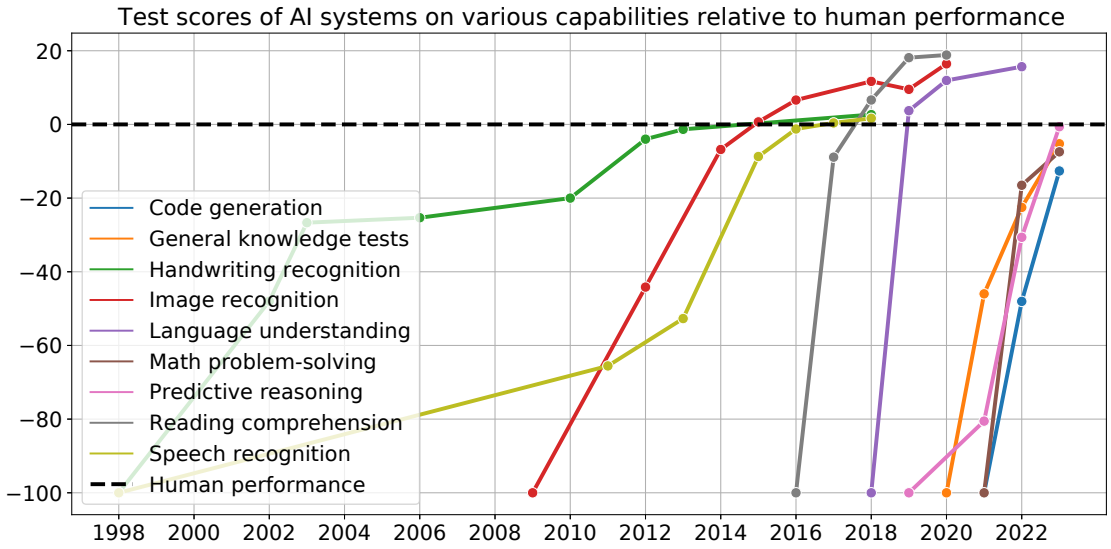


Figure 1.7 – **AI test scores across different capabilities relative to human performance.** In each domain, the initial AI performance is designated as $-100$. Human performance serves as the baseline, set to zero. Crossing the zero line indicates that AI performance surpasses human performance. Data from Kiela et al. [2023].

**Other interpretable models.** Among standard machine learning algorithms, the Naive Bayes classifier and $k$-Nearest Neighbors ($k$NN) are intrinsically interpretable [Hastie et al., 2009, Molnar, 2020]. The Naive Bayes model calculates the likelihood of a specific outcome $y$ occurring given the observed features $x$ by assuming feature independence:

$$\mathbb{P}(y \mid x) \propto \mathbb{P}(y) \prod_{j=1}^{b} \mathbb{P}(x_j \mid y) \ .$$

This assumption simplifies the interpretation, as the contribution of each feature $x_j$ to the prediction can be understood through its conditional probability given the class label $y$.

The $k$NN model predicts the class of a data point by identifying its $k$ nearest neighbors in the feature space and assigning the majority class among them:

$$\hat{y} = \arg\max_{\ell \in \mathcal{C}} \sum_{i \in \mathcal{N}_k(x)} \mathbb{1}_{y^{(i)}=\ell} \, ,$$

where $\mathcal{N}_k(x)$ represents the indices of the $k$ nearest neighbors of $x$, $\mathbb{1}$ is the indicator function, and $\mathcal{C}$ is the set of possible labels to predict. This method is interpretable because predictions are directly based on the proximity of data points, providing clear insight into how the neighbors influence the prediction. An illustration for $k = 3$ and $\ell \in \mathcal{C} = \{A, B, C\}$ is shown in Figure 1.6.

## 1.2 A brief overview of contemporary AI

**The current state of AI demonstrates superior performance compared to humans in various tasks** [Mnih et al., 2013,0, Silver et al., 2016, Doshi-Velez and Kim, 2017], as highlighted

in the Figure 1.7. This success has contributed to making AI an omnipresent and influential presence in society today. The driving force behind this rapid evolution has been machine learning, especially since the rise of artificial neural networks: far from the interpretable models seen in Section 1.1.3. These networks consist of multiple layers of interconnected nodes, the *neurons*, each performing simple calculations involving linear combinations of inputs, typically followed by a nonlinear activation function. However, when combined across multiple layers, these simple neurons enable the network to perform highly complex computations. As AI evolves at a rapid pace, **the complexity of machine learning models continues to increase.** This increasing intricacy often surpasses human's ability to understand the mechanisms that guide such models. This issue becomes particularly relevant when such models are employed in critical sectors as those mentioned in Section 1.1.2, where any misinterpretation or error could have serious repercussions.

The aim of this section is to provide a concise overview of the main developments in the field of artificial intelligence. It starts from the basics of neural networks and progresses to the modern large language models (LLMs) that define the current technological landscape. This is intended to underscore the notable disparities in complexity compared to, say, the linar models outlined in Section 1.1.3, thereby elucidating the challenges of interpretability posed by state-of-the-art models.

### 1.2.1 Neural networks

The seminal work on the perceptron [Rosenblatt, 1958a] laid the groundwork for neural networks by introducing a probabilistic model for information storage and organization in the brain. The perceptron, a single-layer neural network, is a binary classifier that maps its input (a real-valued vector) to an output value (a single binary value) using a set of weights and a biases. This was followed by significant advancements in the 1980s with the development of backpropagation, a method for training neural networks through gradient descent [Rumelhart et al., 1986]. Backpropagation allows the weights of the network to be updated in a way that minimizes the error between the network's output and the desired output for a given input, effectively *learning* from the data. For detailed explanations of such techniques, and a comprehensive treatment of neural networks, refer to classical textbooks [Bishop and Nasrabadi, 2006, James et al., 2013].

Figure 1.8 illustrates a simple neural network architecture comprising three hidden fully-connected layers. This schematic representation highlights the flow of data through the network, from the input layer to the output layer. Each neuron in a given layer is fully connected to every neuron in the subsequent layer. Specifically, each input neuron sends its signal to every neuron in the first hidden layer. Similarly, each neuron in the first hidden layer sends its processed signal to every neuron in the second hidden layer, and so on. This pattern continues until the output layer is reached.

The connections between neurons, represented by lines, signify the weights and biases that are adjusted during the training process. These weights and biases determine the strength and nature of the signals being passed between neurons, allowing the network to learn and make predictions. Importantly, these connections are enhanced by applying non-linear transformations. These **non-linearities, typically implemented as activation functions, introduce a level of complexity that enables the network to learn non-linear relationships within the data.** Without these non-linearities, neural networks would be limited to modeling linear relationships, which are far less effective for real-world tasks.

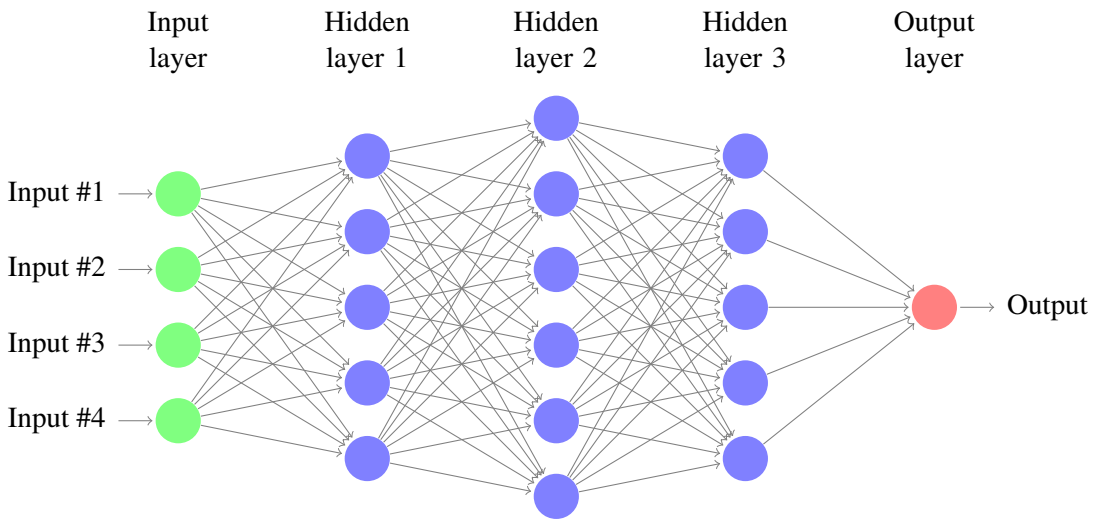


Figure 1.8 – **A schematic representation of a feed-forward neural network architecture** with three fully-connected hidden layers. Each circle represents a neuron and each line represents a connection between neurons. The green circles denote the input layer, the blue circles denote the hidden layers, and the red circle denotes the output layer. Each neuron in a layer is connected to all neurons in the subsequent layer, illustrating the complexity and interconnected nature of deep learning models. Importantly, these connections are not simply linear summations. Instead, they often incorporate non-linear functions, allowing the network to learn complex patterns and relationships within the data.

Convolutional Neural Networks (CNNs) were first proposed by Fukushima [1980] for image classification tasks. LeCun et al. [1989] demonstrated their wide applicability by successfully applying them to handwritten digit recognition. This innovation enabled the development for more complex models, such as AlexNet [Krizhevsky et al., 2012], which demonstrated the power of deep learning with large-scale data by winning the ImageNet [Deng et al., 2009] competition in 2012. AlexNet's architecture (illustrated in Figure 1.9), comprising eight layers, contained around 60 million parameters. **This substantial parameter count was pivotal for capturing the intricate patterns within the vast ImageNet dataset**, marking a significant leap in architecture complexity compared to earlier models.

Further advancements included the introduction of Long Short-Term Memory (LSTM) networks [Hochreiter and Schmidhuber, 1997], which addressed the challenge of learning long-term dependencies in sequential data. LSTMs introduce a memory cell that can maintain its state over time, and gating units that regulate the flow of information into and out of the cell, making them particularly suited for tasks such as language modeling and time-series prediction.

Generative Adversarial Networks (GANs) enable the generation of realistic synthetic data through adversarial training [Goodfellow et al., 2014]. This subfield excels in creating entirely new data, including text, images, and even videos. The synthetic data produced by GANs closely resembles real-world data, blurring the boundaries between the artificial and the authentic.

The inherent complexity of neural networks, while being the foundation of AI's successes, makes it difficult, when not impossible, to fully understand their inner workings. **The complex interaction between layers, weights, and non-linearities creates a "black-box" that hinders**

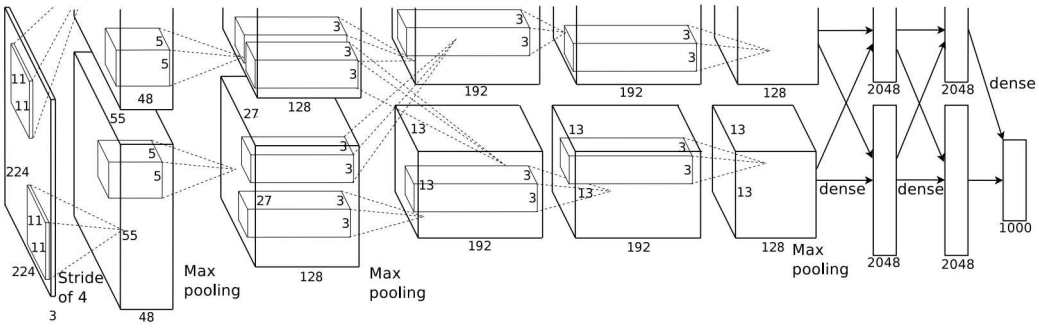


Figure 1.9 – **AlexNet's architecture** includes 60 million parameters and 650,000 neurons. It features five convolutional layers and three fully connected layers. These layers are connected by non-linear activation functions, typically ReLU (Rectified Linear Unit), and interspersed with max pooling layers for dimensionality reduction. Figure from Krizhevsky et al. [2012].

**the understanding of how these models arrive at their predictions.** As neural networks become more complex, they also become more efficient and consequently spread more and more in various fields.

### 1.2.2 Transformers

This section briefly outlines the key concepts of transformers. A more focused explanation, including the introduction of additional related concepts, will be provided with technical details in Chapter 6. Recently, the attention mechanism [Bahdanau et al., 2015], transformed neural networks by enabling them to selectively attend to different segments of input sequences. The Transformer model, introduced by Vaswani et al. [2017], builds upon this advancement. **Transformers quickly emerged as the state-of-the-art architecture, particularly in natural language processing tasks.** However, their versatility extends beyond this domain, finding applications in various other fields due to their effectiveness in capturing long-range dependencies and handling sequential data. Transformers have demonstrated remarkable applicability in Generative AI, where their ability to model complex relationships within data has led to unprecedented advancements. This has nowadays reached a level where the distinction between synthetically generated and real-world data is virtually impossible (Metz [2022]). As a result, transformers have emerged as the go-to technology for various domains, opening up new frontiers in image and video generation to text and audio synthesis.

Large Language Models (LLMs) [Devlin et al., 2019, Yang et al., 2019, Brown et al., 2020, Zhao et al., 2023] exemplify these advancements. These powerful models harness the power of increasingly larger architectures, such as transformer-based neural networks. Transformers are able to process vast datasets efficiently, enabling LLMs to decipher intricate relationships within the data and generate outputs that achieve human quality. This confluence of advancements has recently culminated in the emergence of groundbreaking applications. Popular LLMs include GPT-3 (Generative Pre-trained Transformer) counting 175 billion parameters [Achiam et al., 2023], and PaLM (Pathways Language Model), with 540 billion parameters [Chowdhery et al., 2023]. This trend of increasing model capacity over time is illustrated in Figure 1.10, where the red dot is the AlexNet architecture illustrated in Figure 1.9.

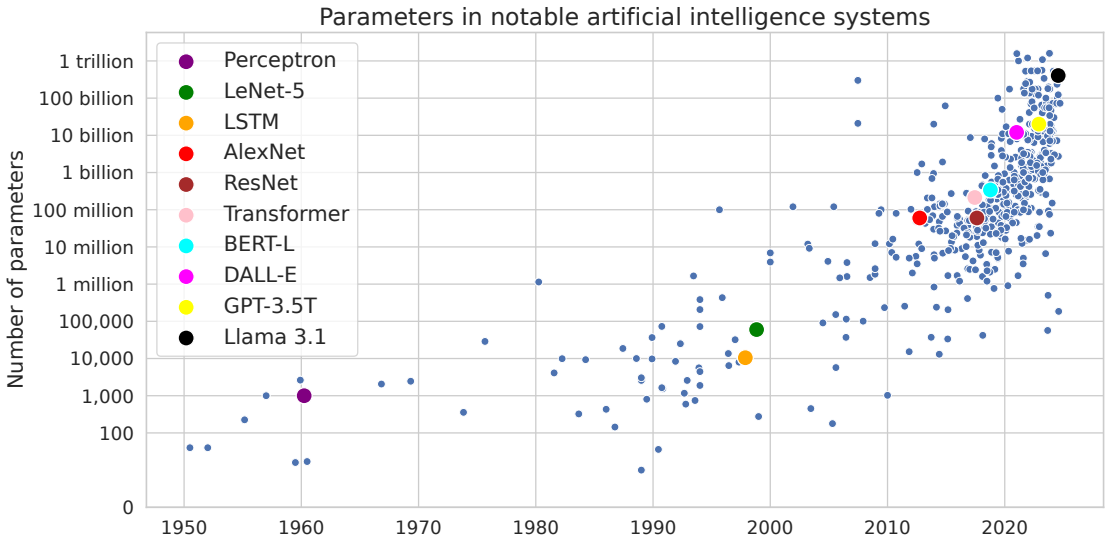


Figure 1.10 – **Evolution of the number of parameters in notable artificial intelligence systems from 1950 to 2024.** The figure illustrates the exponential growth in model complexity, ranging from early models such as the Perceptron and LeNet-5 to modern architectures like Transformer-based models including BERT-L, DALL-E, and the latest versions of GPT-3.5T and Llama 3.1. This trend highlights the increasing capacity of AI systems, which correlates with improvements in performance and the need for enhanced interpretability to ensure transparency and trust in high-stakes decision-making scenarios. Data from Epoch [2024].

Notably, ChatGPT [Bahrini et al., 2023], the first widely used AI chatbot, brought artificial intelligence to the forefront of public debate due to its widespread adoption (see Figure 1.1) and impressive capabilities. Following its success, other chatbots have emerged, such as Copilot, also based on GPT, and Gemini, which relies on PaLM. In addition, text-to-image generation using transformer architectures has emerged as a focal point. Applications like Stable Diffusion [Rombach et al., 2022], Midjourney [Oppenlaender, 2022], and DALL-E [Borji, 2022] have enabled users to convert textual descriptions into compelling visuals. The innovation continues unabated, with the advent of text-to-video generators like Sora [Liu et al., 2024], which showcase the ability to synthesize realistic video content from textual prompts. Driven by commercial applications, the development of efficient and accessible AI tools remains a key focus for developers. Companies are actively exploring lightweight versions of these powerful models, aiming to achieve similar capabilities with lower computational requirements.

The rise of Transformer models has been an even greater double-edged sword. On the one hand, their growing complexity has fueled remarkable progress in artificial intelligence. On the other hand, it has further complicated the already difficult challenge of interpretability. As these models become more intricate, **understanding their inner workings and decision-making processes becomes increasingly difficult. Concurrently, as they became more widespread, interpretability become more urgent.**

### 1.2.3 Black-boxes

The rapid transition from linear models and decision trees, described in Section 1.1.3, to increasingly complex neural networks (as seen in Figures 1.8 and 1.9), and eventually to transformers and large language models (LLMs), has rendered these models opaque and difficult to understand [Selbst and Barocas, 2018]. These complex models are often referred to as "black-boxes" [Benítez et al., 1997], because their decision-making processes are not easily interpretable.

**The term "black-box model" in machine learning describes systems whose internal workings are either too complex to understand or are not accessible** [Lipton, 2018]. Deep neural networks, for example, fall into this category. As shown in Figure 1.9, it is not straightforward to decipher the internal processes that lead from inputs to outputs. While these models can make highly accurate predictions, the process of transforming input data into output is not easily interpretable or transparent.

Additionally, a black-box model could be a proprietary system owned by a third-party company. For business reasons or due to intellectual property rights, the company may not disclose the model's internal algorithms, training data, or feature importance. Users can input data and receive predictions via an interface, but the underlying decision-making process remains hidden. This could easily be the case in Example 1.1.1 on the loan application: the bank might have used third-party models, for example, to calculate the credit score or risks (see Figure 1.2), and therefore does not have full access to the model. In this manuscript, the term *black-box* will be used to refer to any model that is not inherently explainable, without distinguishing between specific cases.

## 1.3 From AI concerns to the *right to explanation*

Understanding the broader context and implications of AI is essential for grasping its full impact on society and technology. This section briefly introduces (Section 1.3.1) the risks and implications of the AI innovations discussed in the previous section, highlighting potential challenges and unintended consequences. Then, Section 1.3.2 illustrates the attempts of public institutions in many countries to regulate their applications, emphasizing the importance of governance and ethical considerations in the deployment of AI technologies. These discussions provide valuable insights into the technical implications observed in similar regulatory efforts, hinting at how societal needs and requirements shape the development and implementation of AI systems.

### 1.3.1 AI risks and concerns

It has already been noted that the proliferation of AI presents a range of risks and concerns spanning ethical, economic, and social dimensions. Specifically, Section 1.1.2 discussed how AI can perpetuate biases and discrimination if not properly managed, leading to unfair treatments and decisions, and posing serious ethical risks.

Despite incredible performance on many benchmarks (see Figure 1.7), AI often makes glaring errors in real-world cases [Hutson, 2022], failing in practical tasks and showing significant biases, such as difficulty in correctly recognizing the faces of dark-skinned women. A striking example of this is a case from 2021, when an AI system mistakenly labeled videos of black men as "primates" [BBC News, 2021], raising concerns about racial biases in AI. A related example is gender bias in automatic translations, where algorithms often perpetuate gender stereotypes [Zhang et al., 2023b].

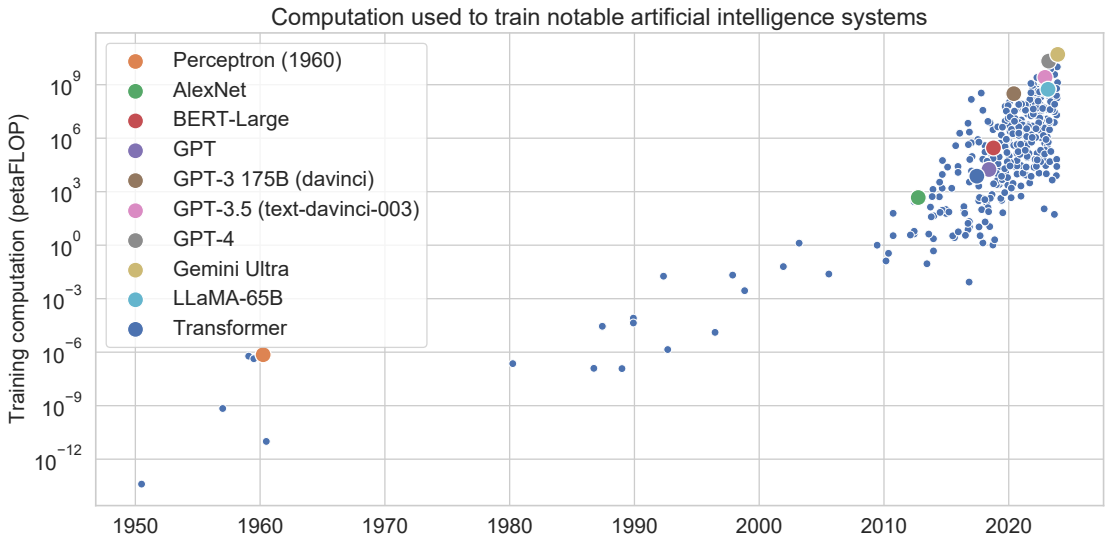


Figure 1.11 – **Computation employed in training notable artificial intelligence systems**, quantified in total *petaFLOP* ($10^{15}$ floating-point operations). Data from Epoch AI [2024].

The study shows that roles like CEO, scientist, or engineer are translated as male, while roles like nurse, baker, or wedding planner are translated as female, demonstrating an **inherent bias in translation systems. Interpretability is a key tool to mitigate these phenomena and ensure fairness** [Caton and Haas, 2024, Berk et al., 2021].

However, human intervention is often necessary to clean the data from the start. This has also led to different sort of problems: In January 2023, an investigation [Perrigo, 2023] revealed that OpenAI employed underpaid workers to make ChatGPT less toxic. These workers had to label texts containing extremely violent and sexual content to train the AI to recognize and filter such content. The reliance on low-cost labor and the precarious working conditions of labelers raise serious ethical issues in the AI industry. Additionally, recent developments also pose economic challenges: the rapid advancement of AI technology threatens unemployment and economic inequality, as automated processes replace human labor in various sectors. Some argue that technology creates more jobs than it eliminates (Di Battista et al. [2023], Frick [2024]), while other economists are rethinking this, focusing on income inequality and the need for supportive policies [Lane and Saint-Martin, 2021]. There are also warnings that the benefits of AI might not be equally distributed, potentially increasing inequalities (Georgieva [2024]). Jarrahi [2018] supports the idea of human-AI collaboration to improve productivity; Salvagno et al. [2023] explores the use of ChatGPT for generating high-quality scientific writing.

Another current issue is the so-called “hallucinations”: responses generated by AI that contain false or misleading information presented as facts [Zhang et al., 2023b]. **This phenomenon is particularly concerning as it can undermine trust in AI-based technologies.** Additionally, AI systems are becoming increasingly sophisticated in generating fake content that appears convincingly real (Williams [2024]). One of the most alarming examples of this is the creation of deepfakes, where AI is used to produce realistic but fake news, videos and audio recordings [Mirsky and Lee, 2021, Verdoliva, 2020]. These deepfakes can be so convincing that they are impossible to distinguish from authentic content, posing significant risks to privacy, security, and public trust.

The proliferation of deepfakes and other AI-generated fake content can heavily manipulate public opinion and spread misinformation (Bueermann and Perucica [2023]).

A recent report by AI Forensics investigates the use of generative AI imagery in the 2024 French political campaigns (Schueler et al. [2024]). Every case of detected AI-generated imagery has been curated to dramatize narratives through aesthetic choices, employing exaggerated storytelling tactics. These images focus on issues through a politically biased lens, depicting "factually misleading imagery intended to enforce radicalized ideologies." According to Europol [2022], "as much as 90% of online content may be synthetically generated by 2026." The implications are profound, as highlighted by various experts and researchers who emphasize the urgent need for regulatory frameworks and advanced detection tools to combat the spread of AI-generated misinformation [Marsden and Meyer, 2019].

Moreover, due to the enormous computational resources required to train these large models (see Figure 1.11), a critical issue is the significant energy consumption associated with training neural networks and transformers. Strubell et al. [2020] and Thompson et al. [2023] highlight the environmental impact and energy demands of deep learning models, emphasizing the need for policy changes to address these concerns, while Patterson et al. [2021] estimate the carbon footprint of transformer models. Narayanan et al. [2021] analyze the energy efficiency of training language models on GPU clusters, offering strategies to reduce consumption.

**There is ongoing debate about how much these models should be allowed to grow and the extent to which their associated risks should be accepted.** Bender et al. [2021] recommend prioritizing the evaluation of environmental and financial costs, and suggest investing resources in curating and carefully documenting datasets rather than indiscriminately ingesting everything available on the web.

Lastly, AI is widely used in military applications (Effoduh [2021]), including the development of lethal autonomous weapons, espionage and intelligence operations, and immigration control [Morgan et al., 2020, Longpre et al., 2022, Adam, 2024]. The use of AI in these areas is already critical and debatable, introducing additional issues of transparency, accuracy, robustness, and control. **Who is responsible for a potential AI error?**

### 1.3.2 Right to explanation

Following growing public concern, **institutions are slowly attempting to regulate critical AI applications**, emphasizing the need for trustworthiness. This trend was pioneered by the European Union's "General Data Protection Regulation" (GDPR), implemented in 2016 (EU [2016]). **The GDPR introduced the concept of a "right to explanation," recommending that individuals be informed of the rationale behind automated decisions that impact them**, particularly when these decisions deny access to something of value (like a loan) based on the information they provided. While the GDPR does not impose a direct legal obligation for explaining automated decisions [Wachter et al., 2017a], it has introduced the principle, paving the way for future regulations in this area. Subsequently, in 2019, the European Commission released the "Ethics Guidelines for Trustworthy AI" [European Commission, 2019]. This influential document outlines three key pillars that AI systems should strive for throughout their lifecycle: they should be lawful (*i.e.,* compliant with all applicable laws and regulations), ethical (adhering to ethical principles and values to minimize bias and ensure fairness), and robust (from both technical and social aspects, to minimize unintended harm and ensure reliable operation). The principle of **explainability is defined in the text as a fundamental principle for Trustworthy AI**, stating that **the processes behind AI deci-**

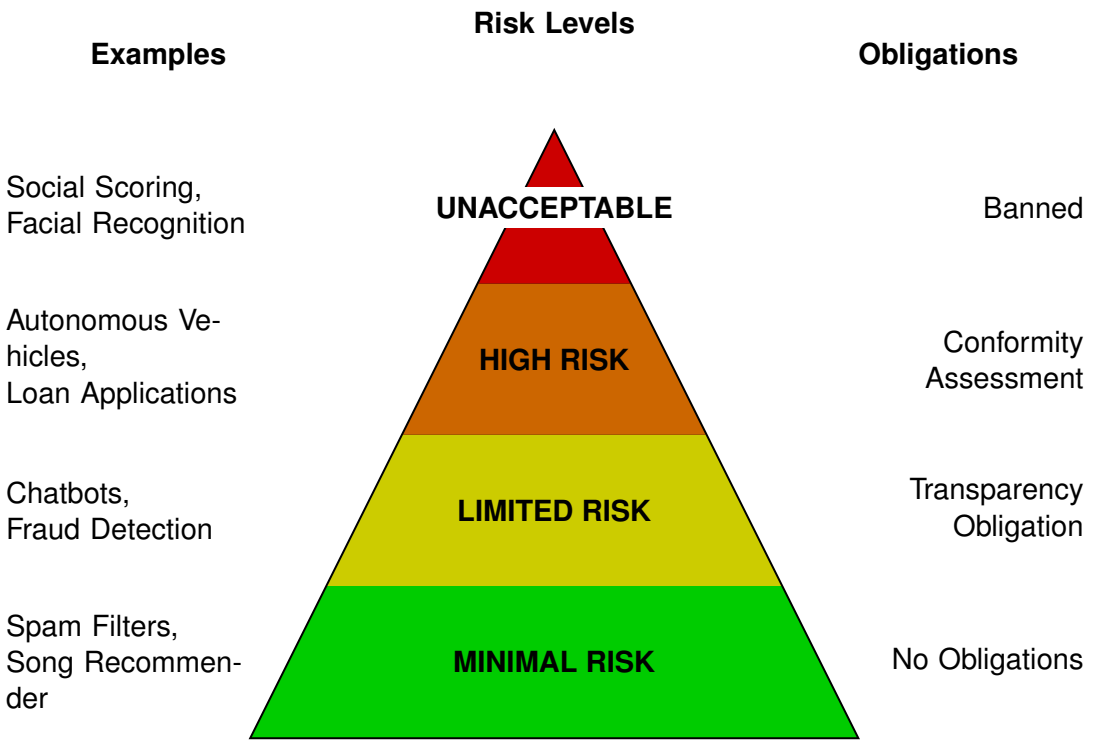


Figure 1.12 – **The EU AI Act classifies AI systems by risk level**, with proportional corresponding regulatory approaches. Unacceptable risk applications, like social scoring and facial recognition, are banned due to their threat to fundamental rights. High-risk, such as recruitment software, requires strict oversight to mitigate potential harm. Limited-risk, exemplified by chatbots, necessitates transparency obligations. Finally, minimal-risk applications, like music recommenders, face no regulations due to their negligible potential harm.

**sions should be clear and understandable, and, automated decisions should be explained to those directly and indirectly affected.**

In October 2023, the USA Administration issued a significant executive order on Artificial Intelligence [Harris and Jaikaran, 2023]. This order aimed to balance promoting innovation in AI with mitigating the potential risks of the technology. It directed federal agencies to address these risks while using AI themselves and established new safety requirements for developers of advanced AI systems.

More recently, after a long technical, political, and legal journey, in March 2024, the European Parliament approved the AI Act. Its final draft (EU [2024]) has just been released in July 2024: **the first real regulation on artificial intelligence will come into effect on August 1, 2024**. This legislation aims to strike a balance between fostering innovation in AI and mitigating potential risks associated with the technology. The core principle of the AI Act lies in classifying artificial intelligence systems based on the level of risk they pose (see Figure 1.12 for a schematic representation):

1. *Unacceptable risk*: the Act institutes a complete ban on certain critical applications, including social scoring systems that categorize people based on socio-economic factors and real-time biometric identification systems like facial recognition. Exceptions for law enfor-

cement might be granted in limited circumstances, such as for national security or severe criminal investigations.

2. *High-risk*: AI systems that could potentially endanger safety or fundamental rights, such as those used in critical infrastructure management or employment decisions, are subject to stricter regulations. These systems require thorough assessments before deployment and must be registered in a central EU database. Additionally, **the Act emphasizes transparency, granting users the right to challenge AI decisions** and demanding clear labeling of AI-generated content like images or videos (*e.g.,* deepfakes).
3. *Limited risk*: systems that may raise some ethical concerns, but the overall risk to individuals is considered low. Examples might include AI-powered fraud-detection systems, customer service chatbots, and recommendation engines. The focus is on **ensuring these systems are fair and unbiased in their operation**, minimizing potential harm to users.
4. *Minimal risk*: systems that pose minimal risk to individuals and typically require little to no oversight from regulatory bodies. Examples include spam filters, image recognition software, AI-powered games, weather forecasting models.

The EU AI Act is a pioneering piece of legislation that sets a global precedent for the regulation of artificial intelligence. It acknowledges the transformative power of AI, while also recognizing the potential risks associated with its misuse. By classifying AI systems based on their level of risk, the Act ensures that high-risk applications are subject to stringent oversight, including regular audits and compliance checks, thereby safeguarding public interest and individual rights.

The aforementioned Ethics Guidelines for Trustworthy AI, which form the foundation of the AI Act, are instrumental in shaping the ethical landscape of AI development and deployment. Grounding these guidelines are seven key principles:

1. *Human agency and oversight*: AI should empower humans and respect fundamental rights. This means employing ***human-in-the-loop*** **methods to ensure human involvement in decision-making.**
2. *Technical robustness and safety*: AI systems must be robust, secure, and accurate. Developers should incorporate fail-safe mechanisms and prioritize data quality to minimize unintended harm.
3. *Privacy and data governance*: respect of user privacy and data protection. This requires robust data governance practices to ensure data quality, integrity, and legitimate access.
4. *Transparency*: people should understand how AI systems work, and **their decisions should be explained in a manner adapted to the stakeholder concerned.** Users should be aware they are interacting with AI and understand its capabilities and limitations.
5. *Diversity, non-discrimination and fairness*: **AI should be inclusive and avoid bias**, promoting equal access for all. **Developers must actively identify and mitigate potential biases in data and algorithms to prevent marginalization of vulnerable groups.**
6. *Societal and environmental well-being*: AI advancements should benefit all humans, present and future. This means developing sustainable and environmentally friendly systems that consider the broader social and environmental impact.
7. *Accountability*: clear mechanisms for accountability are crucial. Developers should design auditable systems, allowing for assessment of algorithms, data, and design processes. Additionally, accessible grievance procedures must be established.

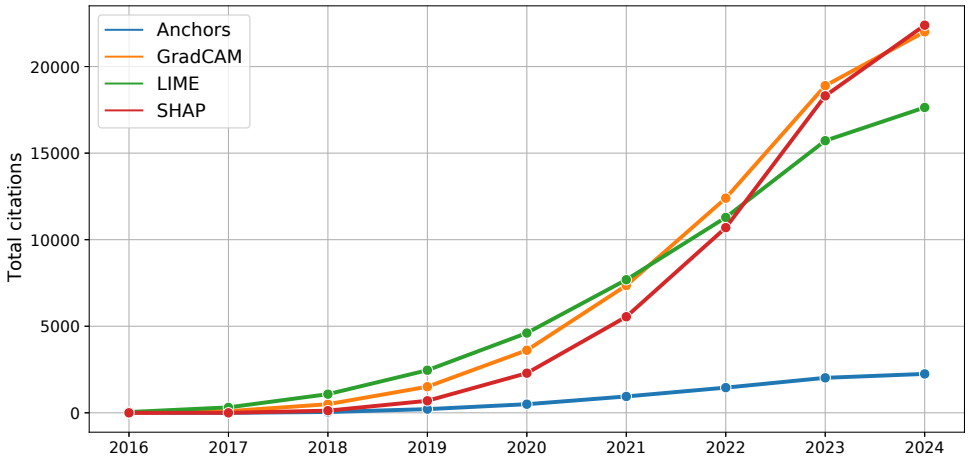


Figure 1.13 – **Cumulative citations over time for key XAI papers according to Google Scholar.** The graph includes Anchors [Ribeiro et al., 2018], GradCAM [Selvaraju et al., 2017], LIME [Ribeiro et al., 2016], and SHAP [Lundberg and Lee, 2017]. Last update: 7th June 2024.

**The newly enacted AI regulation emphasizes transparency as a cornerstone**, sparking discussions on the significance of explainability across both high-risk and minimal-risk systems [Panigutti et al., 2023]. However, the discourse is far from settled. Numerous pivotal questions remain unanswered, including the criteria for adequate transparency and the tools necessary for effective human oversight, leaving much to be clarified in the practical implementation of these principles. One of the most pressing inquiries pertains to the adequacy of existing interpretability methods in satisfying the stringent transparency requirements stipulated by the regulation. Popular techniques such as LIME [Ribeiro et al., 2016], SHAP [Lundberg and Lee, 2017], GradCAM [Selvaraju et al., 2017], and Anchors [Ribeiro et al., 2018] are widely adopted to provide insights into model decisions after they have been made. However, **it is not yet clear whether these methods are suitable for meeting the new requirements.**

While these methods can elucidate the reasoning behind complex models, their explanations are restricted to specific instances and may not capture the global behavior of the model. On the other hand, inherently interpretable models, such as the decision trees or linear regression discussed in Section 1.1.3, offer transparency by design, making them easier to understand and interpret. However, **these models often sacrifice predictive performance for interpretability, which might not be desirable in high-stakes applications where accuracy is paramount.** There are essentially two main approaches to addressing this issue: developing methods capable of explaining the predictions of existing complex models or prioritizing the design and use of inherently interpretable models. Each approach has its own set of trade-offs and potential impacts on the deployment of AI systems in critical domains.

## 1.4 Introduction to Machine Learning Intepretability

***Machine Learning Intepretability*, popularly referred to as eXplainable AI (XAI), aims to make AI systems' decision-making processes understandable to humans.** It focuses on enhancing the transparency and interpretability of AI models, addressing the *black-box* nature of modern automated systems. The research field of XAI has grown immensely and continues to

expand rapidly, becoming a dynamic and prolific area of research [Guidotti et al., 2018b, Adadi and Berrada, 2018, Linardatos et al., 2021, Bodria et al., 2023]. Every day, dozens of new scientific papers are published, exploring novel methodologies and applications of interpretability across various domains of artificial intelligence. To give an idea of the field's breadth, the surveys by Guidotti et al. [2018b] include 143 references, Linardatos et al. [2021] 165, and Bodria et al. [2023] counts 177 references. Just on the specific sub-field of concept-based interpretability, Poeta et al. [2023] has 126 references. Figure 1.13 illustrates how this topic is increasingly trending in research. Additionally, the implementation of these methodologies is evident in many toolboxes and, as seen above, in public debate (The Economist [2024b]) and regulation.

Given this vast landscape, this overview does not aim to be exhaustive. For comprehensive coverage, refer to recent surveys [Linardatos et al., 2021, Bodria et al., 2023] and the book by Molnar [2020]. Instead, this section presents a brief taxonomy and key methods that will be used throughout this thesis.

Interpretability methods in machine learning can be classified based on several criteria, including the scope of explanation (*global* vs. *local*, discussed in Section 1.4.2), the timing of their application (*intrinsic* vs. *post-hoc*, elaborated in Section 1.4.3), their dependence on the underlying model (*model-specific* vs. *model-agnostic*, detailed in Section 1.4.4), and the format of the explanation (*rule-based* vs. *feature attribution*). These factors significantly influence their behavior and characteristics.

### 1.4.1 Terminology

The field of Explainable AI grapples with a fundamental challenge: the lack of a universally accepted definition for a good explanation. This ambiguity arises because **different stakeholders have conflicting needs and expectations when it comes to interpretability.** One influential non-mathematical definition, proposed by Miller [2019], posits that interpretability hinges on "the degree to which a human can understand the cause of a decision." Further complexities emerge when considering the relationship between terms like *explainability, interpretability,* and *explanation.* Indeed, the International Organization for Standardization (ISO) defines [for Standardization, 2020] interpretability as the "level of understanding how the underlying (AI) technology works", while *explainability* as the "level of understanding how the AI-based system came up with a given result." This manuscript adopts the convention proposed by Miller [2019] and Molnar [2020], using *interpretable* and *explainable* interchangeably, while reserving the term *explanation* for the specific output generated by explaining an individual prediction.

### 1.4.2 Global *vs.* local

A critical distinction in interpretability methods lies in their scope of explanation. *Global interpretability* refers to methods that aim to explain the overall behavior of the model and how it arrives at its predictions across the entire dataset. Examples include *Interpretable Decision Sets* [Lakkaraju et al., 2016] and Setzu et al. [2020], both of which generate *rule-based* global explanations but in contrasting ways. Lakkaraju et al. [2016] propose an *interpretable-by-design model*, inherently compromising between accuracy and explainability. Conversely, Setzu et al. [2020] focus on generating rules that globally approximate any already existing black-box model. This method follows a "local-to-global" approach, where it aggregates insights from individual predictions to construct a generalized explanation summarizing the overall decision logic of the

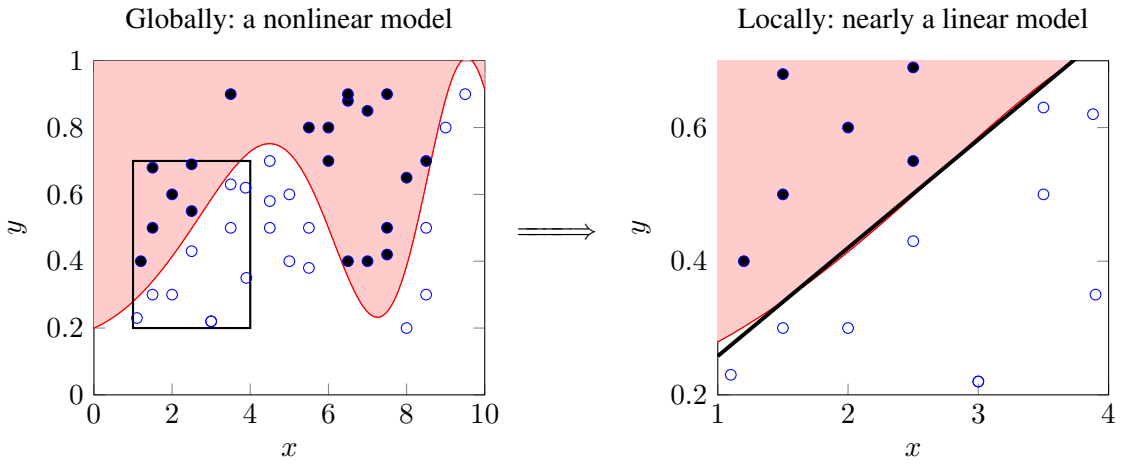


Figure 1.14 – **Illustration of local interpretabiliy.** Even for complex, non-linear models, a simpler, linear approximation can be achieved by focusing on a local region. The red line represents the decision boundary, with points above classified as positive (blue dots) and those below as negative (white dots). The right panel represents the black rectangle in the figure on the left. The black line on the right is the local linear approximation of the model, making the model's decisions more understandable and transparent by 'zooming in' on a specific area, transforming global complexity into local simplicity.

black-box model. Similarly, [Setzu et al., 2021], combines local and global explanations, further illustrating the diverse approaches to achieving global interpretability. Additionally, methods discussed by Guidotti et al. [2019,0,0] and Albini et al. [2020] enhance global interpretability by providing factual, counterfactual (Section 1.4.9), and relation-based counterfactual explanations for Bayesian network classifiers. Anjomshoae et al. [2020], propose Contextual Importance and Utility (CIU) to explain predictions.

In contrast, ***local interpretability*** **focuses on explaining the reasoning behind individual predictions** [Linardatos et al., 2021, Bodria et al., 2023]. This granular understanding is crucial for several reasons. Local explanations can help identify potential biases or errors in the model's decision-making process for a specific instance. Consider again the Example 1.1.1. If a loan application is denied for an individual with a good credit score, a local explanation could reveal that the model placed undue weight on a particular feature (*e.g.*, zip code) that might be correlated with socio-economical factors.

**A common idea is to locally approximate any complex black-box model with a simpler, inherently interpretable model for a specific prediction**. In essence, complex models like those mentioned in Section 1.2 can be accurately approximated locally by simpler models like those presented in Section 1.1.3, with the significant advantage that the latter are interpretable and can therefore provide insights into the model's behavior on the instance of interest. This idea is illustrated in Figure 1.14.

Several approaches fall under the umbrella of local interpretability methods. LIME, short for *Local Interpretable Model-Agnostic Explanations* [Ribeiro et al., 2016], is arguably the pioneering method of this category. **LIME approximates the complex model locally around a specific instance by building a simpler, interpretable model that can explain the prediction for that**

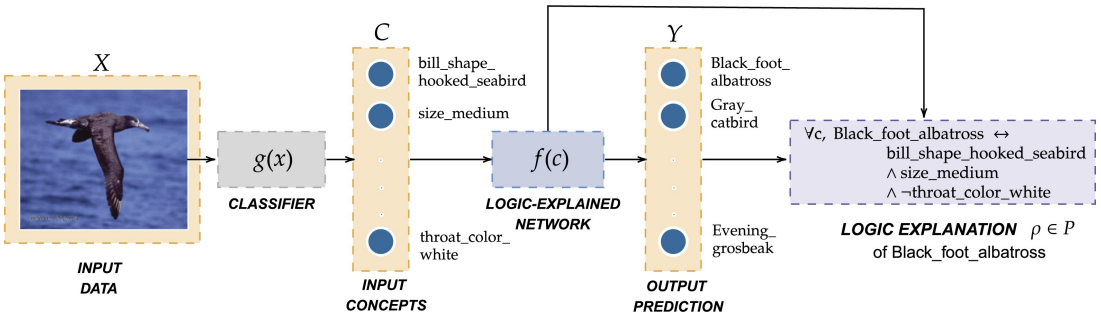


Figure 1.15 – **Architecture of Logic Explained Networks.** A LEN is applied atop a convolutional neural network to classify the bird species and provide explanations for the classification. Fig. 2 in Ciravegna et al. [2023].

**instance** (by default, a linear model). Similarly, SHAP [Lundberg and Lee, 2017] assigns importance scores to each feature based on their contribution to the model's prediction for a specific instance. This allows for an understanding of which features were most critical in driving the model's prediction for that data point.

Additionally, **Anchors provide concise rule-based explanations that capture the reasoning behind a prediction, balancing between local and global interpretability** [Ribeiro et al., 2018]. Anchors optimize for both providing the most concise rule for an individual prediction and ensuring a broad applicability across the dataset, thus achieving global coverage. This nuanced balance is further explored in Chapter 4. Amoukou and Brunel [2022] propose an explainer based on *Minimal Sufficient Rules*: a concept akin to Anchors. This explainer is capable of handling continuous features in regression tasks, thereby eliminating the need for the discretization of continuous features, a process that can be prone to errors and is often necessitated by other sampling-based explainers. A similar method is LORE [Guidotti et al., 2018a], which uses a decision tree as local surrogate.

Some popular local methods based on feature attribution [Ribeiro et al., 2016, Lundberg and Lee, 2017], are also leveraged for global explanations. This is achieved by aggregating feature importance weights across the entire dataset. This results in assigning an importance score to each input feature, reflecting its average influence on model predictions for the entire data collection.

### 1.4.3 Explainable by design *vs.* post-hoc

Furthermore, XAI methods can be divided into two main subcategories. One area of research focuses on developing models that are *interpretable/explainable by design.* Meanwhile, another area explores *post-hoc* methods for analyzing already trained models without altering their architecture. As said in Section 1.1.3 that some models, such as small decision trees and linear models, are intrinsically interpretable thanks to their simple structure. However, these do not achieve high accuracy on many tasks and have not demonstrated a good ability to generalize well from data, being effectively surpassed by more complex architectures as those presented in Section 1.2.

**Explainable-by-design architectures aim to strike a balance between model accuracy and interpretability.** The already mentioned *Interpretable Decision Sets* [Lakkaraju et al., 2016], for example, are a way to build machine learning models that are both accurate and easy to understand. This is achieved by using a set of independent `IF-THEN` rules, where each rule explains a

specific prediction condition. This makes them simpler and more interpretable than complex models, while still maintaining high accuracy. Ross et al. [2017] proposes training neural networks with constraints on input gradients to ensure models make decisions based on relevant features. Senetaire et al. [2023], instead, proposes a modular self-interpretable probabilistic model class that allows for instance-wise feature selection.

Explainable Boosting Machine (EBM, Nori et al. [2019]) and Neural Additive Models (NAM, Agarwal et al. [2021]) are both designed to balance performance with interpretability as advanced variants of Generalized Additive Models (GAMs). EBM employs a boosting algorithm to iteratively train feature functions, allowing each feature to contribute additively to the final prediction. NAM combines deep neural networks with the interpretability of GAMs by training separate subnetworks for each feature. Continued Fractions Nets (CoFrNets, [Puri et al., 2021]) use continued fractions to represent neuron outputs, enabling precise computation of feature contributions.

Logic Explained Networks (LENs, [Ciravegna et al., 2023, Barbiero et al., 2022]) is a concept-based method [Poeta et al., 2023] that ensure the generated explanations are directly connected to the model's internal logic by integrating "explainability layers" within neural networks. An illustration is shown in Figure 1.15. In essence, **this class of models learns human-readable predicates or concepts during training, along with the model parameters.**

Conversely, ***post-hoc* interpretability methods are applied to models that have already been trained.** These methods operate on pre-trained models, where the underlying architecture and learned parameters are fixed. While direct modification of the model's architecture is not feasible, **some post-hoc approaches can still leverage access to the model's internal parameters to gain insights into its decision-making process.** These include gradient-based (discussed in Section 1.4.5) and attention-based methods (covered in Chapter 6). Additionally, there are *model-agnostic* approaches (Section 1.4.4) that do not require any information about the internal workings of the model.

Post-hoc interpretability methods also include *perturbation-based* methods (Section 1.4.6), which analyze how changes to the input data (*e.g.*, occluding pixels in an image or masking features in tabular data) affect the model's predictions. By observing how these perturbations impact the predictions, it is possible to identify which features are most influential for the model. As discussed in Section 1.4.6, LIME, SHAP, RISE, and Anchors fall into this category.

**Criticisms of post-hoc methods.** Post-hoc explanations have been criticized for lacking explicitness, faithfulness, and stability. They often do not align well with the internal logic of the model they aim to explain and can vary considerably with slight changes in the input data. Alvarez Melis and Jaakkola [2018] formalize the class of self-explainable models, proposing desiderata for explanations, and demonstrate that post-hoc methods do not meet these requirements. Similarly, Rudin [2019] critiques the use of post-hoc explanations for black-box models, arguing that these methods often fail to meet the standards required for high-stakes decisions. In particular, Rudin [2019] advocates for developing and using inherently interpretable models that do not sacrifice transparency for performance. Furthermore, post-hoc explanations particularly lack a strong mathematical foundation, as detailed in Section 1.5.3.

However, although post-hoc interpretability methods may not perfectly align with the architecture of the model being explained, they offer several significant advantages. They are highly versatile and can be applied to a wide range of (pre-trained) models, often regardless of their un-

derlying architecture [Bodria et al., 2023, Queen et al., 2024]. This allows for their use across different domains without needing significant model alterations.

Conversely, some ad-hoc methods require human intervention to define the concepts and logical constructs that the network must learn [Barbiero et al., 2022, Ciravegna et al., 2023]. This process is time-consuming and can introduce errors and biases. The need for domain-specific knowledge adds complexity and subjectivity, potentially undermining transparency and trustworthiness.

**Post-hoc interpretability methods are often the only viable option in scenarios where modifying the model architecture is not feasible due to time, resource, or commercial constraints** [Saporta et al., 2022], as mentioned in Section 1.2.3. In such cases, these methods provide a practical solution for achieving model interpretability without necessitating a complete redesign. They enable insights into the model's decision-making process after training, which is particularly useful for deployed models [Lundberg and Lee, 2017, Sundararajan et al., 2017a].

Finally, as machine learning models have evolved independently from the field of Explainable AI, many innovations have been developed without considering interpretability. In such cases, post-hoc methods remain relevant and increasingly popular (see Figure 1.13) due to their adaptability and ability to provide meaningful insights. These methods ensure continued importance in the interpretability landscape by offering practical solutions to understand and analyze models that were not initially designed with interpretability in mind [Saporta et al., 2022].

**This thesis focuses on post-hoc interpretability methods for machine learning models.** It does not address global interpretability or the design of specific architectures for training inherently interpretable models.

### 1.4.4 Model-dependent *vs.* model-agnostic

Another critical distinction in interpretability methods lies in their dependence on the underlying model. ***Model-dependent*** **methods leverage information specific to the model's architecture or internal parameters.** For instance, interpreting weights in linear models or analyzing the tree structure of decision trees are inherently model-specific approaches. These methods require a detailed understanding of the model's inner workings to explain its predictions.

Popular model-dependent approaches include DeepLIFT, introduced by Shrikumar et al. [2017], which propagates activation differences to identify important features in neural networks. Arras et al. [2017] focused on interpreting predictions of recurrent neural networks, particularly in text sentiment analysis.

In attention-based [Bahdanau et al., 2015] models such as transformers [Vaswani et al., 2017], some techniques examine the attention weights allocated to input data [Chefer et al., 2021, Mylonas et al., 2023], revealing the model's focus for prediction, as elaborated in Chapter 6. These weights can indicate where the model focuses its attention when making a prediction, theoretically highlighting the most influential input features.

Gradient-based approaches also fall under this category [Simonyan et al., 2014]. Smilkov et al. [2017] developed SmoothGrad, which enhances gradient-based methods by averaging noisy sensitivity maps to produce clearer visual explanations for image classifiers. Class Activation Mapping (CAM, Zhou et al. [2016], Selvaraju et al. [2017], Chattopadhay et al. [2018], Zhang et al. [2023a]) utilizes the gradients of a target class to create a localization map highlighting crucial image regions for predicting the label. By analyzing the activations of the convolutional layer,

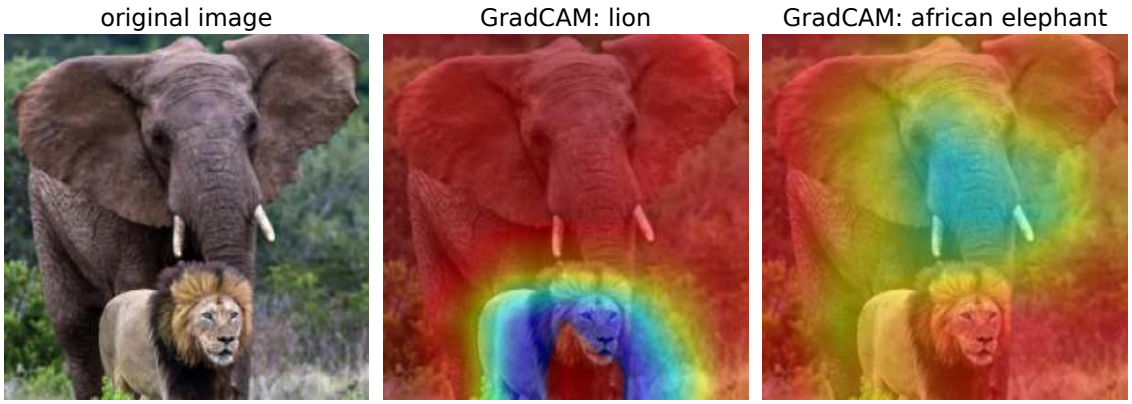


Figure 1.16 – **GradCAM results on an image classifier**, based on ResNet-50 [He et al., 2016]. The first panel shows the original image (from ImageNet, Deng et al. [2009]). The second panel highlights the most influential regions (according to GradCAM) for classifying the image as a *lion*. The third panel displays the regions most influential for classifying the image as an *african elephant*.

CAM identifies the regions of the input image that contribute most significantly to the classification decision.

In contrast, ***model-agnostic*** **methods treat any model as a *black-box*.** They do not require knowledge of the model's inner workings and rely solely on analyzing the input features and corresponding outputs. Techniques such as permutation importance, feature interaction analysis, LIME, SHAP, Anchors, LORE [Guidotti et al., 2018a], MAPLE [Plumb et al., 2018], DBA [Vlassopoulos et al., 2020] fall under this category. These methods offer the advantage of being applicable to any trained model, regardless of its underlying architecture. **The versatility of model-agnostic methods makes them suitable for a broader range of scenarios compared to other interpretability approaches**, which may either necessitate intervention during model training [Ciravegna et al., 2021, Rigotti et al., 2021] or access to specific model parameters [Selvaraju et al., 2017, Lopardo et al., 2022, Mylonas et al., 2024].

### 1.4.5 Gradient-based interpretability

The distinction between model-specific and model-agnostic methods is not always clear-cut. Gradient-based explanations, for example, can be applied across various (differentiable) architectures despite relying on internal model workings. This demonstrates a spectrum from highly specific methods to those more generally applicable. Gradient-based methods are typically local and post-hoc, offering feature importance by quantifying each input's contribution to predictions. However, they are not entirely model-agnostic as they require access to internal parameters to compute gradients. High gradients indicate significant feature influence on the model's prediction. In practice, **gradient-based interpretability methods compute the model's gradients with respect to input features to identify which features most influence the model's predictions.**

Li et al. [2016] compute the gradient $\nabla_x F(x)$ of the output $F(x)$ with respect to each input feature $x_1, \ldots, x_d$, highlighting features where small changes in input lead to large changes in output. Sometimes, $x_1, \ldots, x_d$ are vectors themselves (for instance, they are embedding vectors in the case of language models). To derive per-token importance weights, several strategies exist.

The primary approaches involve taking the mean value [Atanasova et al., 2020], the $L^1$ norm [Li et al., 2016], or the $L^2$ norm [Poerner et al., 2018, Arras et al., 2019, Atanasova et al., 2020] of $\nabla_{x_1} F(x), \ldots, \nabla_{x_d} F(x)$. *Gradient* $\times$ *Input* [Denil et al., 2014], instead, takes the dot product between the gradient and the input feature value, $x_j \cdot \nabla_{x_j} F(x)$, emphasizing features with significant impact. However, gradients can fail to satisfy the sensitivity requirements [Shrikumar et al., 2017]: differing features between an input and a baseline with different predictions should have non-zero attributions. Sundararajan et al. [2017a] address this issue with *Integrated Gradients*, which provide more reliable attributions by averaging gradients along a path from a predefined baseline instance $\tilde{x}$ to the actual input instance $x$:

$$\text{IntegratedGradients}_j(x) = (x_j - \tilde{x}_j) \int_0^1 \nabla_{x_j} F(\tilde{x} + \alpha(x - \tilde{x}))\, \mathrm{d}\alpha\,.$$

*Expected Gradients* [Erion et al., 2019] extend Integrated Gradients by averaging the gradients over a distribution of baselines rather than a single baseline, enhancing robustness and interpretability by accounting for variability in input distributions.

Gradient-based attribution methods are effective in interpreting neural network decisions [Ancona et al., 2018], generating saliency maps that highlight important regions in images or significant words in text data [Simonyan et al., 2014, Denil et al., 2014, Li et al., 2016]. GradCAM [Selvaraju et al., 2017] uses gradient information from the last convolutional layer to create saliency maps by calculating gradients of a target class's output with respect to feature map activations, pooling these gradients to form a coarse heatmap. In Figure 1.16, GradCAM is used to produce saliency maps for explaining image classifications. GradCAM++ [Chattopadhay et al., 2018] improves upon this by providing more accurate and detailed explanations through a weighted average of pixel-wise gradients. Other variations include Score-CAM [Wang et al., 2020], which eliminates the need for gradients by using forward pass scores as weights; Eigen-CAM [Muhammad and Yeasin, 2020], which employs eigenvector projections for more uniform scores; Ablation-CAM [Ramaswamy et al., 2020], which uses ablation analysis to determine feature map importance; and CXPlain [Schwab and Karlen, 2019], which frames explanation as a causal learning task, enhancing interpretability through a causal perspective. **Despite their effectiveness, CAM-based methods have been proved to incorrectly assign importance to parts of the image that the model cannot see** [Taimeskhanov et al., 2024].

SmoothGrad [Smilkov et al., 2017] bridges gradient-based and perturbation-based explanations by reducing noise in gradient-based explanations. It does this by averaging gradients from multiple noisy input versions, leading to clearer attributions.

### 1.4.6 Perturbation-based interpretabiliy

Perturbation-based interpretability includes techniques used to understand how machine learning models make predictions by observing changes in the output when the input data is slightly modified or *perturbed.* **The basic concept involves systematically modifying parts of the input and analyzing the resulting variations in the model's predictions** [Covert et al., 2021]. Perturbation-based methods can offer both local and global explanations and are generally applicable to any type of machine learning model, including black-box models like neural networks, because they do not require access to the model's internal structure or parameters.

In general, small changes are first made to the input data. These changes can be as simple as altering the value of a single feature, adding noise, or removing certain features entirely. The

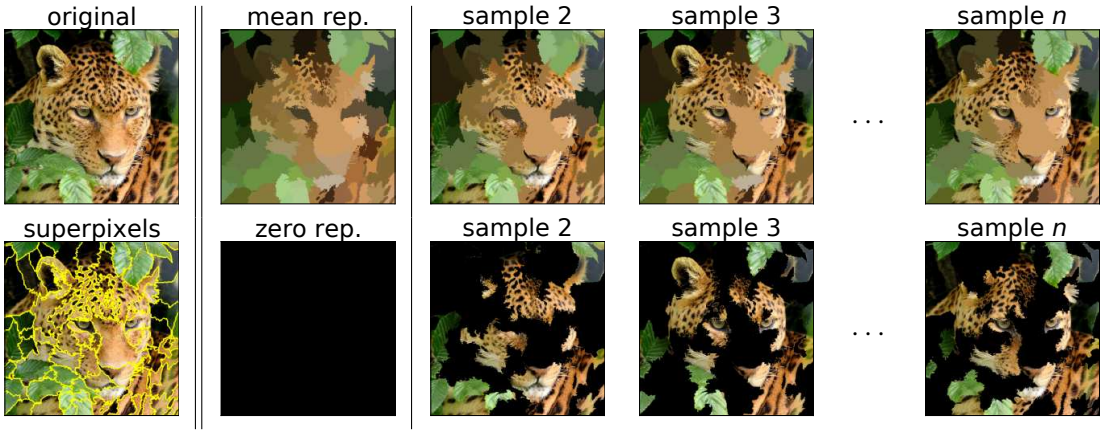


Figure 1.17 – **LIME's sampling scheme for images.** First, the original image is split into superpixels, which are contiguous segments of pixels with similar colors or other features (by default, LIME uses quickshift [Vedaldi and Soatto, 2008] for segmentation). In this example, the image is divided into 87 superpixels. The replacement image is then computed by either averaging the color within each superpixel (top row) or setting each superpixel to a zero value, *i.e.*, a predetermined color (bottom row). Finally, $n$ samples are generated by randomly replacing superpixels, with each superpixel having a replacement probability of 0.5 by default.

model's output is then observed before and after the perturbation. By comparing the differences in predictions, insights can be gained about the importance and influence of the perturbed features on the overall result. By analyzing how much and in what way the prediction changes in response to perturbations, it is possible to determine which features are most important to the model's decision-making process. **Features that cause significant changes in the output when modified are considered more important.**

LIME, [Ribeiro et al., 2016], is the pioneering perturbation-based method that popularized the concept of local, post-hoc, model-agnostic explanations for any model making predictions on structured data, text, or images. LIME perturbs the input data by generating a set of perturbed samples around the instance, observes the changes in predictions, and fits an interpretable model (by default, using linear regression) to approximate the local behavior of the complex model, thereby making it locally interpretable.

Figure 1.17 illustrates the sampling that LIME performs to explain image classifications. In this example, an image classifier (Inception, Szegedy et al. [2016]) has been trained on the ImageNet [Deng et al., 2009] dataset. The figure shows a photo of a leopard, which the model correctly classifies with high confidence. **LIME can explain which parts of the image the model focused on the most by generating a sample of perturbed images, "turning off or on" parts of the image, *i.e.*, the *superpixels*.** These are the interpretable components used as input features for the surrogate model. In the case of a linear surrogate, each superpixel essentially has a weight proportional to its importance. The same idea can be applied to text classifiers, where words are randomly removed to assess their impact on the model's prediction (see Figure 1.18).

Many works build upon the foundations of LIME, addressing its limitations and extending its capabilities [Zafar and Khan, 2019, ElShawi et al., 2019, Shankaranarayana and Runje, 2019, Bramhall et al., 2020]. SHAP (SHapley Additive exPlanations, Lundberg and Lee [2017]) and Anchors [Ribeiro et al., 2018] are two popular contenders within the domain of perturbation-based

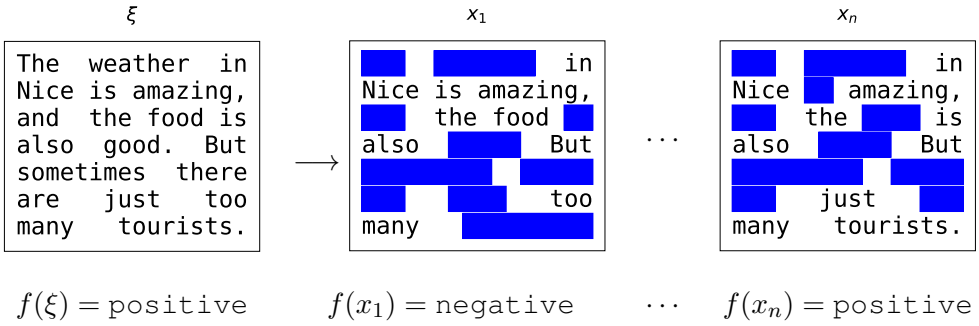


Figure 1.18 – **Illustration of the perturbation-based approach for explaining text classifiers.** The original sentence $\xi$ is perturbed by randomly removing words to create modified versions $(x_1, x_2, \ldots, x_n)$. Each perturbed instance is then evaluated using the classifier $f$, and the impact of the removed words on the prediction outcome is measured. The goal is to identify which words contribute most significantly to the model's decision.

interpretability, sharing similarities with LIME. Like LIME, both SHAP and Anchors are designed to explain the predictions of any model operating on structured data, text, or images. SHAP leverages foundations from cooperative game theory [Shapley, 1953] to assign each feature an importance value based on how it contributes to the prediction. On the other hand, Anchors focuses on identifying the smallest *anchor*, *i.e.*, a subset of features that sufficiently changes the prediction, providing a more concise explanation compared to LIME's perturbation-based approach. RISE (Randomized Input Sampling for Explanation, Petsiuk et al. [2018]) specializes in generating saliency maps tailored for image data by randomly sampling and masking regions of input images to understand model decisions at the pixel level.

Most perturbation-based explainers follow the *explaining by removing* framework [Covert et al., 2021]. **The basic idea of removal-based explanation methods is to remove a feature and observe how the model's behavior changes.** In many scenarios, the direct *removal* of features is impractical or computationally expensive. When dealing with tabular data, this would imply removing and retraining the model for every possible combination of features, resulting in $2^b$ trained models, where $b$ is the number of features, which is computationally infeasible. In computer vision tasks, the analogous solution would be to occlude one or more pixels to the model, which may not necessarily be straightforward. Therefore, in practice, feature removal is simulated using various techniques. Approaches include setting features to zero [Zeiler and Fergus, 2014, Petsiuk et al., 2018, Schwab and Karlen, 2019], using pre-defined default values [Ribeiro et al., 2016, Dabkowski and Gal, 2017]. In tabular data, this can involve setting the feature to be removed to a zero value or replacing it with the average value (*i.e.*, setting $\tilde{x}_j = 0$ or $\tilde{x}_j = \bar{x_j}$). For images, pixels (or *superpixels*) can be "removed" by setting them to black or to the average pixel value (see Figure 1.17). Alternatively, a random value from can be assigned to simulate feature removal, by marginalizing features using their conditional distribution $\tilde{x}_j \sim \mathbb{P}\left(X_j \mid X_k = x_k\,,\; k \neq j\right)$ [Lundberg and Lee, 2017].

Crucially, it becomes evident that the sampling mechanism used to construct **the local neighborhood profoundly influences the quality of the ultimate explanation.** In particular, as illustrated in Section 1.5.1, it can potentially lead to out-of-distribution (OOD) samples, which can compromise the validity of the explanation. Perturbation-based methods generate explanations by

creating a set of perturbed samples around the instance of interest and then querying the model on these samples. If the generated samples are OOD, meaning they fall outside the distribution of the training data, it violates a fundamental principle of machine learning: **the training and test data must come from the same distribution.** This violation can lead to erratic model responses and unreliable explanations.

### 1.4.7 Concept-based interpretability

**Concept-based methods focus on explaining AI decisions by identifying and presenting human-understandable concepts** rather than features or model parameters [Poeta et al., 2023]. By mapping model decisions to relatable concepts such as objects, actions, colors, shapes, or relationships, concept-based XAI allows users to comprehend not only which features contribute to a decision but also why these features are relevant in the task context.

Several approaches have been developed to implement concept-based interpretability in AI models. One prominent method is *Testing with Concept Activation Vectors* (T-CAV, Kim et al., 2018), which quantifies the influence of human-interpretable concepts on neural network predictions. T-CAV creates vectors in the model's latent space representing specific concepts and assesses how variations in these vectors affect the model's output. This method provides a measure of how much a concept contributes to a decision, offering insight into the inner workings of complex models. Another approach, *Automatic Concept-based Explanations* (ACE, Ghorbani et al., 2019), is an unsupervised method that segments images at multiple resolutions to capture a range of concepts and clusters these segments in the network's latent space, filtering out outliers. The importance of each concept is then calculated using the T-CAV score, which measures the influence of a concept on class prediction.

*Concept Bottleneck Models* (CBMs, Koh et al., 2020) introduce an intermediate layer where each neuron corresponds to a distinct human-defined concept. These models ensure that decisions are based on comprehensible concepts by predicting the concept values first, which are then used for the final prediction [Poeta et al., 2023]. Despite their transparency, a naive implementation may result in performance loss due to concept bottlenecks not capturing all relevant task-specific information. *Logic Explained Networks* (LENs, Ciravegna et al., 2021) co-learn explanations alongside neural network parameters, solving clustering tasks and providing logical explanations about cluster assignment in terms of frequent concept relations. LENs achieve transparency by predicting concept values first and using them for final predictions, though they also face potential performance trade-offs.

*Self-explanatory Neural Networks* (SENN, Alvarez Melis and Jaakkola, 2018) and *Bottleneck Concept Learners* (BotCL, Wang et al., 2023) employ unsupervised approaches to create explainable-by-design networks. SENN uses a concept encoder to derive a clustered representation of features from the input and generates class-concept relevance scores, outputting a linear combination of these basis concepts and their relevance scores. BotCL builds on SENN by using a slot attention-based mechanism to predict concept scores, ensuring consistency in learned concepts through specific regularization.

**Trade-offs between interpretability and performance vary across these approaches, with some ensuring high transparency at the cost of slight reductions in accuracy**, while others maintain competitive performance levels. *Invertible Concept-based Explanations* (ICE, Zhang et al., 2021) and *ConceptSHAP* [Yeh et al., 2020] enhance post-hoc methods by improving concept

identification and importance assessment. ConceptSHAP modifies SHAP values to reflect the sufficiency of concepts in representing the class.

Recent developments include a framework for identifying known concepts in embedding spaces, connecting with classical methods like Principal Component Analysis (PCA) [Leemann et al., 2023], and a model-agnostic method grounded in axioms of linearity, recursivity, and similarity [Feng et al., 2024], ensuring reliability and applicability across various models.

### 1.4.8 Example-based interpretability

Another relevant category is *example-based* (or *prototype-based*) interpretability. **Example-based interpretability methods offer to understand model decisions by identifying representative examples within the dataset**, often called prototypes, that the model uses to make predictions. These prototypes act as tangible references, allowing users to see concrete examples that illustrate how the model arrives at its conclusions. This approach is related to the $k$-Nearest Neighbors (Section 1.1.3), where predictions are based on the closest examples in the training data (see Figure 1.6).

Prototype-based methods such as *ProtoPNet* [Chen et al., 2019], *ProtoTree* [Nauta et al., 2021], and *Deformable ProtoPNet* [Donnelly et al., 2022] enhance interpretability in image classification tasks by explaining model decisions through image prototypes. For instance, ProtoPNet identifies prototypes within the training data that most resemble the input image. ProtoTree combines decision tree structures with prototype learning to offer a hierarchical view of how prototypes influence decisions. In natural language processing, *ProtoTex* [Das et al., 2022] applies the prototype-based approach using prototype tensors for classification. This method identifies prototypical text segments that are most representative of each class, offering an interpretable way to understand how the model processes and classifies text data.

### 1.4.9 Counterfactual explanations

Alternatively, counterfactual explanations use counterexamples (counterfactuals), which show how small changes to the input could lead to different outcomes, thus highlighting the model's decision boundaries and offering insights into its behavior under different scenarios. Counterfactual explanations are a key component of providing recourse; they offer insights into how the outcome would have been different if certain input features were altered. Essentially, counterfactuals describe hypothetical scenarios that indicate what changes a user can make to achieve a more desirable outcome, thus enabling practical recourse. An illustration is shown in Figure 1.19. **These explanations identify an alternative instance that is similar to the original input but leads to a different outcome** [Guidotti, 2022, Pawelczyk, 2024]. For example, if a loan application is denied (Example 1.1.1), a counterfactual explanation might suggest that if the applicant's income were slightly higher, the application would be approved.

LORE, *LOcal Rule-based Explainer* [Guidotti et al., 2018a], has already been mentioned: it is a local and model-agnostic method that leverages decision trees to generate post-hoc explanations. By following the specific path in the decision tree (see Figure 1.4) corresponding to the example being explained, LORE produces factual explanations. Simultaneously, it leverages different paths followed by synthetic neighbors in the local neighborhood to retrieve counterfactual rules, illustrating conditions that can be varied to change the output decision. Wachter et al. [2017b] proposed a method to identify the minimal changes needed in the input features to change the model's predic-

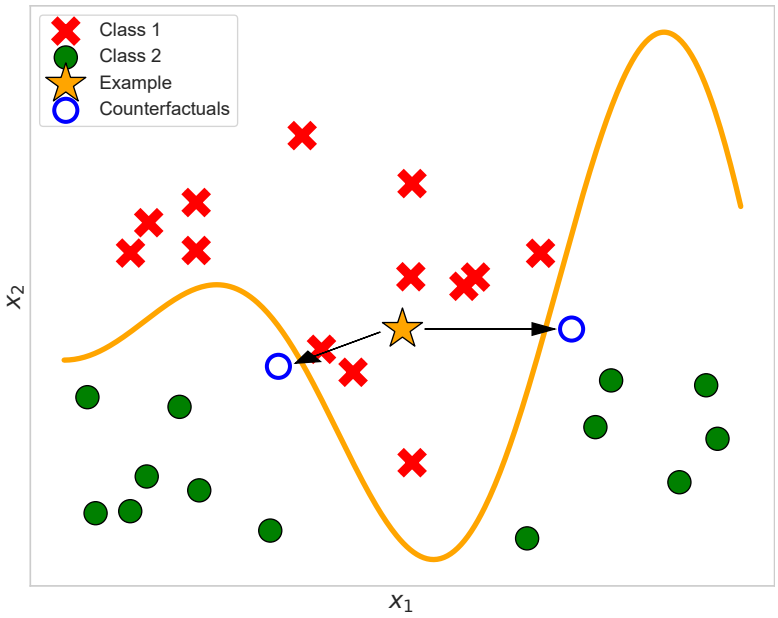


Figure 1.19 – **Illustration of counterfactual explanations in a classification scenario.** The star represents the *Example* to be explained, belonging to *Class 1*. The arrows highlight the paths to *Counterfactual* examples. The decision boundary emphasize the necessary changes for the instance to be classified differently.

tion, balancing the difference in prediction and the similarity to the original input. **Several works emphasize the importance of generating not only accurate but also plausible counterfactuals** [Wachter et al., 2017b, Pawelczyk et al., 2021,0]. Moreover, Charachon [2023] defines three properties, *Relevance*, *Regularity*, and *Realism*, which are essential for generating effective visual explanations in the context of medical image analysis. Relevance ensures that the highlighted features in the explanation directly correspond to the classifier's decision-making process. Regularity focuses on the consistency and smoothness of the explanations, avoiding noise and complexity to enhance interpretability. Realism guarantees that the explanations appear authentic and plausible within the context of the input data, thereby improving their credibility and usefulness for medical practitioners. These properties collectively aim to create explanations that are not only accurate but also interpretable and practical for end-users.

Similarly, *Contrastive Explanation Method* (CEM, Dhurandhar et al. [2018] provides both necessary factors ("pertinent positives") that must be present for a certain prediction and unnecessary factors ("pertinent negatives") that should be absent, formulating these as an optimization problem to find the smallest changes needed. C-CHVAE (short for *Counterfactual Conditional Heterogeneous Variational AutoEncoder*) generates plausible counterfactuals by ensuring they are not outliers and are close to correctly classified examples, using a variational autoencoder instead of relying solely on distance measures [Pawelczyk et al., 2020b,0]. DICE, *Diverse Counterfactual Explanations* [Mothilal et al., 2020], focuses on generating diverse and plausible counterfactuals by solving an optimization problem with constraints, ensuring multiple ways to change the outcome. FACE, *Feasible and Actionable Counterfactual Explanations*, emphasizes actionable counterfactuals by identifying feasible paths within the data distribution, ensuring the suggested changes are realistic and actionable [Poyiadzi et al., 2020].

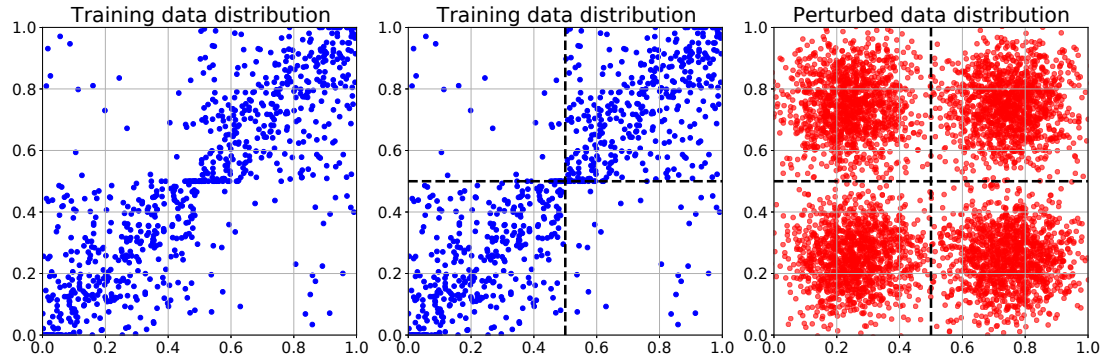


Figure 1.20 – **Illustration of the *out-of-distribution* problem in LIME for tabular data.** The *left panel* shows the training data used by a black-box model. In the *middle panel*, the feature space is split into quantiles for each feature, highlighting highly unbalanced data. The *right panel* reports the density of samples generated by LIME: it generates random samples centered in each box. Note that by default, LIME would divide each axis into quartiles.

Additionally, Albini et al. [2020] propose counterfactual for Bayesian Network Classifiers to identify key factors influencing the classification, indicating which factors need to change to alter the outcome. Abrate and Bonchi [2021] propose a method to produce counterfactual explanations for graph classifiers: it introduces counterfactual graphs, which are minimally modified versions of the original graph that result in a different classification, thereby providing insights into the critical features influencing the model's decisions in brain network classifications. Crupi et al. [2022] generate counterfactual explanations by intervening in the latent space of machine learning models. Pawelczyk et al. [2023] use generative models to generate realistic recourses, methods to ensure the robustness of these recourses against small changes. Jain et al. [2024] introduces *CAVIAR*, a novel method for generating counterfactual explanations for visual recommender systems. It identifies minimal perturbations to an item's visual features, causing it to drop from a user's top-$K$ recommended list. Finally, Pawelczyk [2024] deeply analyze the reliability of algorithmic recourse through realistic and robust counterfactual explanations.

## 1.5 Open challenges

In previous sections, it has been demonstrated that interpretability in machine learning represents one of the most pressing and fascinating challenges of contemporary AI. Despite the significant progress made in recent years, **there are still many open questions that need to be addressed to ensure that machine learning models are not only accurate but also understandable and reliable.** Among the numerous outstanding research questions, this section focuses on three main points that motivate the work presented in this thesis: the problem of *out-of-distribution* samples (Section 1.5.1), the lack of standardized evaluation metrics (Section 1.5.2), and the absence of solid mathematical foundations (Section 1.5.3).

### 1.5.1 Out-of-distribution samples

As discussed in Section 1.4.6, one of the primary approaches to explaining black-box models without altering their architecture or using their internal parameters is to perturb the instance to be explained and evaluate how the output changes as a result. This allows for the analysis of the impact of individual features on the model's predictions. However, it becomes evident that **the sampling mechanism used to construct the local neighborhood not only influences the quality of the final explanation but also leads to fundamentally different explanations.** This aspect, despite being critical, remains underexplored.

Consider a dataset comprising house sizes in square meters and the number of rooms. When perturbing a single instance, various sampling strategies can be employed. For example, one approach is setting the numerical values to zero, which would generate samples with zero rooms or zero square meters. Alternatively, using the average value or selecting a random value from the dataset could produce samples with unusual combinations, such as 10 square meters and 20 rooms or 1 room for 100 square meters.

This poses a challenge as **it may lead to out-of-distribution (OOD) samples, querying the model in parts of the feature space where it is not adequately trained** [Jacovi and Goldberg, 2021, Hase et al., 2021]. Therefore, this can violates one of the key assumptions in machine learning: the training and evaluation data come from the same distribution [Vapnik, 1998, Hooker et al., 2019]. Thus, potentially compromise the validity of the explanation. This issue is illustrated in Figure 1.20, in the specific case of LIME for tabular data.

Similarly, perturbing a text can involve completely removing tokens or words [Nguyen, 2018, DeYoung et al., 2020], or replacing them with fixed feature values [Ribeiro et al., 2016,0, Lundberg and Lee, 2017]. LIME, SHAP, and Anchors, for instance perturb the text to explain by replacing them with mask tokens like `UNK`, thus significantly alter the original context.

Sturmfels et al. [2020], Haug et al. [2021] have compared different replace functions for vision and tabular data, respectively. However, concerns about samples leading to unreliable explanations persist. Fong and Vedaldi [2017] and Mothilal et al. [2020] emphasized the need for realistic perturbations to avoid OOD issues in generating trustworthy explanations. Hooker et al. [2019] benchmarked various interpretability methods, noting significant inaccuracies due to OOD samples. Qiu et al. [2021] and Hsieh et al. [2021] proposed methods to resist the impact of OOD data but acknowledged the persistent challenge it poses. Jethani et al. [2021] demonstrated that interpretability methods could still lead to unreliable explanations when encountering OOD samples. Additionally, Slack et al. [2020] demonstrated that **adversarial attacks can exploit vulnerabilities, further undermining the reliability of these post-hoc explanation methods.** Their work revealed that by subtly manipulating the input data, one could generate explanations that are significantly misleading, highlighting a critical weakness in these popular explainability techniques.

Various solutions have been proposed to mitigate OOD samples, and their efficiency strongly depends on the setting and the application context. If the dataset used to train the model comes from the same distribution as the test data, it may more easily be leveraged to generate perturbed samples that remain approximately in-distribution. Additionally, the type of task significantly influences the outcome: in natural images, the objects of interest and the overall content can vary significantly in the database. In contrast, in more specific tasks, such as medical image classification for object or pathology detection, the variability is reduced. For example, in the case of chest X-rays, all the images in the database present similar structures, with a dark background, the thorax in the center of the image, and very similar characteristics among patients. In this context,

synthetic perturbations tend to have a greater impact on the generated image, which ends up no longer belonging to the distribution of real images. For example, Charachon [2023] leverages domain translation techniques to produce an in-distribution stable and counterfactual image.

On tabular data, LORE [Guidotti et al., 2018a] uses a genetic algorithm to explore the decision boundary close to the data point of interest. In contrast, SHAP [Lundberg and Lee, 2017] approximates the evaluation of any combination of input features by simulating feature removals, allowing it to estimate Shapley values accurately [Shapley, 1953]. Hooker et al. [2019] suggested that marginalizing predictions over counterfactual inputs and using counterfactuals close to real inputs can help mitigate the OOD issue by maintaining a closer alignment with the training data distribution. Additionally, Jethani et al. [2021] proposed training models on counterfactual inputs to make them in-distribution, thereby improving the faithfulness of explanations to the task model. Furthermore, Delaunay et al. [2020] put forth an advanced approach to Anchor [Ribeiro et al., 2018] sampling, tailored specifically for tabular data. Their methodology generates samples that are more representative of the local neighborhood, which potentially reduces the generation of OOD data and enhances the accuracy of explanations. Jacovi and Goldberg [2021] concept of social alignment further supports the need for explanations that align with user expectations and the model's behavior, reinforcing the importance of these solutions in producing reliable and interpretable explanations.

### 1.5.2 Lack of consensus for evaluation

The evaluation of explanation and interpretability methods in machine learning lacks consensus, particularly regarding metrics and ground truth. Essentially, **there are no globally accepted metrics or clear, unambiguous definitions.** When it comes to quantitatively evaluating explanations for predictions made by machine learning models, four main desiderata are typically mentioned [Nguyen, 2018, Guidotti et al., 2019, DeYoung et al., 2020, Margot and Luta, 2021, Bhatt et al., 2021, Bodria et al., 2023]: *Simplicity*, *Faithfulness*, *Broadness*, and *Stability*. *Simplicity* (*i.e., low complexity*) measures how easily the explanation can be understood: **explanations should be simple and not involve too many features, making it easier for humans to understand.** *Faithfulness* (or *Fidelity*): measures how accurately an explainer mimics the predictions of the black-box model [Guidotti et al., 2019, DeYoung et al., 2020]. *Coverage* (or *Broadness*) refers to the general applicability of the explanation across different contexts. *Stability* (or *Sensitivity*): checks if similar inputs yield similar explanations [Alvarez Melis and Jaakkola, 2018]; the explanation should be stable and not change significantly with small perturbations in the input.

However, several ways have been proposed to assess the performance and reliability of explainers [Bodria et al., 2023]. *Deletion* and *insertion* metrics [Petsiuk et al., 2018] gauge the impact on model performance when important features, as identified by the explainer, are removed or added. The *deletion* metric observes how model accuracy degrades when important features are systematically removed, while the *insertion* metric examines how accuracy improves as important features are added back. *Monotonicity* [Luss et al., 2021] assesses whether the model's performance increases with the addition of more important features. *Running time* is also crucial metric, focusing on the computational efficiency of the explainer in generating explanations.

Notably, Jethani et al. [2021] propose *EVAL-X* and *REAL-X*, designed to enhance the evaluation and generation of explanations. EVAL-X evaluates explanations by estimating the true data-generating distribution for any input subset, using an evaluator model to avoid out-of-distribution issues common in other methods. REAL-X selects minimal feature subsets that maximize data li-

kelihood under the true data distribution estimate, ensuring selected features genuinely contribute to predictions without inflating performance metrics. Several benchmarks and tools have been developed to address the challenges in evaluating explanation methods in machine learning.

Some works employ *user studies* to evaluate explanations, recognizing that humans are the ultimate users of these explanations. Rong et al. [2023] analyze user studies in explainable AI conducted over recent years, categorizing them by focus on trust, understanding, usability, and human-AI collaboration performance. The review identifies a lack of integration of cognitive and social science insights and discusses best practices for designing and conducting user studies. The authors propose guidelines to better align XAI development with user needs. However, no metric has become a standard, and many definitions exist for the same. The problem is that users tend to prefer explanations that align with their idea of the task: they seek a good explanation for the problem itself, not for what the model does. This thesis focuses on explaining the predictions made by the model, while users are biased by their own ideas in this evaluation.

It is important to stress again that there is a lack of standardization in the quantitative evaluation of explanation methods. For the aforementioned desiderata, various metrics have been proposed, but they come with different mathematical definitions and/or experimental settings. This can lead to contradictory conclusions, as a metric, even if used with the same goal, measures different things concretely when defined differently [Hsia et al., 2024]. Additionally, the type of explainer used clearly impacts the results: evaluations for global and local explanations, for example, are significantly different. Furthermore, ad-hoc methods (specifically designed for a particular model) tend to be more faithful to the model compared to post-hoc and model-agnostic methods, but the advantages of the latter are difficult to quantify. Another issue is that the evaluation changes based on the type of data as well as the type of explanation. As we will see in Chapter 3, **it is not easy to compare methods based on feature importance with those based on rules** [Margot and Luta, 2021].

All of this presents additional challenges and issues, resulting in inconsistent and incomparable results across studies due to the absence of a unified approach or common ground truth. Without a widely accepted benchmark for explainable AI, the assessment of explanation methods remains complicated, hindering progress in the field.

### 1.5.3 Lack of mathematical foundation

Numerous methods have been proposed to explain machine learning predictions, each with its strengths and limitations. While these methods offer significant benefits, they often lack a robust theoretical foundation [Garreau and Luxburg, 2020], which can undermine their reliability and the trust placed in them. Often, their mechanisms are either overlooked or insufficiently studied, potentially making the explainer as opaque as the prediction it seeks to elucidate. **This risks creating a *black-box explainer* for understanding a *black-box model,* defeating the purpose of interpretability.** Consequently, using an explainer without a clear understanding of its mechanisms can lead to misinterpretations of the model's behavior, undermining trust and potentially resulting in biased decisions [Lipton, 2018].

Consider again an automated loan application system (Example 1.1.1). A bank operator might use an explainer to understand which features most influenced the loan decision. Different explainers might prioritize features differently based on their mechanisms, such as those closest to decision boundaries or those with extreme values. Without understanding how these explainers

work, the operator might misinterpret the model's behavior, leading to incorrect decisions, such as approving risky loans or rejecting safe ones.

SHAP [Lundberg and Lee, 2017], is frequently cited as a theoretically grounded approach to explaining machine learning predictions. SHAP leverages concepts from cooperative game theory, specifically Shapley values [Shapley, 1953], to provide a unified measure of feature importance. Shapley values provide a fair distribution of a total payout to players based on their contributions to the overall game. In the context of machine learning, the "game" is the prediction task, and the "players" are the input features. The Shapley value of a feature represents its average contribution to the prediction, considering all possible coalitions (subsets of features). Mathematically, the Shapley value $\phi_j$ for a feature $j$ is defined as

$$\phi_j(v) = \sum_{S \subseteq N \setminus \{j\}} \frac{|S|! \cdot (|N| - |S| - 1)!}{|N|!} \left[ v(S \cup \{j\}) - v(S) \right] , \tag{1.3}$$

where:

— $N$ is the set of all features,
— $S$ is a subset of $N$ not containing feature $j$,
— $v(S)$ is the value (or payoff) function that represents the contribution of the subset $S$ of features to the overall prediction.

This formula calculates the Shapley value by considering the marginal contribution of feature $i$ across all possible subsets $S$. The factorial terms $|S|!$ and $(|N| - |S| - 1)!$ represent the number of permutations of features where subset $S$ appears before and after feature $j$, respectively, ensuring that each selection of a subset of features is weighted appropriately.

Calculating exact Shapley values in practice can be computationally infeasible. The main problem is the need to retrain the model for each subset of features to accurately determine their contributions. Shapley values assume that the model works on a subset of the features and requires retraining. Even with methods to simulate the removal of subsets of features, the exponential growth in the number of possible coalitions makes the computation impractical for complex models with many features. Specifically, for a model with $b$ features, there are $2^b$ possible subsets of features, and evaluating the contribution of each feature across all these subsets is computationally prohibitive. Therefore, alternative methods and approximations, such as using sampling techniques or simplified models, are often employed to estimate Shapley values in practice.

To address this, the official SHAP repository * proposes several approximation methods to estimate Shapley values efficiently. These methods include

— *Kernel SHAP*, which uses a weighted linear regression model to approximate Shapley values by sampling coalitions and weighting them based on their likelihood.
— *Tree SHAP* [Lundberg et al., 2020] is an optimized algorithm designed for tree-based models, leveraging their structure to compute Shapley values in polynomial time.
— *Deep SHAP* combines DeepLIFT [Shrikumar et al., 2017] and Shapley values to approximate contributions in deep learning models.

Each of these approximation methods aims to balance the trade-off between computational efficiency and the accuracy of the Shapley value estimates. Despite the use of approximations, SHAP remains a popular choice for explaining machine learning models (see Figure 1.13). In addition,

*. https://github.com/shap/shap

SHAP is designed to handle different types of data, including tabular, image, and text data. However, some significant details in their implementation change accordingly, and these are not always explicitly documented. In general, there may be discrepancies between the methods described in the papers and the default settings in the official implementation. The reliance on approximation techniques introduces variability in the explanations provided by SHAP, depending on the specific implementation and parameters chosen. This variability can affect the interpretability and reliability of the explanations, particularly for complex models. Therefore, it is crucial to understand the limitations and assumptions underlying its approximation methods to use it effectively and interpret its results accurately.

However, the approximation is not the only issue: **some research challenges the entire concept of applying Game Theory techniques to interpretability.** Huang and Marques-Silva [2023a] demonstrates that Shapley values can misrepresent feature importance in certain classifiers, and associated predictions, thus incorrectly assign more importance to features that are provably irrelevant for the prediction, and less importance to features that are provably relevant for the prediction. Huang and Marques-Silva [2023b] supports this claim, providing theoretical and empirical evidence of their failure to identify true feature relevance, particularly in rule-based models. Marques-Silva and Huang [2023], Huang and Marques-Silva [2024] critique the use of Shapley values for explanations, also highlighting their potential to mislead by assigning high importance to irrelevant features and underestimating relevant ones. The authors underscore the computational complexity and approximation errors inherent in Shapley value calculations, as seen in SHAP, therefore advocating for alternative, logic-based explanation methods that more accurately reflect feature relevancy.

Anyway, numerous studies build on SHAP and Shapley values [Ibrahim et al., 2020, Frye et al., 2020, Tallón-Ballesteros and Chen, 2020, Lewis et al., 2021]. In particular, Bordt and von Luxburg [2023] extend the concept of Shapley values to demonstrate their ability to recover Generalized Additive Models (GAMs) with interactions. Bordt and von Luxburg [2023] introduce $n$-Shapley Values, a parametric family of local post-hoc explanation algorithms that incorporate interaction terms up to order $n$. This extension allows for a sequence of explanations that range from traditional Shapley Values to a complete decomposition of the function. They show that $n$-Shapley Values can recover GAMs with interaction terms up to order $n$, thereby linking Shapley Values to GAMs and providing a precise characterization of Shapley Values used in explainable machine learning.

Recall that LIME works by perturbing the instance to be explained, querying the model on these generated samples, and training a linear model on the perturbed data. The coefficients of the linear model then represent the importance of each feature. However, the process is not straightforward in practice and requires careful adaptation depending on the type of data being analyzed. Garreau and Luxburg [2020] provide a theoretical analysis of LIME for tabular data (with default setting), showing that its coefficients are approximately proportional to the gradient of the function being explained, thus confirming its ability to identify significant features. However, **an inappropriate parameter selection, such as the number of samples or the definition of the neighborhood, can cause LIME to miss important features**, reducing the reliability of its explanations.

Extending this analysis, Mardaoui and Garreau [2021] and Garreau and Mardaoui [2021] examine LIME in the context of text data and images, respectively. They demonstrate that LIME explanations converge to a limit explanation when a large number of examples are generated.

Additionally, their work uncovers connections between LIME and Integrated Gradients [Sundararajan et al., 2017a]. This connection highlights that both methods share a common theoretical foundation in gradient-based explanations, further validating LIME's approach while emphasizing the need for careful parameter tuning to ensure accurate and meaningful explanations.

With the aim of providing theoretical results to the field, Garreau and Luxburg [2020] propose focusing on how explainers behave when applied to simple, inherently interpretable models such as basic linear models or small decision trees. In these cases, the decision-making process is clear and unambiguous, with a precise understanding of which features impact the outcome. An ideal, trustworthy explainer should accurately capture and highlight these relationships, providing clear and understandable explanations. Conversely, if an explainer fails to elucidate the behavior of simple models, it raises serious doubts about its ability to handle more complex architectures. After all, **if an explainer falls short in simple scenarios, how can we trust it to accurately explain the behavior of more complex models?**

One important reason to assess the reliability of these methods is their necessity for legal and regulatory purposes. Bordt et al. [2022] claim that **post-hoc explanation algorithms (such as SHAP and LIME) are inadequate for legal and regulatory transparency objectives** (as those described in Section 1.3.2) in adversarial contexts, where the explanation provider and receiver have conflicting interests. The authors combine legal, philosophical, and technical arguments to demonstrate that post-hoc explanations can be manipulated, lack clear standards, and do not reliably convey the true reasons behind decisions. Similarly, Pawelczyk et al. [2022b] investigates the conflict between two key principles in data protection regulations: the right to be forgotten and the right to (actionable) explanations. They theoretically and empirically demonstrate that even **a small number of data deletion requests can significantly invalidate the recourses generated by state-of-the-art algorithms.** In addition, Agarwal et al. [2022a] highlight that existing methods, such as SHAP and LIME, are unstable, meaning that small perturbations in input can lead to significantly different explanations.

Fokkema et al. [2023] explores the inherent conflict between robustness and recourse in feature attribution methods for machine learning models and prove that **no single attribution method can be both recourse sensitive and robust for all models.** This impossibility applies to the aforementioned methods like LIME, SHAP, Integrated Gradients. However, they suggest potential workarounds to mitigate the conflict between robustness and recourse. Bilodeau et al. [2024] examine the limitations of feature attribution methods, showing that these can provably fail to improve on random guessing for simple tasks. More broadly, Bordt and von Luxburg [2024] argue that many explanation algorithms are mathematically complex but lack clear interpretation, leading to potential misinterpretations. They propose a distinction between the mathematical properties of explanation algorithms and their interpretation, also suggesting that complex algorithms without clear interpretation are essentially black-boxes. The paper emphasizes the need for explanation algorithms to explicitly state the interpretable questions they are designed to answer and to be studied rigorously through empirical validation.

Several works explore the mathematical foundations of counterfactual explanations [Pawelczyk, 2024], which are hypothetical scenarios that illustrate how altering certain input features can change the outcome of an algorithm's decision (described in Section 1.19). These explanations are crucial for providing algorithmic recourse, allowing individuals to understand and take actionable steps to reverse unfavorable decisions, such as a loan denial (Example 1.1.1). Pawelczyk et al. [2023], Fokkema et al. [2024], Leemann et al. [2024b] address the challenges related to ensuring

the reliability and robustness of these recourse mechanisms, making sure that the suggested actions are realistic, feasible, and effective even in the presence of data changes or noise.

Other explore the foundations of concept-based explanations [Poeta et al., 2023], focusing on the identifiability and interpretability of learned embedding spaces. Leemann et al. [2023] introduce a framework for concept discovery in embedding spaces, emphasizing the need for identifiable methods that can reliably recover known concepts. Feng et al. [2024] proposes an axiomatic and model-agnostic approach for producing robust concept-based explanations.

Recently, with the rise of transformer models [Vaswani et al., 2017], the attention mechanism [Bahdanau et al., 2015] has been widely utilized to provide explanations for model predictions, with the intuition to provide insight on the part of the input that the model is focusing on [Chefer et al., 2021, Mylonas et al., 2023]. However, the use of attention weights as explanations is the subject of a broad and complex debate [Jain and Wallace, 2019, Wiegreffe and Pinter, 2019, Serrano and Smith, 2019, Cui et al., 2024]. The efficacy and reliability of attention-based explanations have been questioned [Leemann et al., 2024a]. In Chapter 6, we will delve deeply into this debate, mathematically analyzing attention-based explanations.

## 1.6 Contributions

This thesis focuses on post-hoc interpretability methods for machine learning models. It does not delve into global interpretability or the design of specific architectures for training inherently interpretable models. Instead, the primary focus is on model-agnostic approaches, which completely disregard the internal workings of the underlying model, treating it as a black box.

Thus, this work focuses on solutions based solely on the ability to query the model as needed. Among model-dependent methods, attention-based approaches are explored in Chapter 6 and compared with other techniques. While these methods are still local and post-hoc, explaining individual predictions made by an already trained model, they require access to certain internal parameters, specifically attention weights.

The manuscript begins with an empirical comparison of popular existing methods, particularly LIME and Anchors, in Chapter 3. Lopardo and Garreau [2022] examined whether these methods agree both qualitatively and quantitatively in their explanations. Chapter 4 describes the first theoretical analysis conducted on Anchors. Lopardo et al. [2023a] provide rigorous definitions and employ a robust mathematical framework to understand its behavior. This theoretical understanding allowed for the identification of the strengths and weaknesses of existing methods. With these insights at hand, Lopardo et al. [2023b] proposed a novel solution for text model interpretability. This solution, described in Chapter 5, leverages the strengths of existing methods while addressing their limitations, offering a more robust and reliable approach. This manuscript also contributes to the ongoing debate on the use of attention weights for transformer interpretability. Chapter 6 mathematically and empirically compare attention-based explanations with those obtained from other post-hoc methods [Lopardo et al., 2024]. All theoretical claims are supported by mathematical proofs, discussed in detail in their respective chapters. Furthermore, the claims are validated with numerical experiments in Python. To ensure reproducibility, the code is publicly available on GitHub, supporting the reliability and robustness of the results.

## Publications List

**International conferences:**

— G. Lopardo, D. Garreau, F. Precioso, G. Ottosson (2022). SMACE: A New Method for the Interpretability of Composite Decision Systems. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (**ECML 2022**).

— G. Lopardo, F. Precioso, D. Garreau (2023). A Sea of Words: An In-Depth Analysis of Anchors for Text Data. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics* (**AISTATS 2023**).

— G. Lopardo, F. Precioso, D. Garreau (2024). Attention Meets Post-hoc Interpretability: A Mathematical Perspective. In *Proceedings of the 41st International Conference on Machine Learning* (**ICML 2024**).

**Workshops and national conferences:**

— G. Lopardo, D. Garreau (2022). Comparing Feature Importance and Rule Extraction for Interpretability on Text Data. In *Proceedings of the 2nd Workshop on Explainable and Ethical AI at the 26th International Conference on Pattern Recognition* (**XAIE ICPR 2022**).

— G. Lopardo, F. Precioso, D. Garreau (2023). Understanding Post-hoc Explainers: The Case of Anchors. In *54th Journées de Statistique* **(JDS 2023)**.

**Working paper:**

— G. Lopardo, F. Precioso, D. Garreau (2024). Faithful and Robust Local Interpretability for Textual Predictions. *arXiv:2311.01605 preprint*, 2024.

# CHAPTER 2

# Setting and notation

This manuscript focuses on the application of interpretable machine learning to text-based models. This chapter begins by introducing the general notation used throughout the manuscript in Section 2.1. Following that, Section 2.2 provides a comprehensive overview of text vectorization methods employed in Natural Language Processing (NLP). Since computers can only process numbers, these methods transform documents into numerical vectors. A particular focus is on TF-IDF vectorization, which will be frequently referenced in subsequent chapters. Section 2.3 examines the interpretability of NLP models, covering specific approaches from Section 1.1 that are applicable to text data. Finally, Section 2.4 discusses the metrics and approaches used to evaluate explanations in the specific context of text models, emphasizing the challenges and lack of agreement in quantitatively assessing explanations, as mentioned in Section 1.5.2.

## 2.1 Notation

Throughout this manuscript, $z$ represents any generic document from a corpus $\mathcal{T} \subseteq \mathcal{X}$ (*i.e.*, a collection of texts), while $\xi$ denotes the specific example under exam. Refer to $\mathcal{D}$ as the *global dictionary*: the complete collection of $D = |\mathcal{D}|$ unique words used across $\mathcal{T}$. In practice, it represents the set of distinct words contained in the documents of the corpus $\mathcal{T}$. A document $z$ is essentially a sequence (*i.e.*, an ordered list) $z = (z_1, \dots, z_b)$ of words from this dictionary. For any integer $k$, define $[k] := \{1, \dots, k\}$.

**Definition 2.1.1 (Multiplicity of a word).** Let $m_j(z)$ denote the *multiplicity* of the word $w_j \in \mathcal{D}$ in document $z$, defined as $m_j(z) := |\{k \in [b] \mid z_k = w_j\}|$. When the context is clear, $m_j$ is used as shorthand for $m_j(z)$.

**Definition 2.1.2 (Local dictionary).** Let $z$ be a document composed of words from a global dictionary $\mathcal{D}$. The *local dictionary* $\mathcal{D}_z$ is the subset of $\mathcal{D}$ containing all distinct words present in $z$. Formally,

$$\mathcal{D}_z = \{w_j \in \mathcal{D} \mid m_j(z) > 0\}\,.$$

For instance, consider the restaurant review $\xi$:

$$\xi = \text{“the food is great and the location is nice”}\,.$$

It contains $d = 7$ distinct words from the (global) dictionary $\mathcal{D}$. The local dictionary $\mathcal{D}_\xi$ is:

$$\mathcal{D}_\xi = \{\textit{the}, \textit{food}, \textit{is}, \textit{great}, \textit{and}, \textit{location}, \textit{nice}\} = \{w_1, w_2, w_3, w_4, w_5, w_6, w_7\}\,.$$

It is crucial to differentiate between two key concepts: *tokens* and *words*. A word in a document $z$ is any unique and distinct element within the local dictionary $\mathcal{D}_z$. In this manuscript, **a token is defined as a specific instance of a word appearing at a particular position within a text document.** A document $z$ is as an ordered sequence of tokens $z_1, \dots, z_b$.

In practice, however, the representation of a token can vary depending on the tokenization method used. For example, Byte Pair Encoding [Gage, 1994, Sennrich et al., 2016], which is employed by models such as GPT-2, uses sub-words. This method breaks down words into smaller, more frequent sub-word units to handle rare words and improve model efficiency. This approach allows for a more flexible representation of text, accommodating different levels of granularity.

The example $\xi$ above is therefore defined as

$$\xi = (\textit{the}, \textit{food}, \textit{is}, \textit{great}, \textit{and}, \textit{the}, \textit{location}, \textit{is}, \textit{nice}) = (\xi_1, \xi_2, \xi_3, \xi_4, \xi_5, \xi_6, \xi_7, \xi_8, \xi_9)\,,$$

with $b = |\xi| = 9$ tokens and $d = |\mathcal{D}_\xi| = 7$ distinct words. Note that $d = |\mathcal{D}_\xi| \leq |\xi| = b$: the words “the” and “is” appear twice in $\xi$, but each occurrence is a separate token. In particular, the multiplicities $m_1$ and $m_3$ are equal to 2, while $m_j = 1$ for all $j \in [d]$ with $j \neq 1, 3$. Finally, note that $b = |\xi| = \sum_{j=1}^{d} m_j$.

The goal is to explain the predictions made by a generic prediction model, denoted by $F$, which takes as input textual documents from a corpus $\mathcal{T}$. Unless explicitly stated otherwise, this model is considered as a *black-box* function. Formally, consider a model $F : \mathcal{X} \to \mathcal{Y} \in \mathbb{R}^p$.

This framework encompasses both regression and classification settings. In regression tasks, $\mathcal{Y}$ represents a continuous value. Consider a model predicting housing prices based on textual descriptions, $\mathcal{Y}$ would correspond to a range of possible prices. In *multi-class classification* problems,

$\mathcal{Y}$ represents a set of *confidence scores* for $p$ different classes in $\mathcal{C} = \{0, 1, \ldots, p-1\}$. Thus, $F(z) = (F_0(z), \ldots, F_{p-1}(z))^\top \in [0,1]^p$, such that

$$\sum_{\ell=0}^{p-1} F_\ell(z) = 1\,. \tag{2.1}$$

In practice, for classification tasks, the model $F$ is a function designed to assign a confidence score to each class in $\mathcal{C}$ for a given input. **This confidence score indicates the likelihood that the input belongs to a particular class.** The predicted class is therefore the one corresponding to the highest confidence score.

Consider a model $F$ that makes predictions for a given input $\xi$. The *predicted class* $\ell^\star \in \mathcal{C}$ is the one with the highest confidence score:

$$\ell^\star := \arg\max_{\ell \in \mathcal{C}} F_\ell(\xi)\,, \tag{2.2}$$

where $F_\ell(\xi)$ denotes the confidence score assigned by the model $F$ to the class $\ell$ for the input $\xi$.

While the model $F$ represents the generic black-box model, in the following chapters $f$ refers to the specific out *prediction* to be explained. Unless otherwise specified, $f$ will be used to explain **why the model made a specific prediction** (as in Chapters 3 and 4). Therefore, in the case of classification, $f(z) = F_{\ell^\star}(z)$ for any $z \in \mathcal{X}$. In other cases (Chapter 5), where it is also necessary to understand **why a different class was not predicted**, $f(z) = F_\ell(z)$ will be evaluated to explain the missed prediction of class $\ell \in \mathcal{C}$. However, in general, note that $f : \mathcal{T} \subseteq \mathcal{X} \to \mathbb{R}$.

*Example 2.1.1 –* Consider a (binary) sentiment analysis model (*i.e.*, $\mathcal{C} = \{0, 1\}$) classifying reviews as *negative* (indexed as $\ell = 0$) or *positive* (indexed as $\ell = 1$). The output space $\mathcal{Y} = [0,1]^2$ is a $p = 2$-dimensional vector, with each element reflecting the confidence score for a specific sentiment class. Consider again the example review $\xi =$ "the food is great and the place is nice". The example $\xi$ is classified as *positive*, since $F_1(\xi) > F_0(\xi)$. The classifier $f$ will therefore be the confidence score for the positive class, *i.e.*, $f(z) = F_1(z)$, for any document $z \in \mathcal{X}$.

Without loss of generality, **binary classification will often be considered in this manuscript.** This is because any multiclass classification problem can be reduced to multiple binary classification problems by using a *one-vs-all* approach, where classes are considered mutually exclusive as per Eq. (2.1). For example, in a sentiment analysis model as described in Example 2.1.1, the classifier would output a binary decision based on whether the confidence score for the positive class exceeds that for the negative class. By using this one-vs-all approach, the generality of the analysis in Eq. (2.1) is maintained, while simplifying the discussion to the binary case for clarity and ease of understanding.

In essence, the primary focus of this thesis is to study and develop methods for providing clear and comprehensive explanations of the prediction of interests $f(\xi) \in F(\xi)$ made by any black-box model $F$. This model could be any machine learning model used for text prediction, without any specific assumptions about its nature or architecture unless explicitly stated. The prediction $F(\xi)$ is made on a specific text instance, denoted as the example $\xi = (\xi_1, \ldots, \xi_b)$. The goal is to understand which tokens in $\xi$ contribute to the prediction, and how they do so. This involves determining the most influential tokens driving the prediction and analyzing how changes in these tokens might affect the outcome.

## 2.2 Text vectorizers

Computers, and by extension machine learning models, can only process numerical data. Therefore, **text data is usually encoded into a numerical format for these models to function effectively**. Natural language processing classifiers are typically based on a vector representation, denoted as $\varphi$, of the document [Young et al., 2018]. This transformation process is achieved using *text vectorizers*, tools that transform text into numerical vectors: structured arrays of numbers that models can manipulate. This numerical representation is essential for enabling the application of machine learning algorithms to natural language processing tasks. In practice, the model $F$ can be represented as $F = h \circ \varphi$, where $\varphi$ is a deterministic mapping $\mathcal{X} \to \mathbb{R}^D$ and $h : \mathbb{R}^D \to \mathcal{Y}$ is a function that applies machine learning algorithms to the vectorized representation.

**One basic idea is to simply count the words in a text**, which is the basis of the *Bag of Words* (BoW) model. In BoW, a text document is represented by the multiplicity of each word within it. Each document is converted into a vector where each element corresponds to a specific word from the entire vocabulary, and the value of each element is its multiplicity, *i.e.*, count of how often that word appears in the document. To illustrate this mechanism, consider three simple restaurant reviews:

$$\begin{aligned} z^{(1)} &= \text{“the food is great and the place is nice”}\,, \\ z^{(2)} &= \text{“the place is simply great”}\,, \\ z^{(3)} &= \text{“the price is low”}\,. \end{aligned}$$

This means that the corpus $\mathcal{X} = \{z^{(1)}, z^{(2)}, z^{(3)}\}$. First, construct the dictionary $\mathcal{D}$ from all unique words in $\mathcal{X}$:

$$\begin{aligned} \mathcal{D} &= \{\textit{the, food, is, great, and, place, nice, simply, price, low}\} \\ &= \{w_1, w_2, w_3, w_4, w_5, w_6, w_7, w_8, w_9, w_{10}\}\,. \end{aligned}$$

Next, the BoW representation $\varphi$ is created, by mapping any element of the global dictionary $\mathcal{D}$ to its multiplicity in any document $z^{(i)}$, for $i = 1, 2, 3$, *i.e.*, $\varphi(z^{(i)}) = (m_1(z^{(i)}), \ldots, m_D(z^{(i)}))$:

| | *the* | *food* | *is* | *great* | *and* | *place* | *nice* | *simply* | *price* | *low* |
|---|---|---|---|---|---|---|---|---|---|---|
| | $w_1$ | $w_2$ | $w_3$ | $w_4$ | $w_5$ | $w_6$ | $w_7$ | $w_8$ | $w_9$ | $w_{10}$ |
| $z^{(1)}$ | 2 | 1 | 2 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| $z^{(2)}$ | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| $z^{(3)}$ | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

Here, each row represents a document, and each column represents a word from the global dictionary $\mathcal{D}$. The values indicate the multiplicity of each word in the respective document. For example, “the” appears twice in $z^{(1)}$, hence $m_1(z^{(1)}) = 2$, and once in $z^{(2)}$ and $z^{(3)}$, so that $m_1(z^{(2)}) = 1$ and $m_1(z^{(3)}) = 1$; “food” appears once in $z^{(1)}$ (*i.e.*, $m_2(z^{(1)}) = 1$), but never in $z^{(2)}$ and $z^{(3)}$, thus $m_2(z^{(2)}) = m_2(z^{(3)}) = 0$.

Note that, while BoW is straightforward and effective for capturing word occurrences, it **ignores the order and context of words, focusing only on their presence and frequency**. In particular, in the example above, BoW gives undue weight to common, less meaningful words like “the” and “is.”

The *Term Frequency-Inverse Document Frequency* (TF-IDF) technique [Luhn, 1957, Jones, 1972] refines the BoW approach by considering not just the frequency of words in a document, but also the importance of these words across the entire corpus $\mathcal{T}$. Term frequency (TF) is the count of a word in a document, *i.e.*, its multiplicity, while inverse document frequency (IDF) measures how common or rare a word is across all documents. The TF-IDF score is the product of these two metrics. **The principle is to assign more weight to words that appear frequently in a document, and not so frequently in the corpus.**

The (non-normalized) TF-IDF vectorization is defined as follows:

**Definition 2.2.1 (TF-IDF vectorization).** Let $N$ be the size of the initial corpus $\mathcal{T}$, *i.e.*, the number of documents in the dataset. Let $N_j$ be the number of documents containing the word $w_j \in \mathcal{D}$. The TF-IDF of $z$ is the vector $\varphi(z) \in \mathbb{R}^D$ defined as

$$\forall j \in [D], \qquad \varphi(z)_j := m_j(z) v_j \,,$$

where $v_j := \log\left(\frac{N}{N_j}\right) + 1$ is the *inverse document frequency* * (IDF) of $w_j$ in $\mathcal{T}$.

To illustrate the TF-IDF vectorization, consider the previous example sentences $z^{(1)}$, $z^{(2)}$, and $z^{(3)}$. First, calculate the term frequency (TF) for each word in each document: and this correspond exactly to the BoW representation shown above:

| | *the* | *food* | *is* | *great* | *and* | *place* | *nice* | *simply* | *price* | *low* |
|---|---|---|---|---|---|---|---|---|---|---|
| | $w_1$ | $w_2$ | $w_3$ | $w_4$ | $w_5$ | $w_6$ | $w_7$ | $w_8$ | $w_9$ | $w_{10}$ |
| $z^{(1)}$ | 2 | 1 | 2 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| $z^{(2)}$ | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| $z^{(3)}$ | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

Next, calculate the document frequency $N_j$ for each word $w_j$.

| | *the* | *food* | *is* | *great* | *and* | *place* | *nice* | *simply* | *price* | *low* |
|---|---|---|---|---|---|---|---|---|---|---|
| | $w_1$ | $w_2$ | $w_3$ | $w_4$ | $w_5$ | $w_6$ | $w_7$ | $w_8$ | $w_9$ | $w_{10}$ |
| DF | 3 | 1 | 3 | 2 | 1 | 2 | 1 | 1 | 1 | 1 |

Then, compute the inverse document frequency (IDF) as $v_j := \log\left(\frac{N}{N_j}\right) + 1$.

| | *the* | *food* | *is* | *great* | *and* | *place* | *nice* | *simply* | *price* | *low* |
|---|---|---|---|---|---|---|---|---|---|---|
| | $w_1$ | $w_2$ | $w_3$ | $w_4$ | $w_5$ | $w_6$ | $w_7$ | $w_8$ | $w_9$ | $w_{10}$ |
| IDF | 1.0 | 2.1 | 1.0 | 1.4 | 2.1 | 1.4 | 2.1 | 2.1 | 2.1 | 2.1 |

Finally, compute the TF-IDF score by multiplying the TF by the IDF:

| | *the* | *food* | *is* | *great* | *and* | *place* | *nice* | *simply* | *price* | *low* |
|---|---|---|---|---|---|---|---|---|---|---|
| | $w_1$ | $w_2$ | $w_3$ | $w_4$ | $w_5$ | $w_6$ | $w_7$ | $w_8$ | $w_9$ | $w_{10}$ |
| $z^{(1)}$ | 2.0 | 2.1 | 2.0 | 1.4 | 2.1 | 1.4 | 2.1 | 0 | 0 | 0 |
| $z^{(2)}$ | 1.0 | 0 | 1.0 | 1.4 | 0 | 1.4 | 0 | 2.1 | 0 | 0 |
| $z^{(3)}$ | 1.0 | 0 | 1.0 | 0 | 0 | 0 | 0 | 0 | 2.1 | 2.1 |

∗. As defined in the `scikit-learn` library: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html.

Here, each row represents a document, and each column represents a word from the global dictionary $\mathcal{D}$. The values indicate the TF-IDF score of each word in the respective document. TF-IDF normalizes the importance of terms across the corpus, which helps in highlighting less frequent but potentially more significant words. This allows words like "food" and "price" to stand out more in the analysis, whereas they may be overshadowed by common words in a BoW model.

Note that, in Definition 2.2.1, **once the TF-IDF vectorizer is fitted on a corpus $\mathcal{T}$, the vocabulary is fixed once and forever afterward.** Meaning that, if a word is not part of the initial corpus $\mathcal{T}$, its TF-IDF term is zero. As will be detailed in the following Chapters, explainers like LIME, SHAP, and Anchors perturb documents by replacing tokens with a *mask*: a word, such as `UNK`, that realistically is not present in the initial corpus. Thus, **replacing any token with this word is equivalent to simply removing it, from the point of view of TF-IDF**. However, depending to the underlying model, as discussed in Section 1.5.1, this can cause out-of-distribution samples.

While BoW and TF-IDF are effective methods for text representation, they are limited by their inability to capture the semantic meaning of words beyond mere frequency. BoW and TF-IDF primarily treat text as a collection of individual words, ignoring the order and context in which these words appear. To address some of these limitations, several approaches have been proposed over the years. *$n$-grams* [Shannon, 1951, Broder et al., 1997] consider sequences of $n$ words together, capturing some context but often leading to high-dimensional and sparse representations. *Part-of-speech* (POS) tagging [Petrov et al., 2012] identifies the grammatical role of each word in a sentence (*e.g.*, noun, verb, adjective), providing syntactic context that can enhance text representation. However, these methods still struggle with representing deeper semantic relationships and can be less efficient.

Modern *word embeddings* address this limitation by providing dense vector representations of words in a continuous vector space, where semantically similar words are mapped to nearby points. *Word2Vec* [Mikolov et al., 2013] uses neural networks to learn word representations. The objective is to maximize the probability of the observed word given the context, resulting in vectors where words with similar contexts are closer together. *GloVe* [Pennington et al., 2014] is based on matrix factorization techniques: it constructs a co-occurrence matrix, where each entry represents how frequently a word pair appears together within a specified context window. By factorizing this matrix, GloVe generates word vectors that capture both local and global statistical information of words in the corpus. *Doc2Vec* [Le and Mikolov, 2014] extends the Word2Vec approach to generate vectors for entire documents, allowing for capturing the semantic content and context of entire documents, rather than just isolated words. This makes it particularly useful for tasks like document classification, clustering, and recommendation systems.

In modern natural language processing, more advanced models often utilize approaches where embeddings are learned directly from data. For instance, models like BERT and GPT-3 leverage deep learning techniques to generate context-sensitive embeddings [Gage, 1994, Sennrich et al., 2016]. These embeddings are obtained by training on large corpora using transformers, which consider the entire context of a word or sentence, resulting in more nuanced and accurate representations. This allows for a deeper understanding of language, capturing subtleties and variations in meaning that simpler methods might miss.

## 2.3 Post-hoc explanations in NLP

In this section, the focus is on the interpretability of Natural Language Processing (NLP) models, with an emphasis on post-hoc methods. These methods aim to elucidate the decision-making processes of black-box models, as described in Section 1.1. Addressing the intrinsic complexity and unstructured nature of textual data requires specialized approaches to generate interpretable explanations.

This subfield is demonstrably underexplored [Danilevsky et al., 2020]. For instance, Bodria et al. [2023] states that "explanations for classifiers acting on text data are at the very early stages compared to tabular data and images." Current interpretability techniques predominantly address structured data or computer vision tasks, with less methods specifically designed for textual data. This gap is concerning given the rapid rise of transformers and large language models (Section 1.2), which are becoming integral to increasingly pervasive applications (Section 1.3.1).

This thesis explores various post-hoc interpretability frameworks. Each method is discussed in the context of its application to NLP models, highlighting its strengths and limitations. Additionally, challenges in interpreting complex models such as transformers are addressed, with suggestions for future research to bridge the gap between model performance and interpretability. In summary, the aim is to illuminate the current state of interpretability in NLP, emphasize the need for tailored methods for textual data, and advocate for increased research efforts in this crucial area.

In practical terms, to obtain meaningful explanations from textual predictions, three fundamental strategies emerge: (1) **identifying keywords**, which involves pinpointing a representative subset of tokens that are influential and crucial in the prediction; (2) assigning importance weights to tokens, also known as **sentence highlighting**, which entails attributing weights to each token based on their contribution to the prediction, thereby highlighting the most significant tokens; and (3) producing **counterfactual explanations**: this involves generating hypothetical documents by altering certain tokens in the original text to obtain different predictions. Generating counterfactual and prototype explanations for textual data presents significant challenges due to the semantic richness and sensitivity of text, where small changes in wording can drastically alter the meaning of a sentence.

These three form of explanations, obtained with FRED (the explainer described in Chapter 5) are exemplified by Figure 2.1.

Clearly, many difficulties arise when providing explanations, especially using perturbation-based approaches. Perturbation-based methods involve making small changes to the text to observe how these changes affect the model's predictions. However, even minor alterations can disrupt the coherence and grammatical structure of the text, leading to nonsensical phrases. This disruption often results in out-of-distribution samples (Section 1.5.1), where the modified text no longer resembles the kind of data the model was trained on. Consequently, the model's predictions on these perturbed texts may be unreliable or invalid, complicating the task of generating meaningful and accurate explanations. This issue underscores the need for more sophisticated methods that can perturb text while maintaining its semantic integrity.

### 2.3.1 Identifying keywords

The first strategy involves **rule-based explanations for text by identifying a representative subset of tokens**, often referred to as *keywords*. This process entails isolating a compact yet critical

**(a) Identifying keywords**

Explaining class "positive":

The minimal subset of tokens that make the confidence drop by 50.0% if perturbed is {"decent", "great"}

**(b) Sentence highlighting**


**(c) Counterfactual explanations**

$k = 3$ samples with minimal perturbation classified as *negative*:

"poor drinks decent food dirty service"
"poor drinks decent food awful service"
"poor drinks bad food great service"

Figure 2.1 – **Explaining the prediction of a sentiment analysis model** for the restaurant review "poor drinks, decent food, great service," classified as *positive*. **(a)** The identified keywords. **(b)** Saliency weights of token importance score: dark green (resp., red) means high positive (resp., negative) influence. **(c)** Counterfactual explanations: instances close to the example, but classified as *negative*. Perturbations with respect to the example are in orange.

set of tokens that encapsulate the core meaning of the text and play a decisive role in influencing the model's prediction. These keywords exert the most substantial impact on the model's decision-making process, making them crucial for understanding how the model interprets and prioritizes different parts of the input.

Consider Figure 2.1. The goal here is to identify a small group of words that, on their own, strongly influence the model's decision, meaning they lead to a specific classification. For positive reviews, keywords might include *excellent, amazing,* or *great,* while negative reviews might be flagged by words like *disappointing, terrible,* or *poor.* Mathematically, given the example $\xi = (\xi_1, \xi_2, \ldots, \xi_b)$ and the prediction $f(\xi)$, this class of methods identify a (possibly small) subset of tokens (*e.g.,* $\{\xi_3, \xi_7, \xi_{12}\}$) that mainly trigger the predictor $f$.

This approach is related to rule-based explanations mentioned in Section 1.1, where specific rules or patterns are identified to provide clear and interpretable insights into the model's behavior. As deeply discussed in Chapter 4, **Anchors falls into this category: it explains a decision by identifying the smallest set of words (an *anchor*) such that the model to explain has similar outputs when they are present in a document.** In this context, Anchors essentially tells us that, for instance, if the tokens $\xi_3, \xi_7, \xi_{12}$ are present in the example $\xi$, then the prediction will be *positive*, with a certain confidence. This ensures that the identified tokens are pivotal in driving the prediction.

### 2.3.2 Sentence highlighting

**Importance-based methods assign numerical weights to each token in the text, reflecting its relative importance in determining the model's prediction.** In the context of text data, this enables sentence highlighting to visually emphasize the most influential tokens.

Consider the Figure 2.1 again. In a sentiment analysis task, assigning weights helps us understand how each token contributes to the overall sentiment of the text. Words like "amazing,"

"terrible," or "extraordinary," which convey strong emotions, might receive higher weights compared to common words like "the," "a," or "and," which hold less significance in shaping sentiment. Formally, these methods assign a score $s$ to each input feature, *i.e.*, $s(\xi_1), s(\xi_2), \ldots, s(\xi_b)$, that allows us to highlight the important part of the text (see Figure 2.1), in a similar vein as the saliency maps used for images in Figure 1.3.

**Pertubation-based techniques such as LIME and SHAP are key methods in this category.** LIME creates neighborhoods of sentences by perturbing the input through randomly masking words, which helps in identifying the words that most significantly influence the model's prediction. For a detailed analysis of LIME for text data, refer to Mardaoui and Garreau [2021]. Additionally, Lopardo and Garreau [2022] compare LIME and Anchors explanations, as exposed in Chapter 3. SHAP assigns importance scores to each word using Shapley values from cooperative game theory, providing a unified and theoretically grounded measure of feature importance.

As shown in Chapter 6, **gradient-based methods and attention-based methods can also be used for sentence highlighting.** These methods assign numerical weights to each token in the text to reflect its relative importance in determining the model's prediction. Gradient-based methods calculate these importance scores by leveraging the gradients of the model's output with respect to its input features (see Section 1.4.5). The *Gradient* method uses the raw gradients of the output with respect to the input tokens. The gradient value indicates how much a small change in the input token would affect the output prediction. However, raw gradients can be noisy and may not always provide clear insights. *Gradient×Input* improves upon this by multiplying the gradient by the token embeddings, providing a measure of feature importance that accounts for both the sensitivity (gradient) and the actual value of the token. *Integrated Gradients* addresses the shortcomings of raw gradients by computing the integral of gradients along the path from a baseline input to the token embedding. Similarly, *DeepLIFT* decomposes the output prediction by comparing the activations of the input with those of a baseline. It assigns contribution scores to each input token based on the difference in activations, providing a clear explanation of how each token influences the prediction.

Alvarez-Melis and Jaakkola [2017] presents a method for explaining sequence-to-sequence models by identifying causally related input-output token pairs. Poerner et al. [2018] introduces LIMSSE, an explanation method inspired by LIME specifically designed for NLP, ideal for word-order sensitive tasks (*e.g.*, RNNs, CNNs). LIMSSE evaluates feature importance by creating perturbed versions of input texts and assessing their impact on model predictions. It combines insights from LIME with adaptations for textual data, ensuring more accurate and contextually relevant explanations.

### 2.3.3 Counterfactual explanations

**Counterfactual explanations offer a powerful approach to understanding model predictions by exploring hypothetical scenarios.** These explanations involve modifying the input text to observe how changes affect the model's prediction, providing valuable insights into the model's reasoning process and its sensitivity to specific textual elements. As discussed in Section 1.4.9, counterfactual explanations are particularly useful because **they align with users' thought processes, making them more intuitive and acceptable.**

In the context of NLP models, **a counterfactual explanation typically consists of a text that is similar to the original but with certain tokens removed or altered to change the prediction.** For example, in Figure 2.1, the counterfactual explanations identify specific changes to the text

that could switch the prediction to *negative*. This involve suggesting alternative words, such as replacing "great" with "awful" or "decent" with "bad", thereby demonstrating how these modifications could alter the classification. Formally, counterfactual explanations indicate that replacing specific tokens, such as $\xi_2$ and $\xi_5$, with different words from the dictionary $\mathcal{D}$ would *provably* change the prediction.

Recent works have underscored the potential of counterfactual explanations in providing realistic, aligned, and user-accepted insights. Martens and Provost [2014] introduced *SEDC* (Sentence-Level Explanation for Document Classification), a method tailored for text classifiers that generates counterfactuals by identifying minimal textual changes to alter classification outcomes. Wachter et al. [2017b] expanded the concept by proposing a general framework for counterfactual explanations applicable to various models, especially non-linear classifiers like neural networks. *XSpells* [Lampridis et al., 2020] applies linguistic transformations to create realistic and grammatically correct exemplars and counterfactuals. Pawelczyk et al. [2021] introduced *CARLA*, a benchmarking framework for counterfactual explanations that allows for systematic comparison and evaluation of different methods. Building on this, Pawelczyk et al. [2022a] explored the robustness and reliability of counterfactual explanations, addressing challenges such as feasibility and plausibility in their generation. *CAT* [Chemmengath et al., 2022] uses a BERT [Devlin et al., 2019], a language model, to perturb words while maintaining sentence structure.

Prototype explanations enhance model understanding by showing the similarity between input text and representative examples, offering a contextually rich complement to counterfactual methods. For instance, Wallace et al. [2018] presents Deep-$k$NN, which improves neural network interpretability by using labels of the nearest neighbors during testing. This method explains predictions by finding and using the most similar training examples. Other methods aim to identify similar documents within a dataset to boost interpretability. Croce et al. [2018] and Jiang et al. [2019] suggest comparing input text with similar documents to explain model predictions. These approaches provide insights into individual predictions and offer a broader view of the model's behavior by leveraging similarity to known instances.

## 2.4 Evaluating explanations

Evaluation of text explanations involves assessing how well generated explanations align with human rationales and the faithfulness of these rationales in influencing model predictions.

*ERASER* [DeYoung et al., 2020] evaluates rationales using multiple datasets with human-annotated rationales, proposing metrics to measure the alignment (*plausibility*) and *faithfulness* of these rationales. *Faithfulness* is often assessed by measuring the impact of perturbing or erasing important words on model output. Common evaluation metrics include *Deletion* and *Insertion*, which measure the impact on model performance when important words identified by the explainer are removed or added. For the *Deletion* metric, words are removed in order of importance to observe how model accuracy degrades. For the *Insertion* metric, words are added back in order of importance to examine how accuracy improves. The final score is typically obtained by taking the *area under the curve (AUC)* of accuracy as a function of the percentage of removed or added words [Petsiuk et al., 2018].

*Faithfulness* measures how accurately an explainer's highlighted important words align with the predictions of the black-box model. For example, if an explainer identifies specific words in a review as crucial for a positive sentiment prediction, faithfulness evaluates whether those words

truly influenced the model's decision. *Stability* checks if similar sentences yield similar explanations, ensuring consistency in the importance of words across slight variations in input [Alvarez Melis and Jaakkola, 2018]. Additionally, metrics like *Monotonicity* evaluate whether a model's performance improves with the incremental addition of important words [Luss et al., 2021]. This involves adding words back into a minimal context in order of their importance and observing whether the model's accuracy improves as expected, ensuring that the identified important words genuinely contribute positively to the model's predictions.

Several tools and libraries have been developed to address the challenges in evaluating explanation methods for text. *OpenXAI* [Agarwal et al., 2022b] provides a synthetic data generator, diverse real-world datasets, pre-trained models, state-of-the-art feature attribution methods, and quantitative metrics to assess faithfulness, stability, and fairness of explanations. *Ferret* [Attanasio et al., 2023] is a Python library for comparing explainable AI methods on transformer-based NLP models. It supports methods such as Gradient, Integrated Gradient, SHAP, and LIME, and includes metrics for evaluating faithfulness and plausibility. The *ERASER* benchmark [DeYoung et al., 2020] includes multiple datasets with human-annotated rationales and proposes metrics to assess the alignment and faithfulness of model-generated rationales with human rationales. *Captum* [Kokhlikyan et al., 2020] focuses on implementing explanation methods without emphasizing evaluation, while *Quantus* [Hedström et al., 2023] includes some evaluation metrics but lacks comprehensive benchmarks and up-to-date stability metrics. *CEBaB* [Abraham et al., 2022] introduces a benchmark dataset consisting of restaurant reviews and human-generated counterfactuals to study the causal effects of abstract, real-world concepts on model behavior. It allows for the comparison of various concept-based explanation methods, emphasizing the importance of aspect-level and review-level sentiment annotations to understand model behavior beyond mere input features.

As said in Section 1.5.2, despite significant advancements in developing tools and metrics for evaluating text explanations, there is still no broad consensus on best practices or universally accepted standards. The various metrics and benchmarks, though useful, lack universally precise definitions and often yield inconsistent results across different studies. Furthermore, Hsia et al. [2024] critically examines the effectiveness of saliency-based explanation metrics in NLP, specifically focusing on ERASER and EVAL-X metrics. The authors demonstrate that these metrics can be manipulated to show improved performance without genuinely enhancing the model's explainability. This exposes fundamental issues with these benchmarks and underscores the need for reevaluating their intended goals.

Using feature-attribution methods, it is possible leverage the score function $s$ to order tokens by their importance for a specific prediction. Let $\xi_{(k)}$ represent the $k^{\text{th}}$ most important token of $\xi$ for the prediction $f(\xi)$. Consequently, tokens can be arranged in descending order of importance as $\xi_{(1)}, \xi_{(2)}, \ldots, \xi_{(b)}$. Ideally, two feature importance methods should yield similar token orderings based on these importance scores. Additionally, the top-ranked tokens identified by a feature importance method (*e.g.*, LIME) are expected to overlap with the keywords identified by a rule-based method (*e.g.*, Anchors). However, as empirically demonstrated in Chapter 3 and highlighted throughout this manuscript, such consistency is not guaranteed. The discrepancies between explanations are significant because they indicate that different explainers can produce divergent and sometimes conflicting explanations. This, in turn, can lead to varied and potentially erroneous and dangerous conclusions.

# CHAPTER 3

# Sentence Highlighting *vs.* Keyword Identification in Text Models

The increasing use of complex machine learning algorithms in critical tasks involving text data has led to the development of various interpretability methods, which are essential for understanding and trusting the decisions made by models, particularly in high-stakes domains. Among local interpretability methods, two primary families have emerged: those that compute importance scores for each feature, allowing for *sentence highlighting*, and those that extract simple logical rules, identifying *keywords* when applied to text data. However, Lopardo and Garreau [2022] demonstrate that using different interpretability methods can lead to unexpectedly different explanations, even when applied to simple models where qualitative consistency would be expected. This highlights the need for careful consideration and validation of interpretability techniques to ensure their reliability and usefulness in practice. This chapter defines two representative methods from both families, namely LIME (Section 3.2.1) and Anchors (Section 3.2.2), and reports an empirical comparison from both qualitative (Section 3.3.1) and quantitative (Section 3.3.2) perspectives.

## 3.1 Introduction

As discussed in previous chapters, increasing complexity has been crucial for achieving state-of-the-art performance in natural language processing, allowing large and complex models to permeate high-impact applications. However, the opacity of these models limits their use in sensitive areas such as healthcare and the legal field due to inadequate explanations for individual predictions, hindering social acceptance.

To improve interpretability, numerous methods have been proposed over the past decade. This chapter focuses on local, *post hoc* explanations, which elucidate individual decisions of pre-trained models. These methods vary widely in their intrinsic functionalities. Methods like LIME [Ribeiro et al., 2016] assign attribution scores to features by fitting a linear model on the presence or absence of a feature. As explained in Section 2.3, this allows for sentence highlighting in text data, where each token (feature) has an associated weight. Rule-based methods, such as Anchors [Ribeiro et al., 2018], provide explanations through decision sets that maximize interpretability and accuracy [Lakkaraju et al., 2016]. **Extracting rules that locally approximate the model's behavior in text classification equates to identifying the keywords that best summarize the document under examination.**

As explained in Section 1.5.2, one of the main challenges of Explainable AI is the lack of universally accepted metrics to compare explanations. This is even more problematic when comparing different types of methods, such as rule-based methods and feature-importance methods (see Figure 3.1). Ideally, for the same prediction, the keywords identified by a rule-based method should be highlighted as important by a feature-importance method. Conversely, the top-ranked tokens identified by a feature-importance method (*e.g.*, LIME) are expected to overlap with the keywords identified by a rule-based method (*e.g.*, Anchors). However, due to the lack of uniformity in evaluations on one hand and the absence of theoretical foundations on the other, such consistency is not guaranteed. For instance, in Figure 3.1, LIME identifies "amazing" and "good" as the most influential words for the positive prediction, as they have the highest contribution scores. On the other hand, Anchors indicates that the presence of both "Nice" and "amazing" is sufficient to confidently produce the same prediction as the original input. The discrepancies between explanations are significant because they indicate that different explainers can produce divergent and sometimes conflicting explanations. This, in turn, can lead to varied and potentially erroneous and dangerous conclusions.

This chapter, primarily based on the work of Lopardo and Garreau [2022], compares the explanations provided by LIME and Anchors for text classification models through qualitative and quantitative experiments.

The rest of the chapter is organized as follows: Section 3.2 briefly recalls the methods under scrutiny. Section 3.3 presents the main findings, and Section 3.4 offers the conclusions. The code used for the comparison is available at `https://github.com/gianluigilopardo/anchors_vs_lime_text`, ensuring that experiments are reproducible.

## 3.2 Methods

In this section, the operational procedures of LIME (Section 3.2.1) and Anchors (Section 3.2.2) are briefly recalled, introducing the notation used in the process. For both methods, the official

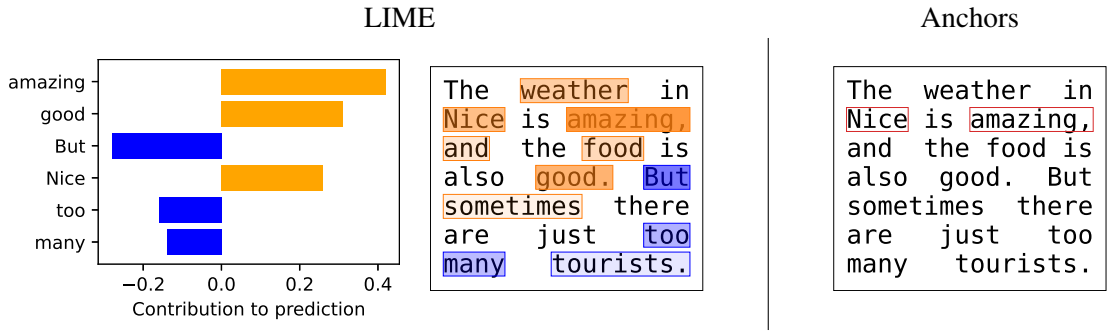


Figure 3.1 – **Comparison of LIME and Anchors explanations on a sentiment analysis model.** LIME provides feature importance scores and highlights contributions of individual words, while Anchors identifies a minimal set of features (highlighted words) sufficient for the same model prediction.

repositories * are used, with all default parameters, as these are the most commonly used configurations.

The main assumption is that the model $F$ takes as input the TF-IDF vectorization of the words, as defined in Section 2.2. This mapping is denoted by $\varphi$. Therefore, any prediction $f(z)$, for $z \in \mathcal{X}$, relies on $\varphi$. As will be demonstrated later, this choice clearly impacts the performance of the models.

### 3.2.1 LIME for text data

LIME [Ribeiro et al., 2016] provides explanations in the form of feature attribution, focusing on the presence or absence of individual words in the document to explain the prediction for $\xi$. Below, the main steps of LIME for textual data are summarized, with a detailed explanation and in-depth analysis available in Mardaoui and Garreau [2021].

1. Generate $n$ (by default, $n = 5000$) perturbed samples $x^{(1)}, \ldots, x^{(n)}$ from the original document $\xi$ by randomly *removing* words.
2. Obtain the predictions $f(x^{(i)})$ for all perturbed samples $i \in [n]$.
3. Train a weighted linear model on these samples to approximate the original model's behavior locally around $\xi$.

**Sampling.** Recall from Section 2.1 that $b = |\xi|$ is the number of tokens in the example $\xi$, and $d = |\mathcal{D}_\xi|$ is the number of unique words in its local dictionary.

The sampling procedure is as follows: for each perturbed document $x^{(i)}$, draw $s_i$, the number of deletions, uniformly at random from $[d]$. Then, draw a subset $S_i \subseteq \mathcal{D}_{x^{(i)}}$ of size $s_i$ uniformly at random, and replace all corresponding words in the document with the mask token `UNKWORDZ`. Specifically, all occurrences of a selected word are removed.

This implies that, by default, LIME provides explanations at the word level rather than the token level. Additionally, under the realistic assumption that the token `UNKWORDZ` was not part of

∗. LIME: https://github.com/marcotcr/lime.
Anchors: https://github.com/marcotcr/anchor

the corpus $\mathcal{T}$ on which the TF-IDF vectorizer was trained, its TF-IDF weight is zero. Therefore, replacing any word with the mask token is equivalent to removing it, as explained in Section 2.2.

**Surrogate model.** Weights $\pi_i$ are assigned to each perturbed sample $x^{(i)}$ based on their similarity to the original document $\xi$. A linear model is then fitted on the TF-IDF vectorizations of the perturbed samples, using the predictions $f(x^{(i)})$ as the target variable. The input to this linear model is the TF-IDF vector for each sample, where each word in the local dictionary is assigned a weight. Words that have been replaced or were not originally in $\xi$ have a weight of zero. The resulting linear model provides a visualization of the scores, indicating the importance of each word in making the prediction for $\xi$.

### 3.2.2 Anchors for text data

Anchors, introduced by Ribeiro et al. [2018], provide explanations in the form of logical conditions that *sufficiently* approximate the model's behavior locally. In the case of textual data, *anchors* are subsets of the tokens in the example $\xi$, and **the goal is to identify the keywords that maximize the *coverage*, while guarantee for high *precision*.** Anchors for text data is properly detailed in Chapter 4, while the main notions are as follows.

Recall from Eq. (2.2) that $\ell^\star$ is the predicted class for the example $\xi$, *i.e.*, the one class corresponding with higher confidence: $\ell^\star := \arg\max_{\ell \in \mathcal{C}} F_\ell(\xi)$. Hence, the prediction of interest is $f(\xi) = F_{\ell^\star}(\xi)$. Further define $y(z) := \arg\max_{\ell \in \mathcal{C}} F_\ell(z)$ as the predicted class for any generic document $z \in \mathcal{X}$.

The precision of an anchor $A$ associated to a prediction $f(\xi)$ is defined by Ribeiro et al. [2018] as $\text{Prec}(A) := \mathbb{E}\left[\mathbb{1}_{y(x)=\ell^\star} \mid A \in x\right]$, where the condition means that all words in $A$ belong to the sample $x$. Since $\text{Prec}(A)$ is generally not available in practice, an empirical estimate of the precision is computed from new samples $x^{(i)}$ of the text.

The core idea of Anchors is to pick an anchor with high precision while preserving some notion of globality. More precisely, Anchors solves (approximately)

$$A \in \underset{\text{Prec}(A) \geq 1-\varepsilon}{\arg\max} \ \text{cov}(A)\,, \tag{3.1}$$

where, by default, $\varepsilon = 0.05$ and the coverage $\text{cov}(A)$ is defined as the probability that $A$ applies to samples. However, due to Anchors' sampling, maximizing the coverage is equivalent to minimizing the length of $A$ (see Lopardo et al. [2023a] for more details).

**Sampling.** The sampling procedure ensures that the behavior of the classifier $f$ is observed in a local neighborhood of $\xi$ while keeping the anchor $A$ fixed. For a given document $\xi$ and each candidate anchor $A \subseteq \xi$, the sampling is performed as follows:

1. Create $n$ (by default, $n = 10$) samples $x^{(1)}, \ldots, x^{(n)}$ by generating identical copies of $\xi$.
2. For each token $\xi_j \in \xi$ not in $A$, randomly select each occurrence $x_j^{(i)}$ with probability $1/2$ and replace selected words with the token `UNK`.
3. Query the classifier on these perturbed samples to compute the empirical precision of the anchor $A$ using Eq. (3.1).

Note that, given the TF-IDF vectorization, reaplacing a token with the `UNKWORDZ` or `UNK` does not make any difference (assuming that both tokens were unkown to the vectorizer $\varphi$): both terms will have zero weight, *i.e.*, masking a token with these words is equivalent to remove them.

**Anchor selection.** Anchors are selected based on their empirical precision and coverage. The goal is to find an anchor $A$ with high precision (by default, $\mathrm{Prec}(A) \geq 0.95$) while maximizing its coverage, defined as the probability that the anchor applies to samples. The user is provided with the shortest anchor that satisfies the precision condition, indicating the key words that consistently lead to the same model prediction.

**Comparison with LIME.** One main difference between LIME and Anchors lies in their sampling methods. LIME removes words based on their presence in the local dictionary $\mathcal{D}_\xi$: if a word is selected, all its occurrences in $\xi$ will be removed. Anchors, on the other hand, treat tokens in $\xi$ as independent, removing individual occurrences randomly. Another notable difference is that LIME evaluates samples on the prediction $f(x)$ by leveraging the confidence scores for each class. Conversely, Anchors directly uses the predicted labels $y(x)$: the precision is assessed based solely on whether the sample receives the same classification as the original example.

## 3.3 Experiments

The main results are presented by comparing LIME and Anchors for text data when applied to simple classification models. The experiments are conducted on three review datasets: Restaurants, Yelp, and IMDB. The task is binary sentiment analysis, where label 1 denotes a positive review and label 0 denotes a negative review, as in Example 2.1.1. In practice, the model is defined as $F : \mathcal{X} \to [0,1]^2$. The predicted class is $y(z) := \arg\max_{\ell \in \{0,1\}} F(z)$. Without loss of generality, the focus is on explaining positive predictions, *i.e.*, examples $z \in \mathcal{T}$ such that $y(z) = 1$, hence $f(z) = F_1(z)$.

In Section 3.3.1, a qualitative comparison of LIME and Anchors is provided by examining individual explanations. In Section 3.3.2, the $\ell$-index is introduced as a metric to evaluate the quality of explanations for text data, measuring their faithfulness to the classifier. Unless otherwise specified, the figures report the average LIME coefficient and the occurrence count for Anchors, each based on 100 runs of the default algorithms.

### 3.3.1 Qualitative Evaluation

#### 3.3.1.1 Simple Decision Rules

This section begins by examining simple decision rules that rely on the presence or absence of specific words in the text. This task, sometimes referred to as *shortcut identification* [Bastings et al., 2022], involves detecting straightforward patterns or features that the model uses to make predictions. These rules are typically represented by indicator functions, which denote whether a word is present in the text. Four cases of increasing complexity are explored to illustrate how decision rules can be constructed and interpreted based on these simple patterns.

**Presence of a given word.** Consider a simple decision rule that returns 1 or 0 based on the presence or absence of a specific predefined word $w_j \in \mathcal{D}$. Specifically, $f(z) := \mathbb{1}_{w_j \in z} = \mathbb{1}_{\varphi(z)_j > 0}$.

Examine an example $\xi$ where $w_j \in \xi$, resulting in $f(\xi) = 1$. In this scenario, **both methods behave as expected**: LIME assigns a high weight to $w_j$ and negligible weights to the other words, while Anchors correctly identifies the keyword, producing the anchor $A = \{w_j\}$, as illustrated in Figure 3.2.

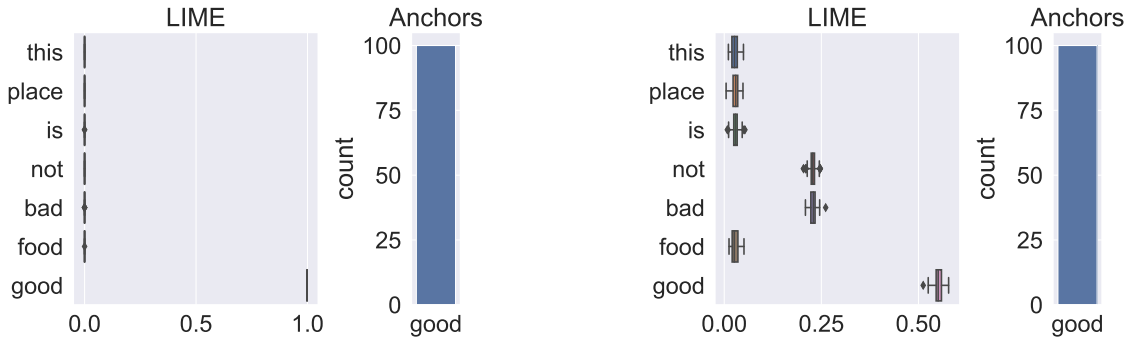


Figure 3.2 – **Comparison of LIME and Anchors explanation on a simple decision rule.** The classifiers $\mathbb{1}_{good \in z}$ (left panel) and $\mathbb{1}_{(not \in z \text{ and } bad \in z) \text{ or } good \in z}$ (right panel) are applied to the same review. Anchors makes no difference between the two: the presence of the word *good* is sufficient to get a positive prediction.

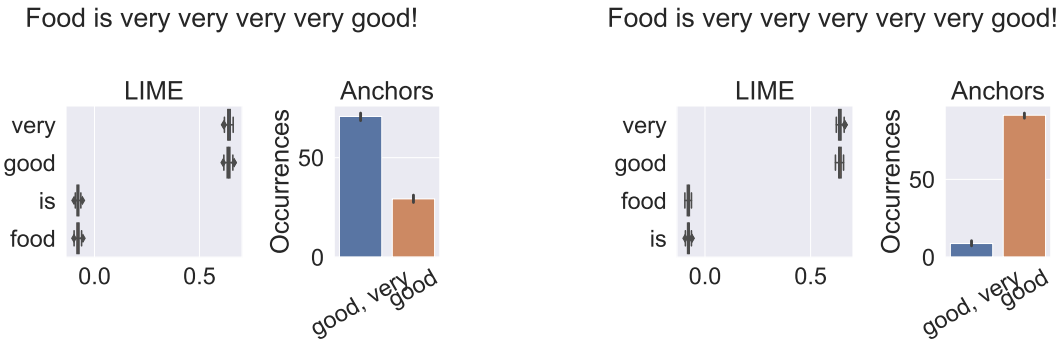


Figure 3.3 – **Making a word disappear from the explanation by adding one occurrence.** The classifier $\mathbb{1}_{(very \in z \text{ and } good \in z)}$ is applied when $m_{very} = 4$ (left) and $m_{very} = 5$ (right).

**Small decision tree.** Consider a small decision tree that checks for the presence of words $w_1$ and $w_2$ or word $w_3$, *i.e.*,

$$f(z) = \mathbb{1}_{(w_1 \in z \text{ and } w_2 \in z) \text{ or } w_3 \in z} \,.$$

Analyze an example $\xi$ where $w_1, w_2, w_3 \in \xi$. LIME assigns the same positive weight to $w_1$ and $w_2$, a higher weight to $w_3$, and negligible weights to all other words, as shown in Mardaoui and Garreau [2021]. In contrast, Anchors only identifies $w_3$ as the anchor. Ideally, both methods would highlight the same words since they are all important for the decision. However, **Anchors does not consider $w_1$ and $w_2$ in its explanation because the presence of $w_3$ alone is *sufficient* for a positive classification, and $\{w_3\}$ is a shorter anchor than $\{w_1, w_2\}$**, thus it has higher coverage.

**Presence of several words.** To generalize the previous example, consider a model that classifies documents based on the presence of a set of words. Let $J = [k] \subseteq [|\mathcal{D}|]$ represent a set of distinct indices associated with distinct words in the dictionary. The model considered is:

$$f(z) = \prod_{j \in J} \mathbb{1}_{w_j \in z} = \prod_{j \in J} \mathbb{1}_{\varphi(z)_j > 0} \,.$$

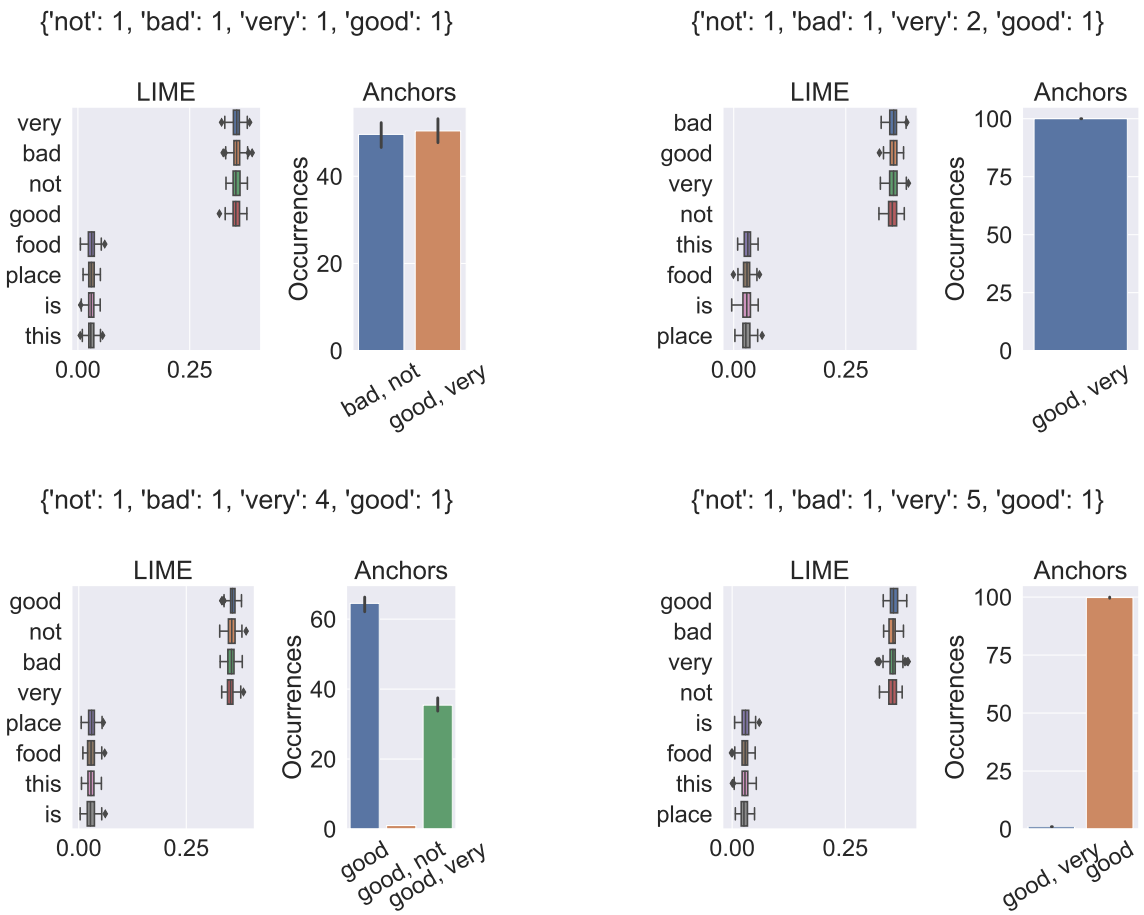


Figure 3.4 – **Anchors explanations depend on words multiplicity.** Comparison on the classifier $\mathbb{1}_{(not \in z \text{ and } bad \in z) \text{ or } (very \in z \text{ and } good \in z)}$ when only $m_{very}$ is changing. LIME assigns the same weight to the important words.

In this scenario, **LIME assigns the same importance to each word in** $J$, regardless of their occurrences (in accordance to Mardaoui and Garreau [2021, Proposition 3]). In contrast, Anchors' explanations are influenced by the number of occurrences of each word (Proposition 6 in Lopardo et al. [2023a]). Specifically, **if the number of occurrences of a word in** $J$ **exceeds a certain threshold, it disappears from the anchors** (see Figure 3.3). This behavior is unexpected and undesirable, especially since this threshold is beyond control. An explicit formula for the threshold, depending on the parameters of Anchors, is provided in Chapter 4.

**Presence of disjoint subsets of words.** Now, consider two disjoint sets of indices $J_1 = [k_1] \subseteq [|\mathcal{D}|]$ and $J_2 = \{k_1 + 1, \dots, k_2\} \subseteq [|\mathcal{D}|]$ with the same cardinality, $|J_1| = |J_2|$. We examine the model

$$f(z) = \prod_{j \in J_1} \mathbb{1}_{w_j \in z} \cdot \prod_{j \in J_2} \mathbb{1}_{w_j \in z} = \prod_{j \in J_1} \mathbb{1}_{\varphi(z)_j > 0} \cdot \prod_{j \in J_2} \mathbb{1}_{\varphi(z)_j > 0} \,,$$

and an example $\xi$ such that $w_j \in \xi$ for all $j \in J_1$ and for all $j \in J_2$.

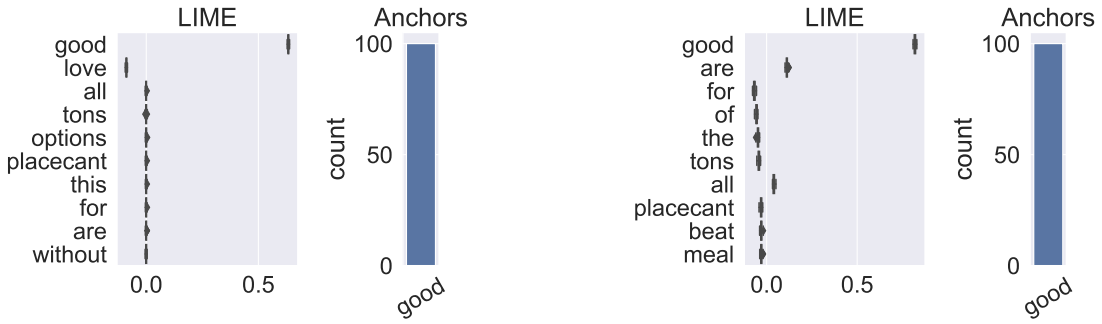


Figure 3.5 – **Comparison of LIME and Anchors on logistic models** with $\lambda_{\text{love}} = -1$, $\lambda_{\text{good}} = +5$, and $\lambda_w = 0$ for other words (left), vs. $\lambda_{\text{good}} = 10$ and $\lambda_w \sim \mathcal{N}(0, 1)$ for other words (right), applied to the same document. *Good* is the most important word for classification in both cases.

LIME assigns the same weight to words in $J_1$ and words in $J_2$. However, Anchors' explanations depend on the multiplicities of words (see Figure 3.4): as the occurrences of one word in $J_1$ (or in $J_2$) increase, the presence of other words in the same subset becomes *sufficient* to get a positive prediction.

#### 3.3.1.2 Logistic models

The focus now shifts to logistic models, adapting the definition from Section 1.1.3 and Eq. (1.2) to the current setting. Let $\sigma : \mathbb{R} \to [0, 1]$ be the sigmoid function, with $\lambda_0 \in \mathbb{R}$ as the intercept and $\lambda \in \mathbb{R}^D$ as fixed coefficients. For any document $z$, the model considered is:

$$f(z) = \mathbb{1}_{\sigma(\lambda_0 + \lambda^\top \varphi(z)) > \frac{1}{2}}.$$

**Sparse Case** Consider a scenario where only two coefficients, $\lambda_1 > 0$ and $\lambda_2 < 0$, are nonzero, with $|\lambda_1| > |\lambda_2|$. In this case, LIME assigns nonzero weights to $w_1$ and $w_2$, and zero importance to the other words. In contrast, Anchors only identifies $w_1$ (see Figure 3.5), as expected, since $w_1$ is the primary word influencing a positive prediction.

**Arbitrary coefficients.** Consider the case where $\lambda_1 \gg 0$, and $\lambda_j \sim \mathcal{N}(0, 1)$ for $j \geq 2$. LIME assigns a large weight to $w_1$ and small weights to the other words, while Anchors only identifies $w_1$ as it is the most influential word for the decision (see Figure 3.5).

When applied to simple if-then rules based on the presence of specific words, the analysis reveals that Anchors exhibit unexpected behavior related to the multiplicities of these words in a document, whereas LIME captures the classifier's support. The experiments on logistic models in Figure 3.5 demonstrate that even when both methods agree on the most important words, LIME captures more detailed information than Anchors.

### 3.3.2 Quantitative Evaluation

When applying a logistic classifier $f$ (Section 3.3.1.2) on top of a TF-IDF vectorizer $\varphi(\cdot)$ (Section 2.2), the contribution of a word $w_j$ is given by $\lambda_j \varphi(()z)_j$, allowing words to be unambiguously ranked by importance. To evaluate an explainer's *faithfulness* (Section 2.4), the similarity

TABLE 3.1 – Comparison between LIME and Anchors in terms of $\ell$-index and computing time.

| | $\ell$-index | | | time ($s$) | | |
|---|---|---|---|---|---|---|
| | Restaurants | Yelp | IMDB | Restaurants | Yelp | IMDB |
| LIME | $0.96 \pm 0.17$ | $0.95 \pm 0.22$ | $0.94 \pm 0.23$ | $0.21 \pm 0.05$ | $0.45 \pm 0.22$ | $0.73 \pm 0.44$ |
| Anchors | $0.67 \pm 0.44$ | $0.29 \pm 0.43$ | $0.22 \pm 0.35$ | $0.19 \pm 0.27$ | $3.83 \pm 13.95$ | $33.87 \pm 165.08$ |

between the $N$ most important words for the interpretable classifier, $\Lambda_N(z)$, and the $N$ most important words according to the explainer, $E_N(z)$, is measured. This method assesses the explainer's ability to accurately identify the most important words for classifying a document $z$.

The $\ell$-index for the explainer $E$ is defined as:

$$\ell_E := \frac{1}{|\mathcal{T}|} \sum_{z \in \mathcal{T}} J\left(E_N(z), \Lambda_N(z)\right) ,$$

where $J(\cdot, \cdot)$ is the *Jaccard similarity* and $\mathcal{T}$ is the test corpus.

Since $N$ cannot be fixed a priori for Anchors, the experiments are conducted as follows. For any document $z$, let $A(z)$ be the obtained anchor and set $N = |A(z)|$. The Jaccard similarity is then computed as $J\left(A(z), \Lambda_{|A(z)|}(z)\right)$ for Anchors and $J\left(L_{|A(z)|}(z), \Lambda_{|A(z)|}(z)\right)$ for LIME.

Table 3.1 reports the $\ell$-index and computation time for LIME and Anchors across three datasets. LIME demonstrates high performance in extracting the most important words while requiring less computational time compared to Anchors. **An anchor $A$ is a minimal set of words that is sufficient (with high probability) to yield a positive prediction, but it does not necessarily coincide with the $|A|$ most important words for the prediction.**

## 3.4 Conclusion

The absence of standard evaluation methods for Explainable AI presents a significant challenge, especially when comparing feature importance-based methods with rule-based methods. This lack of standardized metrics exacerbates the discrepancies observed in machine learning interpretability.

In this chapter, explanations for text data from two popular methods, LIME and Anchors, were compared, highlighting differences and unexpected behaviors when applied to simple models. Observations indicate significant discrepancies: the set of words $A$ extracted by Anchors does not coincide with the set of $|A|$ words with the largest interpretable coefficients determined by LIME.

To compare different forms of explanations (feature importance-based and rule-based), the $\ell$-index was proposed, measuring the ability of explainers to identify the most important words. Experiments show that LIME performs better than Anchors in this task while requiring fewer computational resources. Additionally, experiments revealed unusual behaviors in Anchors, underscoring the need for deeper analysis to understand these anomalies.

Contributing to these discrepancies is the lack of robust theoretical foundations for machine learning interpretability. Despite substantial advancements in prediction accuracy, the interpretability of models—which is crucial for trust and actionable insights—remains inadequately grounded in theory. This gap likely contributes to the inconsistencies observed in explanations provided by different models.

This manuscript aims to address these critical issues by contributing towards the establishment of a more solid theoretical basis for machine learning interpretability, facilitating more reliable and consistent explanations across different methods.

# CHAPTER 4

# An In-Depth Analysis of Anchors for Text Data

Anchors [Ribeiro et al., 2018] has been introduced as a representative post-hoc, rule-based interpretability method. For text data, it explains a decision by highlighting a small set of words (an anchor) such that the model produces similar outputs when these words are present in a document. Empirical observations, as shown in Chapter 3, indicate that Anchors exhibit unexpected behavior even on simple models, such as an unclear dependence on word multiplicities. In this chapter, the first theoretical analysis of Anchors is presented [Lopardo et al., 2023a], assuming an exhaustive search for the best anchor. After formalizing the algorithm for text classification in Section 4.2, explicit results are provided in Section 4.4 for different classes of models using TF-IDF vectorization, where words are replaced by a fixed out-of-dictionary token when removed. The analysis covers models such as elementary if-then rules (Section 4.4.2) and linear classifiers (Section 4.4.3). This analysis is then leveraged to gain insights into the behavior of Anchors for any differentiable classifiers. For neural networks, it is empirically shown in Section 4.5 that Anchors select words corresponding to the highest partial derivatives of the model with respect to the input, reweighted by the inverse document frequencies.

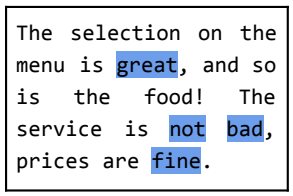


Figure 4.1 – **Anchors explaining the positive prediction of a black-box model** $f$ on an example $\xi$ from the Restaurant review dataset. The anchor $A = \{\textit{great, not, bad, fine}\}$ (in blue), having length $|A| = 4$ is selected. Intuitively, these four words together ensure a positive prediction by $f$ with high probability (`precision` : 0.97), while being not too uncommon (`coverage` : 0.12).

## 4.1 Introduction

In Chapter 3, a comparison between Anchors and LIME revealed some unusual behavior. While LIME has been extensively studied [Mardaoui and Garreau, 2021] and its behaviors are well-understood, a robust theoretical foundation for Anchors has been lacking. The paper by Lopardo et al. [2023a] conducted the first in-depth analysis of Anchors, and this chapter presents those findings.

The focus is on Anchors [Ribeiro et al., 2018], particularly its implementation for text data. For a given prediction, the core idea of Anchors is to provide a simple rule that yields the same prediction with high probability if it is satisfied. These rules can be formulated as the presence of a list of tokens in the document to be explained and are presented as such to the user (see Figure 4.1).

A key question explored is whether Anchors will highlight the important words in the explanation, especially when the model to be explained is intrinsically interpretable and the important words for the prediction are known with absolute certainty.

To provide a comprehensive understanding, the basic concepts of Anchors and its mechanism for text classification are first explained in Section 4.2. Next, the definition of a more tractable, exhaustive version of the algorithm is detailed in Section 4.3, which constitutes the central focus of this study.

To assess the efficacy of Anchors for text data, a theoretical and empirical analysis of its behavior on explainable classifiers is conducted in Section 4.4. This analysis offers new insights that can be extended to broader classes of models. Specifically, in Section 4.5, a surprising result on neural networks is empirically demonstrated, providing valuable insights into the real-world applications of Anchors for explaining document classifiers.

Finally, in Section 4.6, conclusions are drawn and the findings of the study are summarized. Some unexpected results highlight the importance of theoretical analysis for explainers. The insights presented in this chapter are intended to be useful for researchers and practitioners in the field of natural language processing, aiding in the accurate interpretation of the explanations provided by Anchors. Additionally, the framework designed for this analysis can be valuable to the explainability community, both in designing new methods with solid theoretical foundations and in analyzing existing ones.

**Contributions.** This chapter presents the first theoretical analysis of Anchors for text data [Lopardo et al., 2023a], based on the default implementation available on GitHub * (as of June 2024). The analysis primarily relies on simplifying the combinatorial optimization procedure by considering an *exhaustive* version of Anchors, using an out-of-dictionary token when removing words, and assuming TF-IDF vectorization as a preprocessing step. Specifically, the contributions are as follows:

- The Anchors algorithm for text classification is dissected, demonstrating that the sampling procedure can be described simply as an *i.i.d.* Bernoulli removal of words not belonging to the anchor (Proposition 4.2.1).
- The exhaustive version is shown to be stable with respect to perturbations of the precision function, justifying the study of the exhaustive Anchors algorithm (Proposition 4.3.1 and 4.3.2).
- It is established that if the classifier ignores some words, they will not appear in the anchor selected by exhaustive Anchors (Proposition 4.4.1).
- Exhaustive Anchors for simple if-then rules is proven to provide meaningful explanations, though words can be excluded from the explanation if their multiplicity is too high (Proposition 4.4.2).
- Exhaustive Anchors is demonstrated to select words associated with the most positive coefficients reweighted by the inverse document frequency for all linear classifiers (Proposition 4.4.3 and 4.4.4).
- It is empirically shown that exhaustive Anchors picks the words associated with the most positive partial derivatives scaled by the inverse document frequency for neural networks (Section 4.5).

All theoretical claims are supported by mathematical proofs, available in Section A.1, and numerical experiments, detailed in Section A.3. The code for the experiments is available at `https://github.com/gianluigilopardo/anchors_text_theory`. Unless otherwise specified, the experiments use the official implementation of Anchors with all default options.

**Related work.** Among the numerous methods for machine learning interpretability proposed in recent years (Section 1.4), rule-based methods have emerged as popular contenders. This popularity is partly due to users' preference for rule-based explanations over other types [Lim et al., 2009, Stumpf et al., 2007]. *Hierarchical decision lists* [Wang and Rudin, 2015] are valuable for understanding the global behavior of a model by prioritizing the most interesting cases. Lakkaraju et al. [2016] introduced a compromise between accuracy and interpretability to extract small and disjoint rules, which are easier to interpret, and introduced the concept of *coverage*. Alternatively, Barbiero et al. [2022] proposed learning simple logical rules alongside the model's parameters to maintain accuracy.

Several rule-based approaches focus on *local interpretability* (Section 1.4.2), based on the idea that any black-box model can be accurately approximated by a simpler, more understandable model around a specific instance to explain. For instance, *LORE* [Guidotti et al., 2018a] uses a decision tree where the explanation is a list of logical conditions satisfied by the instance within the tree. A central aspect of perturbation-based methods (Section 1.4.6) is the sampling scheme. Delaunay et al. [2020] modified Anchors sampling for tabular data, using the *Minimum Description Length Principle* discretization [Fayyad and Irani, 1993] to learn the minimal number of intervals

*. `https://github.com/marcotcr/anchor`

needed to separate instances from distinct classes. Amoukou and Brunel [2022] proposed *Minimal Sufficient Rules*, similar to Anchors for tabular data, extended to regression models, capable of directly handling continuous features without discretization.

Few local, post-hoc explainability techniques exist for text data (Section 2.3). Among them, *LIME* [Ribeiro et al., 2016] and *SHAP* [Lundberg and Lee, 2017] provide explanations using a linear model as a local surrogate, trained on perturbed samples of the instance to explain. While LIME and SHAP assign a weight to each word of the example, *Anchors* extracts the *minimal* subset of words that is *sufficient* to achieve, with *high probability*, the same prediction as the example. Delaunay et al. [2020] extended Anchors by also considering the absence of words.

Moreover, the lack of standard evaluation methods for Explainable AI (Sections 1.5.2 and 2.4) complicates the comparison between different interpretability techniques, especially when comparing feature importance methods with rule-based approaches. In Chapter 3, LIME and Anchors were compared experimentally, revealing some unexpected behaviors from Anchors that are not yet fully understood. This observation highlighted the need for a deeper theoretical analysis to explain these anomalies.

The primary concern of this work is to provide theoretical guarantees for interpretability methods. For feature importance methods, Lundberg and Lee [2017] offers insights in the context of linear models for kernel SHAP (even with some caveats mentioned in Section 1.5.3), while Mardaoui and Garreau [2021] specifically examines LIME for text data, building on Garreau and Luxburg [2020]. These papers also analyze the behave of LIME on simple if-then rules and linear models. To the best of our knowledge, the question of theoretical guarantees for rule-based methods had not been addressed until Lopardo et al. [2023a], the work explained in this paper, which was the first to provide such an analysis.

## 4.2 Anchors for text data

This section presents the operating procedure of Anchors for text data, as introduced by Ribeiro et al. [2018]. After specifying the setting and notation in Section 4.2.1, the key notions of *precision* and *coverage* are detailed in Section 4.2.2. The algorithm is described in Section 4.2.3, with further details on the sampling scheme provided in Section 4.2.4.

### 4.2.1 Setting and Notation

With the notation presented in Section 2.1 at hand, consider the problem of explaining the decision of a classifier $f$ taking documents as input, denoted by $z$ for a generic document and $\xi$ for the specific example of interest.

The analysis is restricted to *binary classification*, where $f(z) = \mathbb{1}_{F(z)\in R}$, with $F : \mathcal{X} \to \mathbb{R}^p$ being a measurable function and $R$ a collection of intervals of $\mathbb{R}^p$. It is assumed that $F$ relies on a TF-IDF *vectorization* of the documents (Section 2.2), specifically $F = h \circ \varphi$, where $\varphi$ is a deterministic mapping $\mathcal{X} \to \mathbb{R}^D$ and $h : \mathbb{R}^D \to \mathbb{R}^p$. For simplicity, and without loss of generality, the example $\xi$ is always classified as positive, *i.e.*, $f(\xi) = 1$. The models considered take the form:

$$f(z) = \mathbb{1}_{h(\varphi(z))\in R}\,. \tag{4.1}$$

**Definition 4.2.1 (Anchor).** An *anchor* is defined as any non-empty subset of $[b]$, corresponding to a preserved set of tokens in $\xi$. The set of all candidate anchors for the example $\xi$ is denoted

by $\mathcal{A}$. The length of an anchor $A \in \mathcal{A}$, $|A|$, is defined as the number of tokens in the anchor. In practice, an anchor $A$ for a document $\xi$ is represented as a non-empty sublist of tokens present in the document, which is the output of Anchors (illustrated in Figure 4.1).

### 4.2.2 Precision and Coverage

The key concepts of precision and coverage, as defined by Ribeiro et al. [2018], briefly mentioned in Chapter 3, are detailed in the following.

**Definition 4.2.2 (Precision).** The *precision* of an anchor $A \in \mathcal{A}$ is the probability that a local perturbation of $\xi$ is classified as $\xi$. Given the restriction to $\xi$ such that $f(\xi) = 1$, the precision can be expressed as:

$$\text{Prec}\,(A) = \mathbb{E}\left[f(x) = 1 \mid A \in x\right] = \mathbb{E}_A\left[f(x) = 1\right] = \mathbb{P}_A\left(h(\varphi(x)) = 1\right)\,, \tag{4.2}$$

where the expectation is taken with respect to $x$, a random perturbation of $\xi$ that still contains all the words included in the anchor $A$.

The sampling of $x$ is further detailed in Section 4.2.4. For an anchor containing all the words of $\xi$, the precision is exactly 1, while smaller anchors generally have lower precision.

Large anchors with sizes comparable to $b = |\xi|$ are less interesting from an interpretability perspective (the text in Figure 4.1 would be completely highlighted). To quantify this, the notion of *coverage* is used, defined as the proportion of documents in the corpus (the dataset on which the vectorizer is fitted) that contain the anchor. For instance, the coverage of the anchor in Figure 4.1 is $0.12$, meaning that $12\%$ of the reviews contain it.

Precision and coverage are crucial to the Anchors algorithm: in essence, **Anchors seeks an anchor of maximal coverage with a prescribed precision.** This process is detailed in the next section.

### 4.2.3 The Algorithm

In practice, the coverage can be costly to compute, and in many cases, a corpus is not available when the prediction is explained. Since anchors with smaller lengths tend to have larger coverage, a natural solution, used in the default implementation, is to minimize the length instead of maximizing the coverage, leading to:

$$\underset{A \in \mathcal{A}}{\text{Minimize}}\,|A|\;,\;\text{subect to}\;\;\text{Prec}\,(A) \geq 1 - \varepsilon\,, \tag{4.3}$$

where $\varepsilon > 0$ is a pre-determined tolerance threshold (set to $0.05$ in practice). The lower $\varepsilon$ is, the harder it is to find an anchor satisfying Eq. (4.3).

The exact precision of a specific anchor $A \in \mathcal{A}$ is unknown, as the expectation in Eq. (4.2) can be very costly to compute, in general. The strategy used by Ribeiro et al. [2018] is to approximate $\text{Prec}\,(A)$ by $\widehat{\text{Prec}}_n(A)$, an empirical approximation, defined in Section 4.3. The optimization problem in Eq. (4.3) is generally intractable due to the large cardinality of $\mathcal{A}$ in practical scenarios. Consequently, the default implementation applies the *KL-LUCB* [Kaufmann and Kalyanakrishnan, 2013] algorithm to identify a subset of rules with high precision. This subset is then used as a representative of all candidate anchors, finding an approximate solution to Eq. (4.3). In Lopardo et al. [2023a], this optimization procedure is not considered, and instead, an exhaustive version of Anchors is described in Section 4.3.

| | $\xi_1$ | $\xi_2$ | $\xi_3$ | $\xi_4$ | $\xi_5$ | $\xi_6$ | $\xi_7$ | $\xi_8$ | $\xi_9$ |
|---|---|---|---|---|---|---|---|---|---|
| $\xi$ | the | quick | brown | fox | jumps | over | the | lazy | dog |
| $x_1$ | **UNK** | **UNK** | brown | fox | jumps | over | the | lazy | dog |
| $x_2$ | the | quick | brown | **UNK** | jumps | **UNK** | the | lazy | dog |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $x_n$ | the | quick | brown | **UNK** | jumps | over | the | lazy | **UNK** |

Figure 4.2 – **Illustration of Anchors sampling scheme.** Anchors sampling is performed in three main steps: copies generation, random selection, word replacing. Here, for instance, for the fourth token (*fox*), $\{2, n\} \in S_4$ ($B_4 \geq 2$), so the second and the $n$-th copies are considered for replacement.

### 4.2.4 The Sampling

The sampling procedure (briefly introduced in Section 3.2.2) used to compute the precision of an anchor (see Eq. (4.2)) is detailed here. The goal is to observe the behavior of the model $F$ in a local neighborhood of $\xi$, while keeping the set of words in the anchor $A$ fixed. In the official implementation, this is achieved by setting `use_unk_distribution=True` (the default choice). Formally, for a given example $\xi$ and for a candidate anchor $A \in \mathcal{A}$, perturbed samples $x^{(1)}, \ldots, x^{(n)}$ are generated as follows:

1. *Copies generation*: Create $n$ (by default, $n = 10$) identical copies $x^{(1)}, \ldots, x^{(n)}$ of the example to explain $\xi$.
2. *Random selection*: For each token with index $k \in [b]$ not in the anchor, draw $B_k \sim \text{Bin}(n, 1/2)$, representing the number of copies to be perturbed (words in the candidate anchor are the blue columns of Table 4.2).
3. *Word replacement*: For each token not in the anchor, independently and uniformly draw a set $S_k$ of cardinality $B_k$ from the copies to be perturbed. Replace the words in copies whose indices are in $S_k$ with the token `UNK`.

The perturbation distribution described here differs from Ribeiro et al. [2018], which involves replacing selected words with others having the same *part-of-speech* tag based on similarity in an embedding space. **Replacing words with a predefined token can create meaningless sentences that might mislead a classifier by producing unrealistic samples** (as discussed in Section 1.5.1). However, this option is not implemented in the official release. A similar solution, described in Chapter 5, is one of the main contributions of this manuscript. Alternatively, when using Anchors official implementation, it is possible to replace selected words using BERT [Devlin et al., 2019].

This work relies on the `UNK`-replacement because (i) the default choices are likely the most used by Anchors users, necessitating interpretation and theoretical guarantees, and (ii) as detailed in Section 2.2, in the case of TF-IDF vectorization, the `UNK`-replacement exactly replicates word removals. Experiments show that the results hold when BERT-replacement is applied (see Section A.3.8).

Anchors sampling procedure is similar to that of LIME for text data [Ribeiro et al., 2016], with the crucial difference that LIME removes *all occurrences* of a given word when it is selected for removal. More details can be found in Mardaoui and Garreau [2021]. The sampling procedure can be described more simply, as shown in Section A.1.1:

**Proposition 4.2.1 (Equivalent sampling).** *The sampling process described above is equivalent to replacing, for any sample $x^{(i)}$, each token $x_j^{(i)}$ not in $A$ independently with probability $1/2$.*

Intuitively, parsing each line of Table 4.2, Anchors flips an imaginary coin for each word not belonging to the anchor, replaces it in the perturbed example if the coin shows heads, and keeps it if it shows tails. Proposition 4.2.1 provides a simple description of $M_j$, the random variable representing the multiplicity of word $w_j$ in the perturbed sample $x$. For any given anchor $A$, $M_j \sim a_j + \mathrm{Bin}(m_j - a_j, 1/2)$, where $a_j$ is the number of occurrences of $w_j$ in $A$.

## 4.3 Exhaustive p-Anchors

This section presents the central object of our study, *exhaustive* p*-Anchors*. Essentially, this is a formalized version of the original combinatorial optimization problem described in Eq. (4.3) for any evaluation function $\mathrm{p} : \mathcal{A} \to \mathbb{R}$. The procedure is detailed in Section 4.3.1, followed by an exploration of a key stability property that motivates further investigations in Section 4.3.2.

### 4.3.1 Description of the Algorithm

The optimization problem in Eq. (4.3) can be decomposed into two steps. First, select all anchors in $\mathcal{A}$ such that $\mathrm{Prec}\,(A) \geq 1 - \varepsilon$. This subset of anchors is denoted as $\mathcal{A}_1(\varepsilon)$. Note that $\mathcal{A}_1(\varepsilon) \neq \emptyset$ since the full anchor $[b]$ has precision 1. Then, among these anchors, keep those with minimal length, resulting in $\mathcal{A}_2(\varepsilon)$. At this point, it is not clear from Eq. (4.3) which anchors should be selected, so the ones with the highest precision are chosen. In cases of equality (*e.g.*, several anchors with precision 1), the set is referred to as $\mathcal{A}_3(\varepsilon)$. If $\mathcal{A}_3(\varepsilon)$ contains more than one element, an anchor is selected uniformly at random.

Algorithm 1 formally describes this procedure for a generic evaluation function $\mathrm{p} : \mathcal{A} \to \mathbb{R}$, illustrated in Figure 4.3. When using p, the sets constructed are denoted as $\mathcal{A}_k^{\mathrm{p}}(\varepsilon)$, and the selected anchor is denoted as $A^{\mathrm{p}}(\varepsilon)$.

The goal is to provide a flexible framework: Algorithm 1 can be used with $\mathrm{p} = \widehat{\mathrm{Prec}}_n$ or $\mathrm{p} = \mathrm{Prec}$ as the selection function, or any other function that approximates Prec well. When $\mathrm{p} = \mathrm{Prec}$, this version of the algorithm is called *exhaustive Anchors*, whereas when $\mathrm{p} = \widehat{\mathrm{Prec}}_n$ it is called *empirical Anchors*.

Empirical Anchors closely resembles the original Anchors; **the main difference is that empirical Anchors considers all possible anchors, while the original uses an efficient approximate procedure**, which is not considered here. Another difference is that empirical Anchors selects anchors with maximal precision in the third step. This is not necessarily the case with the default implementation, which uses an approximate procedure. Nonetheless, the chosen anchors tend to have high precision, and empirical Anchors and the default implementation produce very similar outputs in practice, as demonstrated in Section A.3.2.

### 4.3.2 Stability with Respect to the Evaluation Function

Applying Algorithm 1 to functions taking similar values on $\mathcal{A}$ leads to similar results, as shown in the following.

**Algorithm 1** An overview of exhaustive p-Anchors.

**Require:** set of candidate anchors $\mathcal{A}$, selection function $\mathrm{p} : \mathcal{A} \to \mathbb{R}$, tolerance threshold $\varepsilon$

1: **select** $\mathcal{A}_1^{\mathrm{p}}(\varepsilon) = \{A \in \mathcal{A} \text{ s.t. } \mathrm{p}(A) \geq 1 - \varepsilon\}$
2: **select** $\mathcal{A}_2^{\mathrm{p}}(\varepsilon) = \underset{A' \in \mathcal{A}_1^{\mathrm{p}}(\varepsilon)}{\arg\min} |A'|$
3: **select** $\mathcal{A}_3^{\mathrm{p}}(\varepsilon) = \underset{A' \in \mathcal{A}_2^{\mathrm{p}}(\varepsilon)}{\arg\max} \mathrm{p}(A')$
4: **select** $A^{\mathrm{p}}(\varepsilon) \in \mathcal{A}_3^{\mathrm{p}}(\varepsilon)$ uniformly at random
5: **return** $A^{\mathrm{p}}(\varepsilon)$

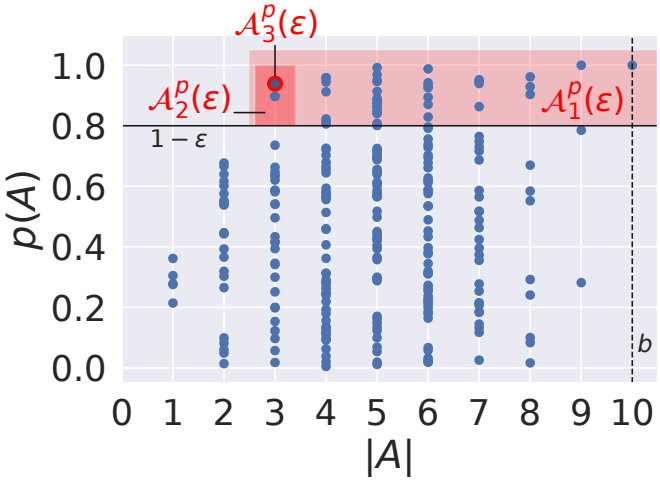


Figure 4.3 – **An illustration of Algorithm 1** with evaluation function $\mathrm{p} = \mathrm{Prec}$. Each blue dot is an anchor, with its length on the horizontal axis and its value for p on the vertical axis. Here, $\varepsilon = 0.2$ and the maximal length of an anchor is $b = 10$ (the length of $\xi$). In the end, the anchor $A$ such that $|A| = 3$ and $\mathrm{p}(A) = 0.9$ is selected (red circle).

**Proposition 4.3.1 (Stability of exhaustive p-Anchors).** *Let $\varepsilon > 0$ be a tolerance threshold, $\mathrm{p} : \mathcal{A} \to \mathbb{R}$ be an evaluation function, and set $A^\star := A^{\mathrm{p}}(\varepsilon)$, the output of exhaustive* p*-Anchor. Assume that (i) $\mathrm{p}(A^\star) \geq 1 - \varepsilon/4$, and (ii) $\mathrm{p}(A) \leq 1 - 3\varepsilon/4$ for any $A \in \mathcal{A}_2^{\mathrm{p}}(\varepsilon) \setminus \{A^\star\}$. Let $q : \mathcal{A} \to \mathbb{R}$ be another evaluation function such that*

$$\delta := \sup_{A \in \mathcal{A}} |p(A) - q(A)| < \frac{\varepsilon}{4}\,. \tag{4.4}$$

*Then $A^{\mathrm{q}}(\varepsilon - \delta) = A^\star$.*

In plain words, if $A^\star$ is a solution with a high value for the chosen p function, and $q$ is a good approximation of p, then running Algorithm 1 on $q$ instead of p will yield approximately the same result. **This is the key motivation for studying** Prec **instead of** $\widehat{\mathrm{Prec}}$**, and later considering further approximations to** Prec**.** One can study directly Prec, or an approximation thereof, and gain insights into the output of the original algorithm. Proposition 4.3.1 is proved in Section A.1.2.

Note that, since the values of the p function are perturbed, an anchor with a smaller length than $A^\star$ could cross the $1 - \varepsilon$ barrier and become a solution for exhaustive $q$-Anchors if the same tolerance threshold $\varepsilon$ is maintained. This is the case for the anchor with length two and the highest

value of $\mathrm{p}$ in Figure 4.3. Therefore, the tolerance threshold must decrease: Proposition 4.3.1 cannot be improved to show that $A^{\mathrm{q}}(\varepsilon) = A^\star$. However, having a large value of $\mathrm{p}$ for $A^{\mathrm{p}}$ (assumption (i)) is not strictly necessary. What is important is that the gap between $A^{\mathrm{p}}$ and the anchor with the second-largest value of the $\mathrm{p}$ function in $\mathcal{A}_2^{\mathrm{p}}$ cannot be filled by $q$ (assumption (ii)). Otherwise, an anchor with the same length could get a larger value for $q$ than $A^{\mathrm{p}}$ and be selected in the final step.

As a first application of Proposition 4.3.1, consider the *empirical precision*,

$$\widehat{\mathrm{Prec}}_n(A) := \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{f(x^{(i)})=1}\,,$$

where $n$ is the number of perturbed samples with $A$ fixed, as described in Section 4.2.4. Then the empirical precision satisfies the following:

**Proposition 4.3.2** ($\widehat{\mathrm{Prec}}_n(A)$ **uniformly approximates** $\mathrm{Prec}$)**.** *Recall that $b$ denotes the number of tokens in $\xi$. Let $\delta > 0$. With probability higher than $1 - 2^{b+1}\mathrm{e}^{-2n\delta^2}$,*

$$\forall A \in \mathcal{A}, \qquad \left|\widehat{\mathrm{Prec}}_n(A) - \mathrm{Prec}\,(A)\right| \leq \delta\,. \tag{4.5}$$

In particular, Proposition 4.3.2 guarantees that $\widehat{\mathrm{Prec}}_n$ and $\mathrm{Prec}$ satisfy Eq. (4.4) with high probability, as soon as $n \gg b/\varepsilon^2$. This is the main motivation for studying *exhaustive Anchors* in the next section: Proposition 4.3.1 shows that *exhaustive Anchors* and *empirical Anchors* will output the same result with high probability. Proposition 4.3.2 is proved in Section A.1.3.

## 4.4 Analysis on explainable classifiers

Before presenting the main results, the implications of the vectorizer under consideration are described in Section 4.4.1. The analysis then examines Anchors' behavior when applied to simple rule-based classifiers in Section 4.4.2 and to linear models in Section 4.4.3. All claims are supported by mathematical proofs, provided in Section A.1, and validated by reproducible experiments.

### 4.4.1 Vectorizers and Immediate Consequences

Recall the TF-IDF vectorizer defined in Definition 2.2.1. As said, once the TF-IDF vectorizer is fitted on a corpus $\mathcal{T}$, the vocabulary is fixed. Thus, if a word is not part of the initial corpus $\mathcal{T}$, its TF-IDF term is zero. As described in Section 4.2.4, Anchors perturbs documents by replacing words with a fixed token `UNK`. It is assumed that the word `UNK` is not present in the initial corpus. Therefore, **replacing any word with this token is equivalent to simply removing it from the perspective of TF-IDF**.

Models considered in this paper are always trained on a (non-normalized) TF-IDF vectorization $\varphi$ as in Definition 2.2.1. However, it is shown in Section A.1.9 that the same results hold for the $\ell_2$-normalized TF-IDF vectorization, defined as

$$\forall j \in [D], \qquad \varphi(z)_j := \frac{m_j(z)v_j}{\sqrt{\sum_{j=1}^{D} m_j(z)^2 v_j^2}}\,.$$

This normalization is the default in the `scikit-learn` † implementation of TF-IDF.

†. https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

Given that the models follow the form of Eq. (4.1), **the exact location of the words in the document does not matter** when the vectorizer is applied. Therefore, when computing precision, only the occurrences of word $w_j$ in anchor $A$ matter. An anchor is thus written as $A = (a_1, \dots, a_d, \dots, a_D)$. Since $a_j \leq m_j$ for all $j \in [D]$, $a_j = 0$ for any $j > d$. Thus, $A$ can be written as $A = (a_1, \dots, a_d)$ without ambiguity.

With this notation, and following the discussion after Proposition 4.2.1, the TF-IDF of $w_j$ in the perturbed sample $x$ will satisfy:

$$\varphi(x)_j = M_j v_j \sim (a_j + \mathrm{Bin}(m_j - a_j, 1/2))v_j \,, \tag{4.6}$$

where $M_j$ is the random multiplicity of the word $w_j$. Intuitively, $M_j$ corresponds to $a_j$ occurrences of $w_j$ which cannot be removed, plus a random number of occurrences depending on the sampling. This has several important consequences in the analysis, the first being:

**Proposition 4.4.1 (Dummy features).** *Let $f$ be defined as in Eq. (4.1) and assume that $h$ does not depend on coordinate $j$. Let $\varepsilon > 0$ be a tolerance threshold. Then, for any anchor $A \in \mathcal{A}_3^{\mathrm{Prec}}(\varepsilon)$, $a_j = 0$.*

Proposition 4.4.1 is a natural property: if the model does not depend on word $w_j$, then it should not appear in the explanation. This concept, often investigated in the interpretability literature, was originally introduced by Friedman [2004]. Here, the vocabulary of Sundararajan et al. [2017b] is used, which introduced the notion as an axiom for feature importance.

Note that Proposition 4.4.1 is not satisfied by the empirical version (with $\mathrm{p} = \widehat{\mathrm{Prec}}_n$), nor by the default implementation of Anchors (see Section A.3.3). Proposition 4.4.1 is proved in Section A.1.4.

### 4.4.2 Simple decision rules

This section focuses on classifiers that rely on the presence or absence of specific words, as in Section 3.3.1.1. In this setting, the function $h$ introduced in Eq. (4.1) takes a straightforward form. With the TF-IDF vectorizer, the *presence* (resp. *absence*) of word $w_j$ in $x$ corresponds to the condition $\varphi(x)_j > 0$ (resp. $\varphi(x)_j = 0$).

Thus, $h$ is the projection on the relevant coordinates, and using Eq. (4.6), the precision of any given anchor can be computed. Here, the case of a model classifying documents based on the presence of a set of words is shown. Additional cases are presented in Section A.3.5.

**Proposition 4.4.2 (Presence of a set of words).** *Assume that $m_1 > \cdots > m_k$. Let $J = \{1, \dots, k\}$ be a subset of $[d]$, and assume that the model is defined as*

$$f(z) = \mathbb{1}_{w_j \in z \,, \forall j \in J} = \prod_{j \in J} \mathbb{1}_{w_j \in z} = \prod_{j \in J} \mathbb{1}_{\varphi(z)_j > 0} \,.$$

*Define the quantities $B := \lfloor \log_2 1/\varepsilon \rfloor$ and $c_0 := \inf\{c \in [k],\ \prod_{\ell=1}^{k-c+1}(1 - 2^{-\ell}) \geq 1 - \varepsilon\}$. If there exists $j \in J$ such that $m_j > B$, then the anchor $A_{c_0}$ such that $a_j = 1$ for all $j \in [\![k - c_0 + 1, k]\!]$ and $a_j = 0$ otherwise, will be selected by exhaustive Anchors. On the contrary, if $m_j < B$ for all $j \in [d]$, the anchor $A_J$ such that $a_j = 1$ for all $j \in [k]$ and $a_j = 0$ otherwise, will be selected.*

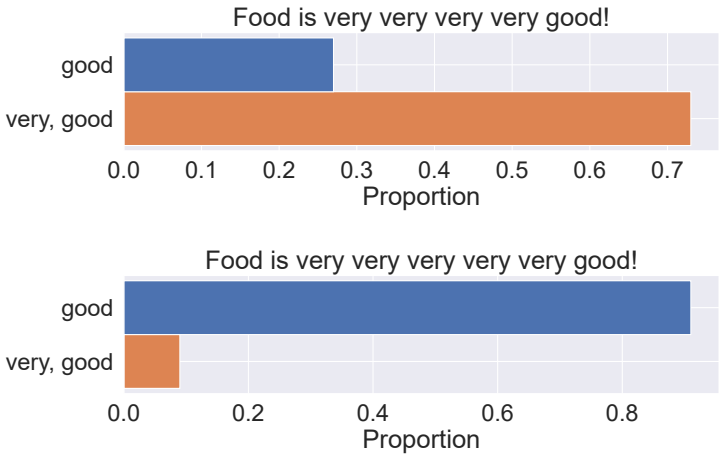


Figure 4.4 – **Effect of adding one occurrence on Anchors explanation.** The model predicts 1 if *very* and *good* are present. The count corresponds to the appearance of the token(s) in the selected anchor over 100 runs of Anchors. When the multiplicity of *very* in the document crosses the breakpoint value $B = 4$ (by default, $\varepsilon = 0.05$), it disappears from the selected anchor with high probability (top panel: $m_{\text{very}} = 4$, bottom panel: $m_{\text{very}} = 5$).

Note that $A_J$ contains all the words with index in $J$, *i.e.*, each word in the support of the classifier $f$. In contrast, $A_{c_0}$ contains the $c_0$ words with the lowest multiplicity among those indexed in $J$. Proposition 4.4.2 is proved in Section A.1.5.

As a concrete example, consider a model classifying reviews as positive if both the words "very" and "good" are present. The review "Food is very good!" is classified as positive, and Anchors will extract $\{very, good\}$ as an anchor, which is sensible. However, if the multiplicity of the word "very" (resp. "good") exceeds the breakpoint value $B = 4$ (by default, $\varepsilon = 0.05$), Proposition 4.4.2 predicts that Anchors will only extract {"good"} (resp. {"very"}). **This example effectively shows a word disappearing from an explanation simply because its multiplicity crosses an arbitrary threshold.** This phenomenon is observed in practice with the default implementation of Anchors (see Figure 4.4), with some variability due to the sampling and the approximate optimization scheme of Anchors.

### 4.4.3 Linear classifiers

This section shifts focus to *linear classifiers*. For any document $z$, the classifier is defined as:

$$f(z) = \mathbb{1}_{\lambda^\top \varphi(z) + \lambda_0 > 0}\,, \tag{4.7}$$

where $\lambda \in \mathbb{R}^D$ is a vector of coefficients, and $\lambda_0 \in \mathbb{R}$ is an intercept. In the context of Eq. (4.1), this translates to $h(x) = \lambda^\top x + \lambda_0$ and $R = (0, +\infty)$.

Eq. (4.7) encompasses several significant examples, two of which are empirically investigated:

— **The Perceptron** [Rosenblatt, 1958b], which predicts exactly as in Eq. (4.7).

— **Logistic Models**, which predict 1 if $\sigma(\lambda^\top \varphi(z) + \lambda_0) > 1/2$, where $\sigma : \mathbb{R} \to [0, 1]$ is the *logistic function*. Since $\sigma(x) > 1/2$ if, and only if, $x > 0$, they can also be rewritten as in Eq. (4.7).

This list is not exhaustive; refer to Hastie et al. [2009, Chapter 4] for a more comprehensive overview.

In this setting, starting from Eq. (4.6), it is shown that the precision satisfies a Berry-Esseen-type statement [Berry, 1941, Esseen, 1942]:

**Proposition 4.4.3** (**Precision of a Linear Classifier**). *Let $\lambda, \lambda_0$ be the coefficients associated with the linear classifier defined by Eq. (4.7). Assume that for all $j \in [d]$, $\lambda_j v_j \neq 0$.*

*Define, for all $A \in \mathcal{A}$,*

$$L(A) := \frac{-\lambda_0 - \frac{1}{2}\sum_{j=1}^{d} \lambda_j v_j (m_j + a_j)}{\sqrt{\frac{1}{4}\sum_{j=1}^{d} \lambda_j^2 v_j^2 (m_j - a_j)}}. \tag{4.8}$$

*Let $\overline{\Phi} := 1 - \Phi$, where $\Phi$ denotes the cumulative distribution function of a $\mathcal{N}(0,1)$. Then, for any $A \in \mathcal{A}$ such that $|A| \leq b/2$,*

$$\left|\mathrm{Prec}(A) - \overline{\Phi}(L(A))\right| \leq C \cdot \left(\frac{\max_j \lambda_j^2 v_j^2}{\min_j \lambda_j^2 v_j^2}\right)^{3/2} \cdot \left(\frac{\max_j m_j}{\min_j m_j}\right)^{3/2} \cdot \frac{1}{\sqrt{d}}, \tag{4.9}$$

*where $C \approx 7.15$ is a numerical constant.*

In other words, when $d$ is large, **the precision of an anchor for a linear classifier can be well approximated by $\overline{\Phi} \circ L$**, and a (local) version of Proposition 4.3.1 can be used to study exhaustive p-Anchors with $\mathrm{p} = \overline{\Phi} \circ L$ instead of exhaustive Anchors.

Nevertheless, a very good fit between the two terms is observed even for small values of $d$ (see Section A.3.4). Proposition 4.4.3 is proven in Section A.1.6. In Section A.1.9, it is shown that in the case of normalized TF-IDF, a constant with the same rate appears.

Typical values for $v_j$ and $m_j$ in this setting range between 1 and 10 (see Section A.3.1). The primary assumption here is that there are no zero components in $\lambda$. Additionally, the result holds only for anchors with a length less than half the document length. This is a realistic constraint, as explanations based on more than half of the document are uncommon in practice and generally lack interpretability. It is possible to refine the constant by ensuring that the anchors are even smaller in relative size.

In view of Proposition 4.3.1, the focus can now be on exhaustive $\overline{\Phi} \circ L$-Anchors. Let $\gamma := \lambda_0 + \sum_j \lambda_j v_j m_j$. Note that, since $f(\xi) = 1$, $\gamma > 0$. Also, define

$$\mathcal{A}_+ := \{A \in \mathcal{A} \mid a_j > 0 \Rightarrow \lambda_j > 0\}, \tag{4.10}$$

as the set of anchors with support corresponding to words with a positive influence.

**Proposition 4.4.4** (**Approximate Precision Maximization**). *Assume that $\lambda_1 v_1 > \lambda_2 v_2 > \cdots > \lambda_d v_d$, with at least one $\lambda_j$ greater than zero. Assume that $\lambda_0 > -\gamma/2$. Then Algorithm 1 applied to the selection function $\mathrm{p} := \overline{\Phi} \circ L$ will select an anchor $A^{\mathrm{p}} \in \mathcal{A}_+$ such that there exists $j_0 \in [d]$ with the following properties: for all $j < j_0$, $a_j = m_j$, $a_{j_0} \leq m_j$, and for all $j \geq j_0$, $a_j = 0$.*

In plain words, Proposition 4.4.4 implies that **for a linear classifier, Anchors keeps only words with a positive influence on the prediction**. Moreover, it **selects the words having the highest $\lambda_j v_j$ values first**, adding them to the anchor until the precision condition is met. This is a reassuring property of Anchors. Proposition 4.4.4 is proven in Section A.1.7.

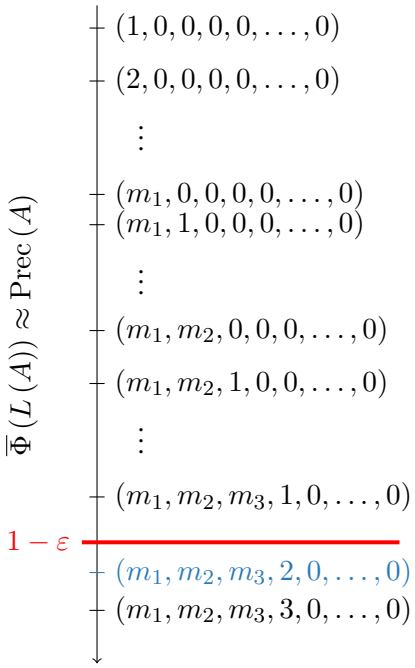


Figure 4.5 – **Illustration of Proposition 4.4.4.** For linear models, the algorithm prioritizes words with the highest $\lambda_j v_j$ values. The minimal anchor that satisfies the precision condition $\overline{\Phi}(L(A)) \approx \text{Prec}(A) \geq 1 - \varepsilon$ is selected, which in this example is $A = (m_1, m_2, m_3, 2, 0, \ldots, 0)$.

To demonstrate this phenomenon, the following experiment was conducted. A logistic model was first trained on three review datasets, achieving accuracies between $85\%$ and $88\%$ on the test set. Anchors was then run with the default setting 10 times on positively classified documents. For each document, the Jaccard similarity between the anchor $A$ and the first $|A|$ words ranked by $\lambda_j v_j$ was measured. The average Jaccard index is reported in Table 4.1, confirming Proposition 4.4.4.

Note that the official Anchors implementation does not apply step 3 of Algorithm 1. When the prediction is *easy* (for instance, $h(\varphi(\xi)) \geq 0.75$ or $g(\varphi(\xi)) \geq 0.85$), $\mathcal{A}_3$ realistically contains more than one anchor, and the algorithm will randomly select among them. This explains the different similarity between the full dataset and the *hard* subset.

It is also noteworthy that the individual multiplicities do not affect the ranking of the $w_j$s, unlike the discussion following Proposition 4.4.2. This behavior is consistent across all models tested (see Section A.3.5).

## 4.5 Anchors on neural networks

In this section, empirical results for neural networks are presented, linking the explanations provided by Anchors with the partial derivatives of the model with respect to the input. Intuitively, while examining a specific prediction, Anchors generates local perturbations of the example to be explained. The behavior of a neural network in such a local neighborhood of the example, at the first order, is approximately linear. This implies that Proposition 4.4.3 roughly holds true, taking

TABLE 4.1 – Validation of Proposition 4.4.4. Average Jaccard similarity between the anchor $A$ and the first $|A|$ words ranked by $\lambda_j v_j$ for a logistic model on positive documents and low-confidently classified subset ($\texttt{pr} = h(\varphi(\xi)) < 0.85$, or $\texttt{pr} < 0.75$).

| | Restaurants | Yelp | IMDB |
|---|---|---|---|
| full | $0.69 \pm 0.2$ | $0.35 \pm 0.2$ | $0.32 \pm 0.2$ |
| $\texttt{pr} < 0.85$ | $0.78 \pm 0.2$ | $0.74 \pm 0.2$ | $0.45 \pm 0.2$ |
| $\texttt{pr} < 0.75$ | $0.82 \pm 0.1$ | $0.80 \pm 0.2$ | $0.59 \pm 0.1$ |

TABLE 4.2 – Average Jaccard similarity between the extracted anchor $A$ and the first $|A|$ words ranked by $\lambda_j v_j$ for a 3 (top table), 10 (middle), and 20-layers (bottom) feed-forward neural network for positively classified documents and low-confidently ($\texttt{pr} = h(\varphi(\xi)) < 0.85$, or $\texttt{pr} < 0.75$) classified subsets.

| | | Restaurants | Yelp | IMDB |
|---|---|---|---|---|
| 3-layers | full | $0.68 \pm 0.2$ | $0.50 \pm 0.3$ | $0.45 \pm 0.3$ |
| | $\texttt{pr} < 0.85$ | $0.68 \pm 0.2$ | $0.88 \pm 0.1$ | $0.52 \pm 0.3$ |
| | $\texttt{pr} < 0.75$ | $0.74 \pm 0.2$ | $0.82 \pm 0.1$ | $0.68 \pm 0.2$ |
| 10-layers | full | $0.76 \pm 0.2$ | $0.56 \pm 0.2$ | $0.55 \pm 0.2$ |
| | $\texttt{pr} < 0.85$ | $0.80 \pm 0.2$ | $0.78 \pm 0.1$ | $0.79 \pm 0.2$ |
| | $\texttt{pr} < 0.75$ | $0.83 \pm 0.1$ | $0.69 \pm 0.2$ | $0.81 \pm 0.2$ |
| 20-layers | full | $0.73 \pm 0.1$ | $0.60 \pm 0.3$ | $0.63 \pm 0.2$ |
| | $\texttt{pr} < 0.85$ | – | $0.81 \pm 0.2$ | $0.69 \pm 0.2$ |
| | $\texttt{pr} < 0.75$ | – | – | $0.74 \pm 0.1$ |

as linear coefficients

$$\forall j \in [d]\,, \qquad \lambda_j := \frac{\partial g(\varphi(x))}{\partial \varphi(x)_j}\,,$$

where $\frac{\partial h(\varphi(x))}{\partial \varphi(x)_j}$ is the partial derivative of the model $g$ with respect to the word $w_j$.

In practice, this means that **Anchors selects the words corresponding to the highest partial derivatives of the model with respect to the input, reweighted by the inverse document frequencies,** until the precision condition is met.

To validate this conjecture, three feed-forward neural networks were trained on three datasets. For each document in the test set, the Jaccard similarity between the anchor $A$ and the first $|A|$ words ranked for all $j \in [d]$ by $\lambda_j v_j = \frac{\partial g(\varphi(x))}{\partial \varphi(x)_j} v_j$ was measured. This approach is similar to the experiments in the previous section. Details on the training are reported in Section A.3.7, with each model achieving approximately $90\%$ accuracy. Anchors was run with default settings 10 times on positively classified documents to account for the randomness in Anchors optimization.

Table 4.2 shows the results for the three networks. There is a significant overlap between the anchors selected by Anchors and the subset suggested by the analysis, becoming a near match for examples that are hard to predict.

As in the previous section, when the prediction is *easy*, *i.e.*, the classifier is very confident about the prediction, the anchor is selected at random among many candidates. This is because it requires a strong perturbation (removing many words) to change the prediction of a confidently classified document.

Since these models perform better than linear models, the confidence is often extremely high, resulting in a more random selection process and thus lower similarity values. More details about the experiments are available in Section A.3.7.

Compared to the Anchors algorithm for text classification, obtaining explanations for a prediction in this manner would be a faster and more efficient procedure. This is because the randomness due to the optimization scheme is avoided, making it more efficient for obtaining explanations on neural networks and, it is conjectured, any other differentiable classifier. However, this would not be a fully model-agnostic approach, as it requires knowledge of the model gradient and the inverse document frequencies for each example to be explained.

This is a somewhat surprising result: without the theoretical analysis and empirical evidence, it would intuitively be expected that explanations for this class of models would be obtained from the gradient reweighted by the input (*i.e.*, the entire vectorization).

Lopardo et al. [2023a] conjecture that for any differentiable classifier, it is possible to predict the behavior of Anchors by extending the results for linear classifiers, considering a first-order approximation of the model.

## 4.6 Conclusion

This chapter presented Lopardo et al. [2023a]: the first theoretical analysis of Anchors. Specifically, the implementation for textual data was formalized, providing insights into the sampling procedure. Anchors behavior was then studied on simple if-then rules and linear models. An approximate, tractable version of the algorithm was introduced, closely aligned with the default implementation. The analysis showed that Anchors provides meaningful results when applied to these models, supported by experiments with the official implementation. Finally, theoretical claims about explainable classifiers were exploited to obtain empirical results for neural networks, yielding a surprising result that links the classifier gradient to the importance of words for a prediction. When access to the model is available, this result can be used as a faster and more efficient method of obtaining explanations.

This work uncovered some unexpected findings that highlight the importance of theoretical analysis in the development of explainability methods. The insights presented in this chapter may be valuable for researchers and practitioners in natural language processing who seek to correctly interpret Anchors explanations. Furthermore, the analysis framework developed can aid the explainability community in designing new methods based on sound theoretical foundations and in scrutinizing existing ones.

# CHAPTER 5

# Faithful and Robust Local Interpretability for Textual Predictions

Local and model-agnostic interpretability methods are versatile but can exhibit unexpected behaviors (Chapter 3) and lack theoretical foundations. Popular explainers like Anchors and LIME sometimes fail on simple tasks and models, making them as opaque as the predictions they aim to clarify. While theoretical analyses exist for LIME [Mardaoui and Garreau, 2021], Anchors were largely unexplored before Lopardo et al. [2023a] (presented in Chapter 4), highlighting the need for a deeper understanding of these methods. This chapter introduces FRED (Faithful and Robust Explainer for textual Documents) as a novel method for interpreting predictions over text [Lopardo et al., 2023b]. FRED offers three key insights to explain a model's prediction: (1) it identifies the minimal set of words in a document whose removal has the strongest influence on the prediction, (2) it assigns an importance score to each token, reflecting its influence on the model's output, and (3) it provides counterfactual explanations by generating examples similar to the original document but leading to a different prediction. The reliability of FRED is established through formal definitions (Section 5.2) and theoretical analyses on interpretable classifiers (Section 5.3). Additionally, empirical evaluation against state-of-the-art methods demonstrates the effectiveness of FRED in providing insights into text models (Section 5.4).

## 5.1 Introduction

Previous chapters have emphasized that local and model-agnostic interpretability methods are particularly well-suited for explaining predictions made by any model for a specific instance without requiring any knowledge of the underlying model. This versatility makes them applicable to a broader range of scenarios compared to other methods that typically intervene during model training or require access to model parameters.

In Chapter 3, it was empirically demonstrated that even popular explainers like Anchors and LIME can exhibit unexpected behaviors, especially on simple tasks and models. Many interpretability methods lack a theoretical foundation, and their behavior on simple and interpretable models is often unclear. Each explainer employs various internal mechanisms, such as sampling, local approximations, and importance measures, which can significantly influence the final explanations (see Section 1.5.3). These mechanisms are often ignored or understudied, making the explainers as opaque as the predictions they aim to clarify.

While theoretical analyses exist for LIME, providing a comprehensive understanding of its behavior on tabular data [Garreau and Luxburg, 2020, Garreau and von Luxburg, 2020], text [Mardaoui and Garreau, 2021], and images [Garreau and Mardaoui, 2021], Anchors remained largely unexplored before Lopardo et al. [2023a]. Using a poorly understood explainer on a complex model can lead to misinterpretations of the model's behavior. Without a clear understanding of the method used and its internal mechanisms, drawing accurate conclusions can be challenging and may even lead to incorrect conclusions. Chapter 4 presented the first theoretical analysis of Anchors to address this gap, particularly for text classification models.

With these analyses available, including on different methods (feature importance-based and rule-based) but following the same principle of studying a model's behavior in the neighborhood of the instance to be explained, it becomes natural to compare the three key dimensions of post-hoc perturbation-based methods [Covert et al., 2021]: 1) how the method removes features, 2) what model behavior the method explains, and 3) how the method summarizes each feature's influence. The paper Lopardo et al. [2023b] leverages insights from these methods to propose a new method for explaining text-based models, ideally encapsulating the advantages of previous methods.

In Lopardo et al. [2023b], *FRED* (Faithful and Robust Explainer for text Documents) is introduced as a novel interpretability framework for text classification and regression tasks. FRED provides three insights to explain a model's prediction: (1) it **identifies the keywords**, *i.e.*, the minimal set of words in a document whose removal has the strongest influence on the prediction, (2) it assigns an importance score to each token, allowing for **sentence highlighting**, and (3) it offers **counterfactual explanations** by generating examples similar to the original document but leading to a different prediction. An illustration of FRED applied to a sentiment analysis task is shown in Figure 2.1.

**Contributions.** This chapter introduces FRED (Faithful and Robust Explainer for text Documents), a novel interpretability framework for text classification and regression tasks, as first presented in Lopardo et al. [2023b]. The framework provides a comprehensive set of tools for interpreting model predictions through various innovative techniques. Specifically, the contributions are as follows:

— FRED identifies the minimal set of words in a document whose removal has the strongest influence on the prediction, providing a clear understanding of crucial features (Section 5.2).

— Each token is assigned an importance score reflecting its impact on the model's output, facilitating a granular analysis of feature relevance (Section 5.2).
— Counterfactual explanations are provided by generating examples similar to the original document but leading to different predictions, enhancing the interpretability of model behavior (Section 5.2).
— A novel sampling scheme that leverages tokens' *part-of-speech* tags is introduced, optimizing the efficiency and effectiveness of the explanations (Section 5.2).
— A rigorous theoretical analysis of FRED's behavior on interpretable classifiers is conducted, ensuring it meets expectations on simpler models (Section 5.3).
— Empirical evaluations against well-established explainers on a variety of models, including state-of-the-art models, demonstrate FRED's effectiveness, particularly on more complex models and larger documents (Section 5.4).
— Theoretical claims are substantiated in Appendix B.1, supported by numerical experiments detailed in Appendix B.2.
— The empirical results highlight that FRED performs better on modern models and larger documents, making it more suitable for realistic cases (Section 5.4).

The code for FRED and the experiments is available at `https://github.com/gianluigilopardo/fred`.

### 5.1.1 Related work

**FRED falls under the category of local, post-hoc and model-agnostic interpretability methods within the field of machine learning interpretability** (see Section 1.4). Local methods explain predictions for individual data points (Section 1.4.2), while post-hoc methods are applied to already trained models, with no intervention during the design phase (Section 1.4.4). Model-agnostic explainers work on any black-box model, without needing access to their internal parameters, requiring only repeated queries. Methods like LORE [Guidotti et al., 2018a] and LIME [Ribeiro et al., 2016] approximate the model locally for a specific instance using a decision tree and a linear model as local surrogates, respectively. Anchors [Ribeiro et al., 2018] extracts provable rules that guarantee the model's prediction (Chapter 4).

More precisely, **FRED is a perturbation-based approach** (Section 1.4.6). As stated in Section 1.5.1, these methods leverage sampling mechanisms to perturb the instance to explain, and therefore evaluate the model change in prediction, but risk generating out-of-distribution data [Hase et al., 2021], leading to inaccurate explanations and vulnerability to adversarial attacks [Slack et al., 2020]. Some works addressed sampling issues of popular explainers. For instance, LORE employs a genetic algorithm for a more realistic sampling, Delaunay et al. [2020] improves Anchor sampling for tabular data, while Amoukou and Brunel [2022] extends *Minimal Sufficient Rules* (similar to Anchors) to regression models, handling continuous features without discretization.

While LIME and SHAP [Lundberg and Lee, 2017] explain predictions by feature importance, Anchors identifies a compact set of features guaranteed to produce the same prediction with high probability [Lopardo et al., 2023a]. Studies show that users prefer rule-based explanations [Lim et al., 2009, Stumpf et al., 2007], such as hierarchical decision lists [Wang and Rudin, 2015], which reveal global behavior, and [Lakkaraju et al., 2016], which balance accuracy and interpretability with smaller, disjoint rules. LIME and SHAP assign importance scores to tokens, while Anchors targets the most significant token set for the prediction. FRED accomplishes both tasks.

Counterfactual explanations (Sections 1.4.9 and 2.3.3), which explore what changes to the input text would cause a different model prediction, offer valuable insights into model behavior. This concept is explored by Wachter et al. [2017b], Pawelczyk et al. [2021,0], emphasizing the importance of generating not only accurate but also plausible counterfactuals to build trust in the model's decision-making process. FRED offers counterfactual explanations by generating examples similar to the original document but leading to different predictions, allowing users to see which slight changes to the text can alter the model's decision-making process.

FRED leverages the *explaining by removing* strategy, where removing features reveals their influence on predictions. While this approach is established for tabular data and images [Covert et al., 2021], it is underexplored for text. FRED isolates token sets within a document and measures the confidence drop upon removal to quantify the impact of each token. It pinpoints a concise subset of words crucial for the prediction and assigns an importance value to each token reflecting its influence on the model's output.

The field of interpretable machine learning often prioritizes practical application over formal guarantees, leading to explanations that may not be reliable [Marques-Silva and Ignatiev, 2022]. To address this, La Malfa et al. [2021] propose a method for generating robust explanations in text models, focusing on minimal word subsets sufficient for prediction and resistant to minor input changes. Inspired by [Garreau and Luxburg, 2020] and its adaptation to text data for LIME and Anchors [Mardaoui and Garreau, 2021, Lopardo et al., 2023a], this chapter includes a theoretical analysis to ensure FRED behaves as expected on well-understood models like linear models and shortcut detection. This check is crucial to guarantee that explanations reflect the model's true inner workings.

## 5.2 FRED

This section introduces FRED, a novel explainer designed to provide faithful and robust explanations for text classification and regression tasks. FRED employs a perturbation-based approach, analyzing the model's behavior on slightly altered versions of the original text. When presented with an example to explain, FRED first generates a perturbed sample (as detailed in Section 5.2.2).

Then, FRED explains model predictions through three key functionalities:

1. **Identifying keywords**: FRED identifies the minimal subset of tokens within the example that, when removed, cause a significant decline in the model's prediction confidence, exceeding a predefined threshold.
2. **Sentence highlighting**: FRED assigns an importance score to each token, reflecting its impact on the final prediction. This score helps understand the contribution of each word to the model's decision.
3. **Counterfactual explanations**: FRED offers counterfactual explanations by generating minimally perturbed samples of the original text that lead to a different prediction.

### 5.2.1 Drop in prediction

Key notations include $z$ as a generic document, $\xi$ as the specific document under consideration, $\mathcal{D}$ as the global dictionary with $D$ unique terms, $\xi = (\xi_1, \ldots, \xi_b)$ as a document composed of $b$ ordered words, $\mathcal{D}_\xi = w_1, \ldots, w_d \subseteq \mathcal{D}$ as the distinct words in $\xi$, with $d \leq b$, and $[k]$ representing the set of integers from 1 to $k$.

This chapter focuses on explaining the predictions of a generic model $F : \mathcal{T} \subseteq \mathcal{X} \to \mathcal{Y} \subseteq \mathbb{R}^p$, referred to as the *black-box*, which takes textual documents as input (as defined in Section 2.1). For classification problems, $F$ maps textual inputs to confidence scores for $p$ different classes, *i.e.*, $\mathcal{Y} = [0, 1]^p$. **The goal is to identify the optimal subset of tokens in the example $\xi$ whose removal significantly decreases the confidence score for the class $\ell \in \mathcal{C}$ of interest: $F_\ell(\xi)$.** As before, the prediction for the class $\ell \in \mathcal{C}$ of interest for any document is $f(z) = F_\ell(z)$.

Finally, a *candidate* explanation is defined as any non-empty ordered sublist of $[b]$, corresponding to tokens of $\xi$. The set of all candidates for $\xi$ is denoted as $\mathcal{C}$. The length of a candidate, $|c|$, is defined as the number of tokens (not necessarily distinct words) it contains.

To assess the impact of *removing* specific words on model predictions, the key concept of *drop in prediction* is introduced.

**Definition 5.2.1 (Drop in prediction).** For any sample $x \in \mathcal{X}$ and model $f$, the *drop in prediction* for is defined as

$$d(x) := \mathbb{E}\left[f(x)\right] - f(x)\,,$$

where $\mathbb{E}\left[f\right]$ is the expected prediction under a sampling distribution, and $x$ is a local perturbation of $\xi$ (detailed in Section 5.2.2).

The influence of a candidate explanation $c$ for a prediction $f(x)$ is subsequently computed using the *candidate drop*.

**Definition 5.2.2 (Candidate drop).** For any sample $x \in \mathcal{X}$ and model $f$, the *candidate drop* is defined as

$$\Delta_c := \mathbb{E}\left[f(x)\right] - \mathbb{E}\left[f(x) \mid c \notin x\right] = \mathbb{E}\left[d(x) \mid c \notin x\right] = \mathbb{E}_c\left[d(x)\right]\,, \tag{5.1}$$

which represents the expected drop in prediction when the candidate $c$ is removed from the original document $\xi$.

Using Eq. (5.1), the drop in prediction of samples where the candidate is perturbed is attributed to any candidate. The optimal candidate, denoted as $c^\star$, is determined by minimizing the size of the candidate subset while ensuring that it causes the average prediction $\mathbb{E}\left[f(x)\right]$ to drop by a significant amount. Formally, this is expressed by the optimization problem:

$$\underset{c \in \mathcal{C}}{\text{Minimize}}\, |c|\ ,\ \text{subject to}\ \ \Delta_c \geq \varepsilon \cdot \mathbb{E}\left[f(x)\right]\ . \tag{5.2}$$

**Empirical drop in prediction.** Calculating the prediction drop for each candidate in closed form, as formulated in Eq. (5.1), requires an exhaustive search and evaluation of $2^b$ candidates, which is impractical for large documents. To overcome this computational challenge, an empirical approach is employed to estimate the prediction drop.

For a given document $\xi$, a set of $n$ samples is generated through random perturbations of words in $\xi$ (detailed in Section 5.2.2). Consider any candidate $c$, and let $n_c$ represent the number of samples where $c$ is absent, defined as $n_c = \left|\{i \in [n] \mid c \notin x^{(i)}\}\right|$. The empirical candidate drop for $c$ is then defined as

$$\widehat{\Delta}_c := \frac{1}{n}\sum_{i=1}^{n} f(x^{(i)}) - \frac{1}{n_c}\sum_{c \notin x^{(i)}} f(x^{(i)}) = \widehat{f(x)} - \frac{1}{n_c}\sum_{c \notin x^{(i)}} f(x^{(i)})\,. \tag{5.3}$$

| | $\xi_1$ | $\xi_2$ | $\boldsymbol{\xi_3}$ | $\xi_4$ | $\boldsymbol{\xi_5}$ | $\xi_6$ | $d(x)$ |
|---|---|---|---|---|---|---|---|
| $\xi$ | Poor | drinks | **decent** | food | **great** | service | $d(\xi)$ |
| $x^{(1)}$ | great | drinks | decent | view | slow | service | $d(x^{(1)})$ |
| $x^{(2)}$ | Poor | seats | bad | boost | great | house | $d(x^{(2)})$ |
| $x^{(3)}$ | good | table | poor | food | awful | service | $d(x^{(3)})$ |
| $x^{(4)}$ | amazing | spot | decent | food | bad | tips | $d(x^{(4)})$ |
| $x^{(5)}$ | Poor | drinks | boring | walk | inept | service | $d(x^{(5)})$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $x^{(n)}$ | Poor | space | average | food | lousy | service | $d(x^{(n)})$ |

| | $\xi_1$ | $\xi_2$ | $\boldsymbol{\xi_3}$ | $\xi_4$ | $\boldsymbol{\xi_5}$ | $\xi_6$ | $d(x)$ |
|---|---|---|---|---|---|---|---|
| $\xi$ | Poor | drinks | **decent** | food | **great** | service | $d(\xi)$ |
| $x^{(1)}$ | UNK | drinks | decent | UNK | UNK | service | $d(x^{(1)})$ |
| $x^{(2)}$ | Poor | UNK | UNK | UNK | great | UNK | $d(x^{(2)})$ |
| $x^{(3)}$ | UNK | UNK | UNK | food | UNK | service | $d(x^{(3)})$ |
| $x^{(4)}$ | UNK | UNK | decent | food | UNK | UNK | $d(x^{(4)})$ |
| $x^{(5)}$ | Poor | drinks | UNK | UNK | UNK | service | $d(x^{(5)})$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $x^{(n)}$ | Poor | UNK | UNK | food | UNK | service | $d(x^{(n)})$ |

Figure 5.1 – **Illustration of FRED `pos-sampling` scheme** (left panel) **and `mask-sampling` scheme** (right panel) for computing the drop of a candidate. For a given example $\xi$, FRED generates $n$ perturbed samples $x^{(1)}, \ldots, x^{(n)}$ by independently perturbing tokens with probability $p(= 0.5)$. Each sample $i \in [n]$ is associated with the model's drop in prediction $d(x^{(i)})$. Finally, the empirical drop $\widehat{\Delta}_c$ of a candidate is computed by averaging the drops over the samples that do not contain $c$. In the example, the candidate consists of the words *decent* and *great*. The samples where both tokens are perturbed are highlighted in gray. The empirical drop associated with {*decent*, *great*} is therefore computed by averaging $d(x^{(3)})$, $d(x^{(5)})$, …, $d(x^{(n)})$.

The sampling scheme ensures that, *with high probability*, there is at least one sample in which each candidate is not present. This definition ensures that, for a large number of samples, the empirical drop is a good estimate of Eq. (5.1), as expressed by the following:

**Lemma 5.2.1 (Convergence of Empirical Drop $\widehat{\Delta}_c$).** *For a candidate explanation $c$, let $n_c$ represent the count of samples in $x$ where $c$ is not included. Then, as $n \to \infty$, the empirical drop in prediction $\widehat{\Delta}_c$ associated with the candidate $c$ converges in probability to*

$$\mathbb{E}\left[f(x)\right] - \frac{\mathbb{E}\left[f(x)\mathbb{1}_{c \notin x}\right]}{\mathbb{P}\left(c \notin x\right)}.$$

This result justifies using Eq. (5.1) in subsequent analysis instead of Eq. (5.3). Lemma 5.2.1 is proved in Section B.1.1 of the Appendix.

### 5.2.2 Sampling scheme

This section details the sampling scheme used to estimate the drop associated with a candidate (see Eq. (5.1)). The goal is to examine the behavior of the model $f$ in the local neighborhood of $\xi$, *i.e.*, when some tokens of $\xi$ are absent. To mitigate the out-of-distribution issue (Section 1.5.1), absent tokens are replaced by random words with the same *Part-of-Speech* (POS) tag [Ribeiro et al., 2018]. This ensures, for instance, that a verb is replaced by another verb and an adjective by another adjective. Additionally, to highlight the difference in prediction, if the absent token has a *polarity* (positive or negative), FRED replaces it with a word with the same POS tag but the opposite polarity.

Indeed, in the example of Figure 5.1, replacing the word "great" with the word "amazing" will likely not result in any significant change in prediction. Conversely, replacing it with "bad" or "awful" will highlight its impact. This approach is referred to as **`pos-sampling`**.

Specifically, given a corpus $\mathcal{T}$, FRED under the `pos-sampling` scheme creates two dictionaries: $\mathcal{D}^{+}_{pos}$ and $\mathcal{D}^{-}_{pos}$, which correspond to the sets of tokens with the same POS tag but with

*positive* and *negative* polarity, respectively (*neutral* words are included in both sets). For a given example $\xi$, FRED generates perturbed samples $x^{(1)}, \ldots, x^{(n)}$ as follows:

1. Create $n$ copies of $\xi$.
2. For each sample, select each token independently with probability $p$.
3. Associate the selected tokens with a pair (*pos*, *polarity*).
4. Replace the selected tokens with a randomly (uniformly) chosen word that has the same *pos* and the opposite *polarity* from $\mathcal{D}^{+}_{pos}$ and $\mathcal{D}^{-}_{pos}$, respectively.
5. Associate each sample $x^{(i)}$ with its drop in prediction $d(x^{(i)})$.

See Figure 5.1 for an illustration.

The sample size $n$ is chosen such that, for each candidate, there is a high probability that at least one sample does not contain it, *i.e.*, such that for each $c \in \mathcal{C}$, $n_c \geq 1$ with probability greater than $\alpha$ (see Lemma 5.2.2).

**Lemma 5.2.2** (**Choosing** $n$)**.** *If the sample size $n$ is greater than or equal to*

$$\frac{\log(1-\alpha)}{\log(1-1/2^{l^{\max}})},$$

*then, for any candidate $c$ with size less than or equal to $l^{\max}$, there exists at least one sample not containing it, with probability at least $\alpha$. Specifically, for any candidate $c$:*

$$n = \left\lceil \frac{\log(1-\alpha)}{\log(1-1/2^{l^{\max}})} \right\rceil \Longrightarrow \mathbb{P}\left(\exists i \in [n] : c \notin x^{(i)}\right) \geq \alpha\,.$$

For "high probability," $\alpha$ is set to $0.95$ by default, with a token perturbation probability $p = 0.5$. The maximum number of words used for an explanation, $l^{\max}$, is set to $10$, as it is not practical to use a high proportion of a text. According to Lemma 5.2.2 (proven in Appendix B.1.2), these choices imply a sample size of approximately $n \approx 3000$.

**Remark.** This sampling scheme requires the availability of a corpus $\mathcal{T}$ and can be computationally expensive, as it necessitates computing the POS-tag and polarity for each token. Additionally, its effectiveness may depend on the underlying model. Alternatively, following the default implementation of the official repositories for Anchor, LIME, and SHAP, the **mask-sampling** method is proposed: simply mask the tokens to be removed with a predefined token.

The `mask-sampling` scheme is often criticized for creating out-of-distribution samples [Hase et al., 2021], as also previously discussed in Section 1.5.1, by generating meaningless sentences. To address this issue, Anchors official implementation includes an option to sample by using BERT to predict the missing words. While this solution is potentially interesting, it may not meet practical expectations. First, replacing words with large language models is extremely resource-intensive: `BERT`-Anchor is ten times slower than `mask`-Anchor, even for small documents [Lopardo et al., 2023a], with no significant improvement in explanations. Second, examining the generated samples often reveals that the sentences remain meaningless. Note that Ribeiro et al. [2018] proposes replacing tokens with random words of the same part-of-speech tag, with a probability proportional to their similarity in the embedding space (however, this is not implemented in Anchors official repository). As previously mentioned, the opposite-sentiment approach is considered more suitable for our purposes, as it better captures the impact of the original token.

**Algorithm 2** An overview of FRED's algorithm. By default, $\varepsilon = 0.15$, $l^{\max} = 10$, $p = 0.5$, and the sample size $n$ is computed according to Lemma 5.2.2.

**Require:** model $F$, example $\xi$, threshold $\varepsilon$, max length $l^{\max}$, perturb probability $p$, sample size $n$

**initialize** $\widehat{\Delta}_{best} = 0$
$x = \text{generate\_sample}(\xi, p, n)$ *according to* `pos-sampling` *or* `mask-sampling`
$\widehat{F(x)} = \text{average}(F(x))$
$d(x) = \widehat{F(x)} - F(x)$ *difference between average and each sample prediction*
**for** $l = 1 : l^{\max}$ **do**:
  $\text{candidates} = \text{get\_candidates}(\xi, l)$ *return subsets with cardinality l*
  **for** $c \in \text{candidates}$ **do**:
    $\widehat{\Delta}_c = \text{compute\_drop}(F, c, x, d(x))$ *average $d(x)$ where $c \notin x$*
    **if** size = 1 **then**:
      $s_c = \widehat{\Delta}_c$ *per-token importance score*
    **end if**
    **if** $\widehat{\Delta}_c \geq \widehat{\Delta}_{best}$ **then**:
      $\widehat{\Delta}_{best} = \widehat{\Delta}_c$, best = $c$
    **end if**
  **end for**
  **if** $\widehat{\Delta}_{best} \geq \varepsilon \widehat{F(x)}$ **then**:
    **return** best
  **end if**
**end for**
**return** best

### 5.2.3 Explanations

Once the sample $x$ is created, the empirical drop associated with a candidate $\widehat{\Delta}_c$ is computed by averaging the drops over the samples that do not contain $c$. See Figure 5.1 for an illustration. The keywords, *i.e.*, **minimal influential subset** of tokens, is then identified using the empirical version of Eq. (5.2):

$$\underset{c \in \mathcal{C}}{\text{Minimize}} \, |c| \;, \text{ subject to } \; \widehat{\Delta}_c \geq \varepsilon \cdot \widehat{f(x)} \,. \tag{5.4}$$

Algorithm 2 illustrates FRED's mechanism, with its output shown in Figure 2.1.

Candidates of size 1 correspond to the tokens in the example. Therefore, FRED assigns an **importance score** $s$ to each token $j \in [b]$ in the example $\xi$ as

$$\forall j \in [b] \,, \quad s_j = \widehat{\Delta}_{\{\xi_j\}} \,. \tag{5.5}$$

In practice, the importance score $s_j$ is the average drop of samples where the $j^{\text{th}}$ token $\xi_j$ is perturbed. Figure 2.1 (b) shows the saliency maps for these scores.

Additionally, for classification tasks, FRED provides **counterfactual explanations** by identifying the samples in $x$ with minimal perturbation (*i.e.*, with the minimal number of perturbed tokens) that lead to a different classification. See Figure 2.1 (c) for an illustration.

## 5.3 Analysis on Explainable Classifiers

To rigorously assess FRED's effectiveness, a comprehensive theoretical analysis is performed. The framework established by Garreau and Luxburg [2020] for LIME, extended for text data by Mardaoui and Garreau [2021], Lopardo et al. [2023a], is leveraged, in the same vein as exposed in Chapter 4 for Anchors. Since there is no perfect way to verify explanation correctness (Section 1.5.2), the analysis ensures that FRED behaves as expected on simple, well-understood models. The investigation focuses on FRED's behavior with inherently interpretable classifiers. Firstly, its ability to identify the most critical token subset when applied to linear models is examined. Secondly, its capability in detecting shortcuts [Bastings et al., 2022] is evaluated.

For the theoretical analysis, two key assumptions about the models are made: as for Chapter 4. First, the focus is on binary sentiment analysis tasks, assuming, without loss of generality, that the example $\xi$ is classified as positive. In other words, the model $f$ outputs a confidence score for the positive class. Second, to simplify the analysis and obtain closed-form solutions, classifiers that operate on the well-established TF-IDF vectorization $\varphi$ of documents are considered (defined in Section 2.2). While not the most recent technique, TF-IDF's simplicity allows for the derivation of closed-form solutions in this analysis.

It is assumed that the mask token is not in the fitted corpus, thus, under the `mask`-sampling (Section 5.2.2), replacing any token with this mask is equivalent to simply removing it from the perspective of TF-IDF. As a consequence, a *candidate* explanation $c$ is any ordered sublist of words in $\mathcal{D}_\xi$, formally defined as $c = (c_1, c_2, \ldots, c_d)$, where $0 \leq c_j \leq m_j$ for all $j \in [d]$. The length of a candidate, denoted as $l$, is defined as $l = |c| = \sum_{j=1}^{d} c_j$.

### 5.3.1 Linear Classifiers

This section focuses on *linear classifiers*, defined by the model

$$f(z) = \lambda^\top \varphi(z) + \lambda_0 \,,$$

where $\lambda \in \mathbb{R}^D$ is a learned vector of coefficients and $\lambda_0 \in \mathbb{R}$ is an intercept term. This setup encompasses a broad range of models, including logistic models and perceptrons, making it a useful starting point for analysis. The decision boundary is determined by the hyperplane where the linear combination of the TF-IDF vector $\varphi(z)$, weighted by $\lambda$ and offset by the intercept $\lambda_0$, is greater than zero.

**Proposition 5.3.1 (Linear Models).** *Let $\lambda, \lambda_0$ be the coefficients associated with the linear classifier defined by Eq. (4.7). Assume $\lambda_1 v_1 > \lambda_2 v_2 > \cdots > \lambda_d v_d$. The solution to Eq. (5.2) is such that (i) words associated with a negative coefficient do not appear in the optimal candidate, and (ii) the algorithm starts including words with higher $\lambda_j v_j$ values until a threshold is met.*

In simpler terms, Proposition 5.3.1 reveals that for a linear classifier, FRED retains only words that positively impact the prediction. It prioritizes words with the highest $\lambda_j v_j$ values, gradually including them in the explanation until the desired confidence level is achieved. An illustration is shown in Figure 5.2. The proof of Proposition 5.3.1 is provided in Appendix B.1.3. Note that this behavior is analogous to Anchors (Section 4.4.3).

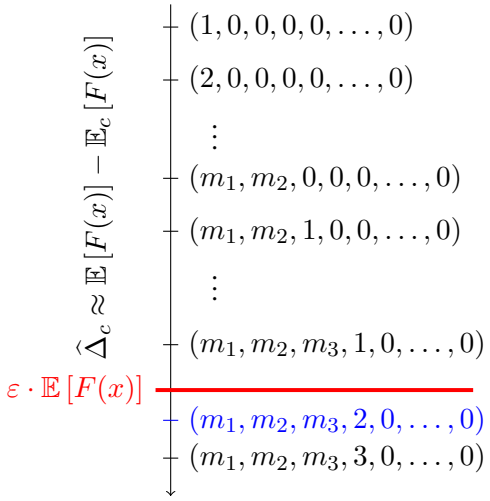


Figure 5.2 – **Illustration of Proposition 5.3.1.** In linear models, Algorithm 2 prioritizes words with the highest $\lambda_j v_j$ values first. The minimal candidate that satisfies the threshold condition is then selected. In this example, the selected candidate is $c = (m_1, m_2, m_3, 2, 0, \ldots, 0)$.

### 5.3.2 Shortcuts Detection

This section focuses on classifiers that rely on the presence or absence of specific tokens (shortcuts) in a document (analogously to Section 4.4.2). This approach is facilitated by the TF-IDF vectorizer, where the presence (or absence) of a word $w_i$ in a document $z$ is captured by the condition $\varphi(z)_i > 0$ (or $\varphi(z)_i = 0$). This characterization allows the exploration of classifiers whose predictions depend on specific terms occurring in the document.

**Proposition 5.3.2 (Presence of shortcuts).** *Assume $m_1 < m_2 < \ldots < m_k$, where $k$ is the maximum number of unique words in the document. Consider the set $J = \{1, 2, \ldots, k\}$ as a subset of $[d]$, and suppose the classifier $f$ is defined as*

$$f(z) = \mathbb{1}_{w_j \in z,\, \forall j \in J} = \prod_{j \in J} \mathbb{1}_{w_j \in z} = \prod_{j \in J} \mathbb{1}_{\varphi(z)_j > 0}.$$

*Then, the resulting optimal candidate $c^\star$ is characterized by $c_1^\star = \min\{m_1, l\}$ and $c_j^\star = 0$ for $j \geq 2$.*

In essence, if a model classifies a document based on the presence of specific words, removing all occurrences of just one of those words is sufficient to change the classification. FRED captures this information by identifying the minimal set of words whose removal would trigger this change.

Proposition 5.3.2 indicates that the smallest set of words necessary to trigger this shift in prediction consists of the word with the fewest occurrences in the document. This property aligns well with the definition of explanations in FRED. The proof of Proposition 5.3.2 is provided in Appendix B.1.4.

## 5.4 Experiments

FRED's explanations quality is assessed against three popular methods: LIME [Ribeiro et al., 2016], SHAP [Lundberg and Lee, 2017], and Anchors [Ribeiro et al., 2018]), with a focus on **faith-**

**fulness**—the adherence of the explanation to the model's behavior—and **robustness**. Evaluations are conducted on three sentiment analysis datasets (Restaurants, Yelp reviews, IMDb) and a hate speech detection dataset (Tweets), each with varying document lengths (details in Appendix B.2). Logistic regression, decision trees, and random forests were trained on each dataset. Additionally, pre-trained RoBERTa [Liu et al., 2019] and DistilBERT [Sanh et al., 2019] models were applied.

**Faithfulness.** Faithfulness is evaluated using **Comprehensiveness** and **Sufficiency** [DeYoung et al., 2020] to assess explainers' ability to identify crucial token subsets, and **AUC-MoRF** (Area Under the Most Relevant First Perturbation Curve) [Kakogeorgiou and Karantzalos, 2021] for importance-based explanations.

Given an explanation $E$ defined as a subset of tokens in the example $\xi$ to explain, **Comprehensiveness** is computed as the difference between the prediction confidence of the entire document $\xi$ and the prediction confidence when the explanation $E$ is *removed*. **Sufficiency** is computed as the difference between the prediction confidence of the entire document $\xi$ and the prediction confidence based only on the explanation $E$. Formally,

$$\text{Comprehensiveness} = f(\xi) - f(\xi \setminus E) \qquad \text{and} \qquad \text{Sufficiency} = f(\xi) - f(E)\,.$$

A high Comprehensiveness score implies that the subset $E$ was indeed influential in the prediction, while a low score suggests it had little impact. A low Sufficiency score implies that the subset $E$ does not sufficiently summarize the document, indicating more information is needed for an accurate prediction.

When applying feature-importance-based explainers, tokens in $\xi$ are ranked according to their score. **AUC-MoRF** is defined as

$$\text{AUCMoRF} = \frac{1}{K}\sum_{k=2}^{K}\frac{f(x_{(k-1)}) + f(x_{(k)})}{2}\,,$$

where $K$ is the maximum number of perturbations, and $x_{(k)}$ is the example $\xi$ after the $k^{\text{th}}$ MoRF perturbation, *i.e.*, after removing the $k$ most important (positive) tokens according to the explanation. A smaller AUC-MoRF value indicates a more faithful explanation. Only tokens with positive scores are perturbed for this computation. Let $E^+$ denote the list of tokens with positive influence according to the explainer. In the experiments, $K$ is set to $K = \min(20, |E^+|)$.

For the three faithfulness metrics, token removals are simulated by replacing the missing token with the mask `UNK`.

**Robustness.** Robustness of an explanation $E$ is computed as follows. First, the explainer is applied to obtain its explanation $E$ as the ground truth for the example. Then, $K$ additional iterations of the explainer are conducted on the same document, yielding $K$ new explanations, denoted as $E_1, E_2, \ldots, E_K$. The Jaccard Similarity between the original explanation $E$ and each $E_k$ is computed for $k \in [K]$. Robustness is then calculated as the average Jaccard Similarity score across the $K$ new explanations and the original explanation. Formally,

$$\forall k \in [K]\,,\ \mathrm{J}(E, E_k) = \frac{|E \cap E_k|}{|E \cup E_k|} \quad \text{and} \quad \text{Robustness} = \frac{\sum_{i=1}^{k}\mathrm{J}(E, E_k)}{K}\,.$$

TABLE 5.1 – Comparison on Roberta for Restaurant reviews ($p = 0.5$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.517(0.49) | 0.548(0.49) | 0.925(0.22) | 0.146(0.11) | 35.814(1.69) | 0.127(0.06) |
| fredpos | 0.517(0.49) | 0.542(0.49) | 0.920(0.23) | 0.198(0.16) | 35.249(1.57) | 0.127(0.06) |
| lime | 0.527(0.49) | 0.528(0.50) | 0.925(0.22) | 0.162(0.13) | 38.834(2.72) | 0.127(0.06) |
| shap | 0.766(0.42) | 0.323(0.46) | 1.000(0.00) | 0.306(0.21) | 1.899(1.74) | 0.127(0.06) |
| anchor | 0.509(0.49) | 0.538(0.50) | 0.850(0.32) | 0.454(0.45) | 15.491(20.40) | 0.126(0.06) |

TABLE 5.2 – Comparison on Random forest classifier for Yelp reviews ($p = 0.5$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | −0.142(0.11) | 0.114(0.04) | 0.804(0.21) | 0.759(0.13) | 0.418(0.18) | 0.117(0.14) |
| fredpos | −0.123(0.12) | 0.080(0.05) | 0.833(0.23) | 0.740(0.09) | 0.452(0.21) | 0.071(0.10) |
| lime | −0.126(0.11) | 0.083(0.04) | 0.933(0.16) | 0.782(0.08) | 0.332(0.10) | 0.061(0.08) |
| shap | −0.132(0.11) | 0.073(0.05) | 0.972(0.09) | 0.776(0.08) | 0.638(0.23) | 0.071(0.10) |
| anchor | −0.049(0.16) | 0.032(0.05) | 0.754(0.33) | 0.942(0.09) | 2.530(6.26) | 0.038(0.04) |

**Results.** In addition to the aforementioned metrics, the average computing time and the average proportion of the document used for the explanation (*i.e.*, $|E| \,/\, |\xi|$) are reported. Note that AUC-MoRF is independent of this proportion. FRED under the `pos-sampling` scheme is referred to as FRED-pos (`fredpos` in the tables), and FRED under the `mask-sampling` scheme is referred to as FRED-mask (`fred` in the tables).

The results compare various models across different datasets: Roberta for Restaurants reviews (Table 5.1, $k = 2$ for Robustness), Random forest classifier for Yelp reviews (Table 5.2, $k = 10$ for Robustness), DistilBERT and Roberta on IMDb (Tables 5.3 and 5.4, $k = 2$ for Robustness), Decision tree and random forest classifier for Tweets hate speech detection (Tables 5.5 and 5.6, $k = 10$ for Robustness).

Both FRED-mask and FRED-pos produce significantly more faithful explanations than other methods. FRED-mask slightly outperforms FRED-pos in terms of sufficiency and comprehensiveness but requires a larger number of tokens (higher proportion). In general, Anchors performs well on small documents (Restaurant dataset) but exhibits highly nonlinear behavior on larger documents, tending to be conservative with the anchor size. It also performs poorly on documents classified with high confidence, as demonstrated by Lopardo et al. [2023a]. LIME and SHAP exhibit similar behavior, with SHAP being significantly more efficient. On small documents, SHAP performs an exhaustive search over all possible token subsets, contributing to its high robustness. Additional details on the experimental setting and results are provided in Appendix B.2.

## 5.5 Conclusion

Building trustworthy AI necessitates interpretable machine learning, especially in critical domains. Existing explainers for text models, however, often grapple with complexity, lack formal grounding, and unreliable performance. Our proposed method, FRED, tackles these limitations by providing three key explanatory insights: 1) identifying the minimal set of crucial words, 2) assigning importance scores to each token, and 3) generating counterfactual explanations. We for-

TABLE 5.3 – Comparison on DistilBERT for IMDb ($p = 0.5$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | −0.002(0.01) | 0.020(0.01) | 0.602(0.15) | 0.970(0.01) | 9.406(0.87) | 0.387(0.13) |
| fredpos | 0.002(0.01) | 0.017(0.01) | 0.499(0.14) | 0.975(0.01) | 9.754(0.65) | 0.381(0.13) |
| lime | 0.001(0.01) | 0.019(0.01) | 0.897(0.10) | 0.972(0.01) | 11.052(1.31) | 0.292(0.15) |
| shap | 0.000(0.01) | 0.016(0.01) | 1.000(0.00) | 0.975(0.01) | 1.934(0.86) | 0.381(0.13) |
| anchor | 0.020(0.01) | 0.003(0.00) | 1.000(0.00) | 0.995(0.01) | 0.633(0.06) | 0.039(0.02) |

TABLE 5.4 – Comparison on Roberta for IMDb ($p = 0.5$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.201(0.39) | 0.249(0.43) | 0.864(0.23) | 0.204(0.17) | 47.849(4.14) | 0.069(0.05) |
| fredpos | 0.454(0.49) | 0.209(0.40) | 0.910(0.27) | 0.328(0.26) | 48.098(4.51) | 0.038(0.02) |
| lime | 0.348(0.46) | 0.219(0.41) | 0.810(0.37) | 0.295(0.33) | 66.732(8.62) | 0.038(0.02) |
| shap | 0.621(0.47) | 0.149(0.35) | 1.000(0.00) | 0.475(0.32) | 15.147(4.37) | 0.038(0.02) |
| anchor | 0.463(0.48) | 0.228(0.42) | 0.640(0.41) | 0.777(0.40) | 55.617(115.33) | 0.037(0.02) |

mally established FRED's reliability through theoretical analysis on interpretable models, while our empirical evaluation assess its effectiveness in surpassing current methods for explaining text predictions.

**Extending FRED to different data types.** FRED can naturally be extended to image classifiers and tabular data, making it a versatile tool for various types of machine learning models. The definitions in Section 5.2 remain applicable, and the analyses in Section 5.3 can be straightforwardly adapted to these different types of data. However, the primary aspect that needs to be addressed for this extension is the sampling process. When adapting the sampling mechanisms from other methods, such as those used in LIME (see for example Figure 1.17), FRED retains its innovative edge due to its original definition of *candidate drop* in Definition 5.2.2. This concept is key to FRED's ability to identify the minimal set of features that most significantly impact the prediction when removed.

In the case of images, typical approaches for perturbation include masking pixels (or groups of pixels) with random values, replacing them with the mean pixel value, or blurring them with black. While these methods are straightforward to implement, they introduce the out-of-distribution issue. Masked or altered images do not resemble the natural images the model was trained on, potentially leading to misleading explanations. Modern vision-language models, incorporating transformer architectures, present an alternative approach for generating perturbed samples. These models could be used to create more realistic and contextually appropriate modifications to the images. However, this approach is extremely resource-intensive. Generating a meaningful sample using a vision-language model requires significant computational power and time, which might not be feasible for all applications.

For tabular data, the challenges are equally significant. Perturbing features in tabular datasets often involves replacing values with mean or median values, random sampling from the feature distribution, or using domain-specific logic to generate plausible alternatives. Each of these

TABLE 5.5 – Comparison on a decision tree for Tweets ($p = 0.5$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.870(0.34) | 0.930(0.26) | 0.975(0.12) | 0.079(0.04) | 0.054(0.01) | 0.140(0.05) |
| fredpos | 0.880(0.32) | 0.930(0.26) | 0.954(0.14) | 0.078(0.04) | 0.065(0.01) | 0.140(0.05) |
| lime | 0.870(0.34) | 0.920(0.27) | 0.881(0.22) | 0.082(0.06) | 0.129(0.01) | 0.140(0.05) |
| shap | 0.880(0.32) | 0.910(0.29) | 1.000(0.00) | 0.088(0.08) | 0.034(0.16) | 0.140(0.05) |
| anchor | 0.890(0.31) | 0.800(0.40) | 0.650(0.39) | 0.109(0.14) | 0.582(0.48) | 0.138(0.05) |

TABLE 5.6 – Comparison on random forest classifier for Tweets ($p = 0.5$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.781(0.21) | 0.295(0.19) | 0.892(0.21) | 0.156(0.05) | 0.512(0.04) | 0.108(0.03) |
| fredpos | 0.784(0.20) | 0.386(0.15) | 0.958(0.15) | 0.149(0.05) | 0.379(0.05) | 0.108(0.03) |
| lime | 0.784(0.20) | 0.384(0.15) | 0.974(0.14) | 0.159(0.05) | 0.452(0.05) | 0.108(0.03) |
| shap | 0.784(0.20) | 0.383(0.15) | 1.000(0.00) | 0.156(0.05) | 0.155(0.15) | 0.108(0.03) |
| anchor | 0.788(0.20) | 0.281(0.19) | 0.493(0.40) | 0.221(0.10) | 8.019(4.09) | 0.108(0.03) |

approaches comes with its own set of challenges, particularly regarding the preservation of the feature distribution and the relationships between features.

In both cases, whether dealing with image or tabular data, the key to effective perturbation lies in generating samples that remain within (or *close to*) the data distribution the model was trained on. This ensures that the explanations provided by FRED are reliable and truly reflective of the model's decision-making process.

# CHAPTER 6

# Attention Meets Post-hoc Interpretability

Attention-based architectures, in particular transformers [Vaswani et al., 2017], are at the heart of a technological revolution (Section 1.2). Interestingly, in addition to helping obtain state-of-the-art results on a wide range of applications, the attention mechanism [Bahdanau et al., 2015] intrinsically provides meaningful insights on the internal behavior of the model. Can these insights be used as explanations? Debate rages on (Section 6.2). This chapter mathematically study a simple attention-based architecture (introduced in Section 6.3) and pinpoint the differences between attention-based explanations (Section 6.3.2) and popular post-hoc approaches, namely gradient-based (Section 6.5) and perturbation-based (Section 6.6). Lopardo et al. [2024] show that they provide quite different results, and that, despite their limitations, post-hoc methods are capable of capturing more useful insights than merely examining the attention weights.

## 6.1 Introduction

The attention mechanism, introduced by Bahdanau et al. [2015], revolutionized neural networks by enabling models to dynamically focus on different parts of input sequences, enhancing their ability to capture long-range dependencies. This innovation laid the groundwork for various deep learning models. The most notable application is the Transformer architecture, introduced by Vaswani et al. [2017], which eliminated the need for recurrent neural networks and convolutional layers, relying solely on attention mechanisms. The Transformer has since become the state-of-the-art in numerous machine learning domains due to its flexibility, performance, and ability to model complex relationships in data (see Section 1.2). Its innovative design and significant improvements in training efficiency have paved the way for the development of advanced models such as BERT [Devlin et al., 2019] and GPT-3 [Brown et al., 2020], revolutionizing NLP.

As a by-product of the attention mechanism, per-token attention weights at a given layer can be easily extracted from the model. These weights are often used as explanations for model predictions, and many researchers have indeed adopted this approach [Chefer et al., 2021, Mylonas et al., 2023]. However, the use of attention mechanisms for explainability has been questioned. Jain and Wallace [2019] critique its clarity, questioning the relationship between attention weights and model output. Conversely, Wiegreffe and Pinter [2019] argue that attention mechanisms remain useful for interpretability, without specifically addressing Jain and Wallace [2019]'s requirements. This debate, which elaborated in Section 6.2, highlights the need for a deeper theoretical foundation for attention-based explanations.

This chapter illustrates a mathematical analysis of attention-based models, first proposed in Lopardo et al. [2024], and their associated explanations, aiming to clarify the merits of each approach beyond experimental validation. The analysis centers on a single-layer multi-head network, detailed in Section 6.3, a simplified variant of the transformer architecture tailored for a binary prediction task. The binary classification restriction is illustrative; the same results hold for multi-label predictions when examining a specific class of interest. While focusing on text classification tasks, the analysis of token-level explanations could also apply to pixels in the context of Vision Transformers.

Specifically, the analysis examines the connections between attention-based explanations and established post-hoc explanations, including gradient-based methods such as *Gradient* [Li et al., 2016], *Gradient*$\times$*Input* [Denil et al., 2014], and perturbation-based approaches like LIME [Ribeiro et al., 2016]. The findings demonstrate that perturbation-based and gradient-based methods provide more insightful explanations than solely examining attention weights in Transformer models. This aligns with Bastings and Filippova [2020], who argue that attention weights, while useful for input token weighting, can be misleading as explanations for model predictions, advocating instead for post-hoc approaches.

**Contributions.** This chapter presents a detailed analysis of attention-based models and their explanations, with a specific focus on addressing the ongoing debate in the literature. The key contributions are as follows:

— Section 6.2 discusses the relevant literature, particularly the debate surrounding attention-based explanations.
— The model under study is described in Section 6.3.
— Attention-based explanations are specifically addressed in Section 6.4.

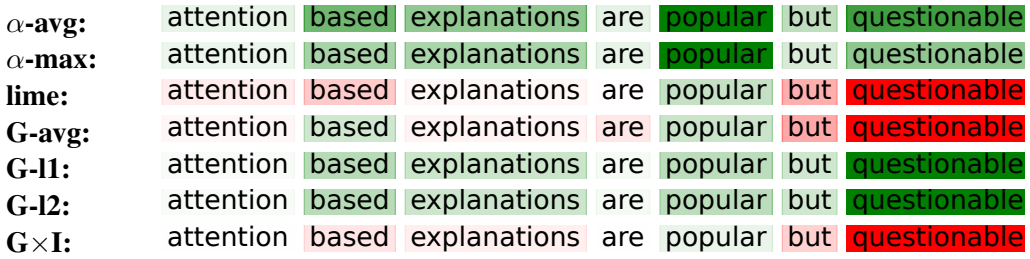


Figure 6.1 – **Different explainers can produce very different explanations.** Here, the *attention* mean ($\alpha$**-avg**) and maximum ($\alpha$**-max**) over the heads, *LIME* (**lime**), the *gradient* mean (**G-avg**), $L^1$ norm (**G-l1**), and $L^2$ norm (**G-l2**), with respect to the tokens, and *Gradient times Input* (**G**$\times$**I**) are employed to interpret the prediction of a sentiment-analysis model. Words with positive (respectively, negative) weights are highlighted in green (respectively, red), with intensity proportional to their weight. In the example, all the explainers identify the word *questionable* as highly significant, while only lime, and G$\times$I highlight a negative contribution. Interestingly, $\alpha$-avg and $\alpha$-max identify the word *popular* as the most important word in absolute terms, in disagreement with the all others.

— Sections 6.5 and 6.6 derive explicit expressions for gradient-based and LIME explanations, respectively, associated with the model. These expressions (Theorems 6.5.1 and 6.6.1) are explicit with respect to the model parameters and the input document, allowing for a precise comparison of these approaches.
— Section 6.7 discusses the main limitations of the work, including the theoretical assumptions underlying the model.
— Conclusions are drawn in Section 6.8.

All theoretical claims are supported by mathematical proofs and empirical validation, detailed in the Appendix. The code for the model and experiments is available at `https://github.com/gianluigilopardo/attention_meets_xai`.

## 6.2 Related Work

The attention mechanism, pioneered by Bahdanau et al. [2015], significantly enhanced neural networks' ability to focus on different parts of input sequences. Various forms of attention mechanisms exist, each characterized by distinct methods of query generation and computation of attention weights. Two primary methods are additive attention, as originally proposed by Bahdanau et al. [2015], and **scaled dot-product attention**, introduced by Vaswani et al. [2017], which is the focus of this study. Despite their differences, these forms are theoretically similar [Vaswani et al., 2017] and yield comparable results [Jain and Wallace, 2019]. This innovation paved the way for numerous deep learning models, notably the Transformer architecture by Vaswani et al. [2017].

Self-attention quantifies the relationship between each token in a sequence and every other token, represented as attention weights. These weights indicate the model's focus on different parts of the input, making it tempting to use them as explanations for the model's predictions. They offer an intuitive way to understand what the model is *paying attention to* when making

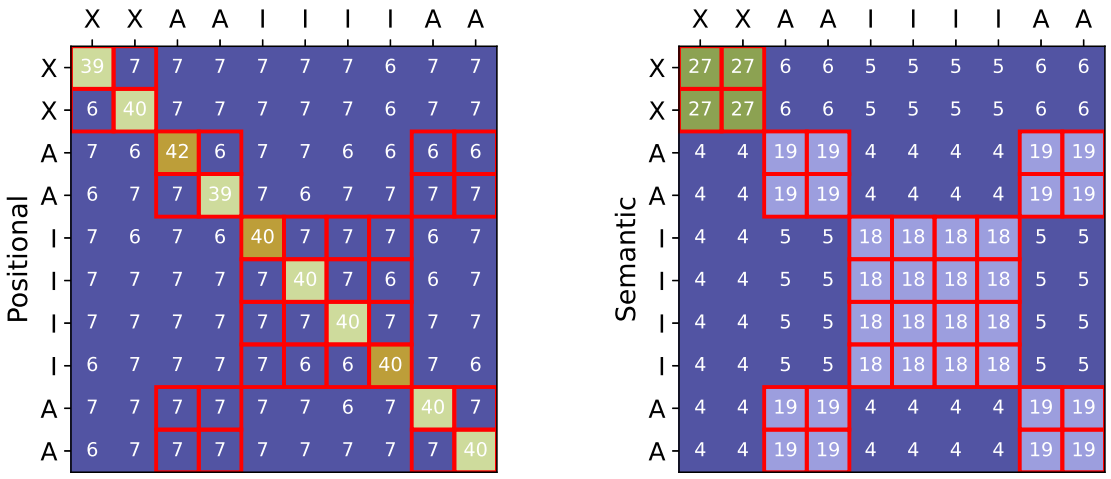


Figure 6.2 – **Attention matrices for the histogram task show two distinct solutions corresponding to different local minima in the loss landscape** [Cui et al., 2024]. The left panel depicts the positional solution, where the attention matrix is largely invariant to the specific input sequence, indicating a focus on positional information. The right panel shows the semantic solution, where attention values vary significantly based on the tokens at each position, emphasizing semantic relationships. Red squares highlight elements where the tokens at positions $i$ and $j$ are identical. These findings imply that the same predict can yield two extremely different attention matrices for the same prediction on the same task, leading to potentially misleading attention-based explanations.

decisions. Indeed, several methods generate attention-based explanations, which are discussed in Section 6.4.

While attention weights provide valuable insights into the model's behavior, their use for explainability has been met with skepticism in the literature, sparking an ongoing debate. This debate, summarized below, addresses whether attention mechanisms genuinely enhance interpretability or if they merely offer a superficial understanding.

### 6.2.1 The debate

Jain and Wallace [2019] offer a significant critique of the relationship between attention weights and model output. They argue, based on experiments across various NLP tasks, that attention weights do not provide meaningful explanations. Specifically, Jain and Wallace [2019] propose two properties that should hold if attention provides faithful explanations: (i) attention weights should correlate with feature importance measures (*e.g.*, gradient-based measures and leave-one-out), and (ii) alternative (or counterfactual) attention weight configurations should yield corresponding changes in prediction. Their experiments suggest that these properties do not hold, leading to the conclusion that attention weights are not suitable for interpretability.

However, Wiegreffe and Pinter [2019] highlight several limitations of Jain and Wallace [2019]'s work. Experimentally, Wiegreffe and Pinter [2019] conclude that "prior work does not disprove the usefulness of the attention mechanism for interpretability." They critique the experimental design proposed for point (ii) while somewhat agreeing with the first observation and its corresponding experimental setup. Specifically, Wiegreffe and Pinter [2019] introduce an end-

to-end model training approach for finding adversarial attention weights, ensuring that the new, adversarial weights are plausible and consistent with the model, contrasting with Jain and Wallace [2019]'s approach of changing only the attention scores, which disrupts the model's training. Furthermore, Wiegreffe and Pinter [2019] argue against the exclusive explanation: "attention is an explanation, not the explanation."

Serrano and Smith [2019] also scrutinize the use of attention for interpretability by manipulating attention weights in pre-trained text classification models and analyzing the impact on predictions. They conclude that attention provides a noisy prediction of the input tokens overall importance to a model but is not a reliable indicator.

More recently, Bibal et al. [2022] provide an overview of the debate on whether attention serves as an explanation, focusing on literature building on Jain and Wallace [2019] and Wiegreffe and Pinter [2019]. Bibal et al. [2022] argue that the applicability of attention as an explanation heavily depends on the specific NLP task. For instance, Clark et al. [2019] demonstrate that BERT's attention mechanism can provide reliable explanations for syntax-related tasks like part-of-speech tagging. Similarly, Vig and Belinkov [2019] present comparable results for GPT-2, showing that syntactic knowledge appears to be encoded across various attention heads and layers. Galassi et al. [2020] further show that attention in transformers focuses on syntactic structures, making it suitable for global explanations.

Brunner et al. [2020] theoretically demonstrate that attention weights can be decomposed into two parts, with the *effective attention* part focusing on the effective input without being biased by its representation. This work is expanded by Kobayashi et al. [2020] and Sun and Marasović [2021], who conduct more in-depth evaluations and find that alternative attention distributions obtained through adversarial training perform poorly, suggesting that the attention mechanism of RNNs indeed learns something useful. This finding contradicts Jain and Wallace [2019]'s claim that attention weights do not provide meaningful explanations.

Currently, there is no definitive theoretical support for either side of the debate on whether attention serves as an explanation. Both Jain and Wallace [2019] and Wiegreffe and Pinter [2019] base their positions primarily on empirical experiments. The subsequent debate has provided valuable insights and provoked thoughtful discussion but has not conclusively proven or disproven the interpretability of attention mechanisms.

Recent works have investigated the role of attention through mathematical examination on specific tasks, similar to our approach. Wen et al. [2024] examine transformer interpretability by analyzing the model's weight matrices and attention patterns in the context of learning a *Dyck language* [Schützenberger, 1963]. The authors demonstrate that vastly different solutions can be reached via standard training, cautioning against making interpretability claims based on inspecting individual components of the model. In particular, the attention pattern of a single layer can be "nearly randomized" and still achieve high accuracy. Similarly, Li et al. [2023] provide a mechanistic understanding of how transformers learn semantic structure through mathematical analysis and experiments on Wikipedia and LDA-generated [Blei et al., 2003] data, showing that both the embedding and self-attention layers can encode topical structures. Even when the attention score is set to be uniform, the transformer can achieve a near-optimal loss, as other parts of the model compensate. Finally, Cui et al. [2024] demonstrate that for a simple counting task (the *histogram task* defined in Weiss et al. [2021]), the loss landscape of a transformer with a dot-product attention layer and positional encodings reveals two distinct solutions: one with an attention matrix largely independent of the input tokens, and another that varies significantly based on the tokens and their semantic content. This results is illustrated in Figure 6.2. Ultimately, these works demonstrate

that there is no evidence to assume that attention scores capture the core information underlying a transformer's predictions.

### 6.2.2 Attention meets post-hoc interpretability

Attention-based explanations are post-hoc since they can be generated after training, but they are not model-agnostic as they require specific internal parameters (attention weights) of the model. Existing literature explores the differences between popular attention-based and other post-hoc explanations, employing diverse methodologies and drawing varied conclusions. Ethayarajh and Jurafsky [2021] formally establish that attention weights are not Shapley values [Shapley, 1953], but attention flows (a post-processed version of attention weights) are, at least at the layerwise level. Thorne et al. [2019] conduct a comparative analysis between post-hoc and attention-based methods. They select key features according to each explainer, subsequently using these features to make predictions and evaluate their accuracy. Their findings indicate that post-hoc methods like LIME and Anchors yield more accurate explanations than attention-based ones when implemented on an LSTM [Sak et al., 2014] for natural language inference.

Neely et al. [2021] evaluate the "agreement as evaluation" paradigm, comparing various explanation methods on Bi-LSTM [Huang et al., 2015] and Distil-BERT [Sanh et al., 2019] models. They conclude that consistency between different explainers should not be a criterion for evaluation unless a proper ground truth is available, contradicting the agreement between Jain and Wallace [2019] and Wiegreffe and Pinter [2019]. They also highlight the theoretical limitations of state-of-the-art explainers and suggest using robust diagnostic tools like those proposed by Atanasova et al. [2020].

Building on Neely et al. [2021]'s work, Neely et al. [2022] find a lack of correlation among explanation methods, particularly in complex settings. They question the existence of an *ideal* explanation and challenge the *agreement as evaluation* paradigm for comparison by demonstrating that similar explanations may not yield correlated rankings.

## 6.3 Attention-based classifier

In this section, the attention-based architecture studied in Lopardo et al. [2024] is introduced. The presentation and notation follow Phuong and Hutter [2022].

### 6.3.1 General Description

This paper considers a set of tokens belonging to a dictionary identified with $[D]$. A document $\xi$ is an ordered sequence of tokens $\xi_1, \ldots, \xi_T$, where $T$ denotes the length of the document. Without loss of generality, it is assumed that the $d$ unique tokens of $\xi$ are the first $d$ elements of $[D]$.

The model $f$ is a single-layer, multi-head, attention-based network followed by a linear layer. More formally:

$$f(x) := \frac{1}{K}\sum_{i=1}^{K} f_i(x) = \frac{1}{K}\sum_{i=1}^{K} W_\ell^{(i)}\tilde{v}^{(i)}(x)\,, \tag{6.1}$$

where $f_i := W_\ell^{(i)}\tilde{v}^{(i)} \in \mathbb{R}^{d_{\text{out}}}$, with $W_\ell^{(i)} \in \mathbb{R}^{1\times d_{\text{out}}}$ being the part of the final linear layer associated with head $i$, and for $i \in [K]$, $\tilde{v}^{(i)}(x)$ is the output of an individual head defined by Eq. (6.9).

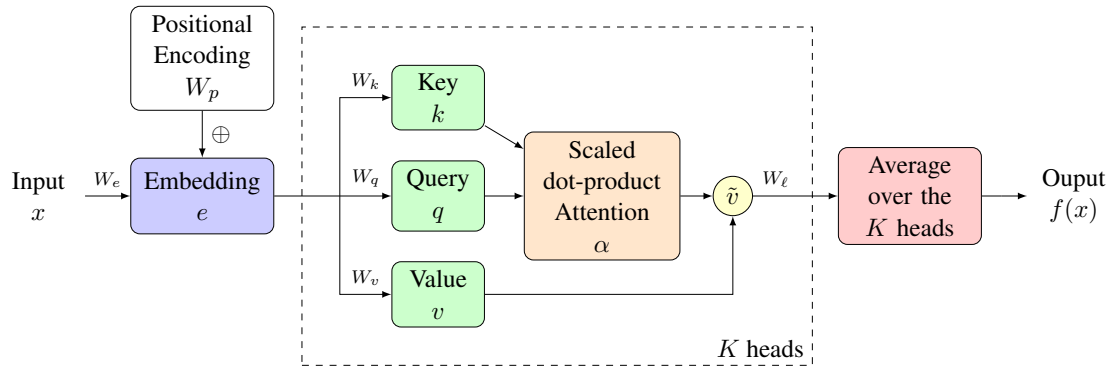


Figure 6.3 – **Illustration of the architecture of the model defined in Section 6.3.** The input text, denoted as $x \in [D]^T$, is transformed into an embedding $e \in \mathbb{R}^{T \times d_e}$ by summing word embeddings and positional encodings as in Eq. (6.2). For each of the $K$ heads, the key $k \in \mathbb{R}^{T \times d_{\text{att}}}$, query $q \in \mathbb{R}^{T \times d_{\text{att}}}$, and value $v \in \mathbb{R}^{T \times d_{\text{out}}}$ matrices are computed by applying linear transformations to $e$ using $W_k, W_q \in \mathbb{R}^{d_{\text{att}} \times d_e}$, and $W_v \in \mathbb{R}^{d_{\text{out}} \times d_e}$, respectively. The attention weights $\alpha \in \mathbb{R}^T$ are then computed as the softmax of the scaled dot-product of $k$ and $q$, as per Eq. (6.8). Then the intermediary output $\tilde{v} \in \mathbb{R}^{d_{\text{out}}}$ is computed are the average of the values $v$ weighted by the attention $\alpha$. Each head outputs the linear transformation $W_\ell \in \mathbb{R}^{1 \times d_{\text{out}}}$ of the $\tilde{v}$ associated with the query corresponding to the `[CLS]` token. The final prediction $f(x)$ of the model is the average of the outputs across all heads.

The value of $f$ is used for classification; for instance, in the sentiment analysis task described in the introduction, document $\xi$ is classified as positive if $f(\xi) > 0$.

### 6.3.2 The attention mechanism

The self-attention mechanism within each head is now described mathematically. Formally, $f_i$ is detailed for a given $i$, temporarily dropping the $i$ index for simplicity.

**Token Embedding.** For each $t \in [T]$, the token $\xi_t = j$ is embedded as

$$e_t := (W_e)_{:,j} + W_p(t) \in \mathbb{R}^{d_e}\,, \tag{6.2}$$

where $W_e \in \mathbb{R}^{d_e \times D}$ is a matrix containing the embedding of all tokens, and $W_p : \mathbb{Z} \to \mathbb{R}^{d_e}$ is a deterministic mapping often called the *positional embedding*. It is common to set

$$\begin{cases} W_p(t)_{2i} & = \cos(t/T_{\max}^{2i/d_e}) \\ W_p(t)_{2i-1} & = \sin(t/T_{\max}^{2i/d_e})\,, \end{cases} \tag{6.3}$$

with $T_{\max}$ being the maximal document size. For all $T < t \leq T_{\max}$, the embedding of the fictitious token value is set to an arbitrary $h \in \mathbb{R}^{d_e}$, while the positional embedding remains the same. In other words,

$$\forall T < t \leq T_{\max}, \quad e_t := h + W_p(t) \in \mathbb{R}^{d_e}\,. \tag{6.4}$$

If $T > T_{\max}$, the last tokens are ignored and the input document is effectively discarded. It is assumed that the embedding matrices are *shared* between the $K$ heads, although the analysis can easily accommodate different embedding matrices for each individual head.

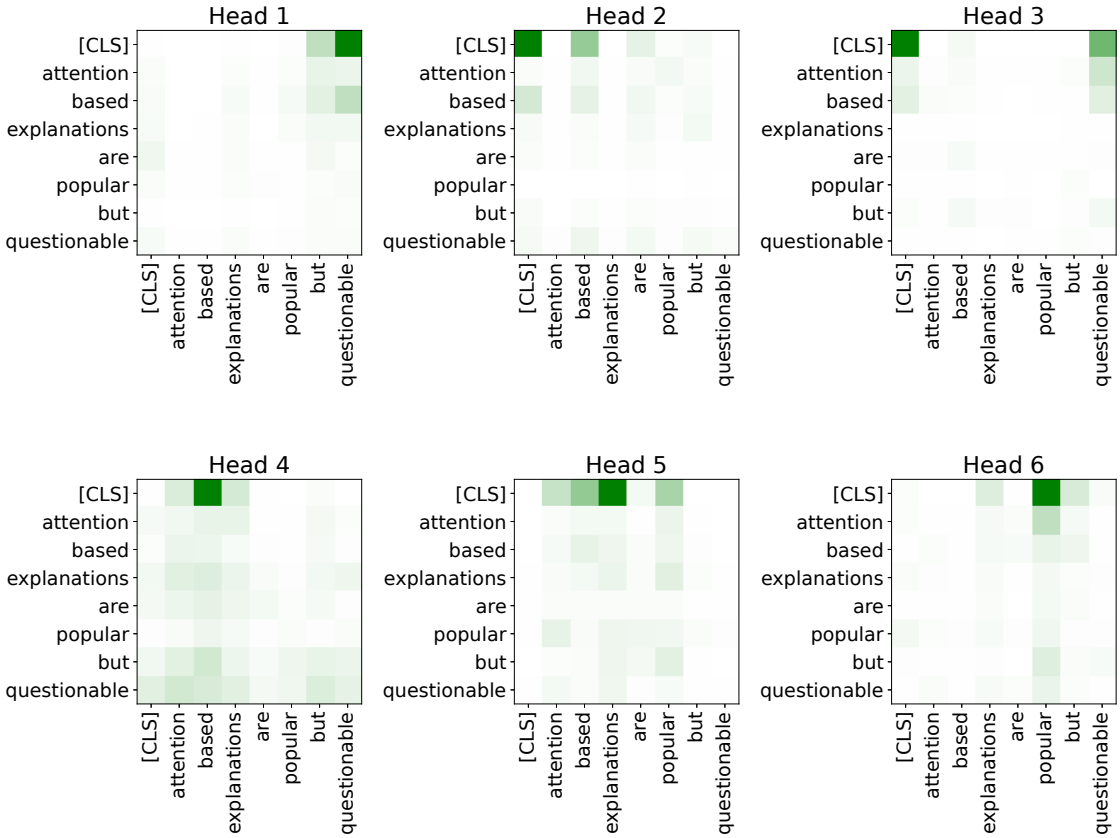


Figure 6.4 – **Attention matrices across the heads.** Each head is represented by a distinct matrix, demonstrating the unique focus each head has on different parts of the document. The matrices illustrate that tokens within the document can carry significantly different weights, indicating the varying importance or relevance of each token in the context of the document. The aggregation of these weights to provide token-level scores is a critical aspect. Note that Eqs. (6.10) and (6.11) correspond to the average and the maximum values, respectively, of the first row across all six matrices.

**Keys, Queries, Values.** Next, these embeddings are mapped to *key*, *query*, and *value* vectors, defined respectively as

$$k_t := W_k e_t + b_k \in \mathbb{R}^{d_{\text{att}}}, \tag{6.5}$$

$$q_t := W_q e_t + b_q \in \mathbb{R}^{d_{\text{att}}}, \tag{6.6}$$

$$v_t := W_v e_t + b_v \in \mathbb{R}^{d_{\text{out}}}, \tag{6.7}$$

with $W_k, W_q \in \mathbb{R}^{d_{\text{att}} \times d_e}$, and $W_v \in \mathbb{R}^{d_{\text{out}} \times d_e}$. For simplicity, the bias vectors $b_k, b_q \in \mathbb{R}^{d_{\text{att}}}$ and $b_v \in \mathbb{R}^{d_{\text{out}}}$ are considered to be zero.

**Attention.** For a given query $q \in \mathbb{R}^{d_{\text{att}}}$, the attention $\alpha_t$ received by each index $t$ is defined as

$$\alpha_t := \frac{\exp\left(q^\top k_t / \sqrt{d_{\text{att}}}\right)}{\sum_{u=1}^{T_{\max}} \exp\left(q^\top k_u / \sqrt{d_{\text{att}}}\right)} . \tag{6.8}$$

The scaling factor $1/\sqrt{d_{\text{att}}}$, although not strictly necessary (since $W_q$ and $W_k$ are learnable parameters of the model), is retained to properly scale the positional embedding.

**Output of the Model.** The intermediary output value before the final linear transformation associated with the query $q$ is

$$\tilde{v} := \sum_{t=1}^{T_{\max}} \alpha_t v_t \in \mathbb{R}^{d_{\text{out}}} . \tag{6.9}$$

Each individual head transforms the $\tilde{v}$ associated with the query corresponding to the `[CLS]` token. Specifically, for $i \in [K]$, $f_i(x) = W_\ell^{(i)} \tilde{v}^{(i)}$.

Limitations and differences between this model and practical architectures are discussed in Section 6.7.

## 6.4 Attention-based Explanations

The **scaled dot-product attention**, introduced by Vaswani et al. [2017] (corresponding to the definition in Eq. (6.8)), measures the relationship among tokens. This mechanism generates a matrix where each entry represents the degree of association between a pair of tokens. Specifically, any attention head $i \in [K]$ produces a $T \times T$ attention matrix $A^{(i)}$ (illustrated in Figure 6.4), where each entry $A^{(i)}_{s,t} = \alpha_t(q_s)$ is the attention value defined in Eq. (6.8), computed with respect to the $s$-th query token.

Furthermore, as this study focuses on a classification model, only the `[CLS]` token, which encapsulates the core of the classification [Chefer et al., 2021], is considered. Formally, this means that only the specific query $q$ linked to the `[CLS]` token is relevant in Eq. (6.8). This is equivalent to selecting the first row of the attention matrices in Figure 6.4.

Note that, in general, a Transformer model is structured as a series of sequential layers, each equipped with a specific number of parallel heads. These heads operate independently, executing the attention mechanism. To produce token-level attention-based explanations, one must aggregate the attention matrices at both the head and layer levels. Mylonas et al. [2023] provide a detailed depiction of these operations; refer to Figure 2 in Mylonas et al. [2023] for a comprehensive illustration.

In our scenario, the model defined in Section 6.3 is single-layered, hence layer-level aggregation is omitted.

As a result, each head produces an attention vector of size $T$ that highlights the focus of the head on each token. However, as depicted in Figure 6.4, heads often concentrate on different sections of the document. Thus, aggregating the $K$ attention vectors is crucial. The two most common aggregation methods involve computing the average vector or determining the maximum value among the vectors for each token. Formally, for any token $t \in [T]$, we define:

$$\alpha\text{-avg}_t := \frac{1}{K} \sum_{i=1}^{K} \alpha_t^{(i)} , \tag{6.10}$$

and

$$\alpha\text{-max}_t := \max_{i \in [K]} \alpha_t^{(i)} \,. \tag{6.11}$$

It is important to note that $\alpha$-avg and $\alpha$-max can lead to very different explanations. Additionally, $\alpha$-avg, $\alpha$-max, and G-l1 generate non-negative weights. Consequently, these methods do not differentiate between words that contribute positively or negatively to the prediction, as depicted in Figure 6.1.

## 6.5 Gradient-based Explanations

This section delves into gradient-based explanations, beginning with an overview of the main methods before computing these explanations for the model proposed in Section 6.3.

### 6.5.1 Methods

This section describes existing gradient-based methods, often referred to as saliency maps, drawing an analogy to a similar technique in computer vision. Given a model $f$ and an instance $x$, the gradient with respect to a token $t \in [T]$ is defined as:

$$\nabla_{e_t} f(x) \in \mathbb{R}^{d_e} \,. \tag{6.12}$$

It is important to note that the gradient $\nabla_{e_t}$ is calculated with respect to the embedding vector $e_t \in \mathbb{R}^{d_e}$. To derive per-token importance weights, several strategies exist. The primary approaches, known as *Gradient* explanations, involve taking the mean value (G-avg) [Atanasova et al., 2020], the $L^1$ norm (G-l1) [Li et al., 2016], or the $L^2$ norm (G-l2) [Poerner et al., 2018, Arras et al., 2019, Atanasova et al., 2020] of the components of Eq. (6.12).

An alternative approach, known as *Gradient times Input* (G×I) [Denil et al., 2014], suggests computing saliency weights by performing the dot product of the gradient from Eq. (6.12) with the input word embedding $e_t$. In this notation, the saliency weights are calculated as $e_t^\top (\nabla_{e_t} f(x))$.

While these methods share a common foundation, the explanations they generate can vary significantly and may even be contradictory. As illustrated in Figure 6.1, the G-l1 and G-l2 methods yield non-negative weights. In other words, these methods do not distinguish between words that contribute positively or negatively to the prediction, contrary to G×I (see Figure 6.1).

### 6.5.2 Gradient of the model

Consider the model described in Section 6.3. The function $f$ is linear with respect to the $f_i$ head, $i \in [K]$, hence, the gradient of $f$ with respect to the token embedding $e_t$ is:

$$\nabla_{e_t} f(x) := \frac{1}{K} \sum_{i=1}^{K} \nabla_{e_t} f_i(x) \in \mathbb{R}^{d_e} \,. \tag{6.13}$$

The primary quantity of interest is the gradient of a single attention head, $\nabla f_i(x)$. Recall that $q$ is the query corresponding to the classification token `[CLS]`. With this notation, the following theorem can be stated (proved in Appendix C.1).

**Theorem 6.5.1 (Gradient Meets Attention).** *The gradient of the model $f$, defined by Eq. (6.1), with respect to the embedded token $e_t$, $t \in [T]$, is:*

$$\nabla_{e_t} f(x) = \frac{1}{K} \sum_{i=1}^{K} \left[ \alpha_t^{(i)} (W_v^{(i)})^\top (W_\ell^{(i)})^\top + \frac{\alpha_t^{(i)}}{\sqrt{d_{att}}} W_\ell^{(i)} \left( v_t^{(i)} - \sum_{s=1}^{T_{\max}} \alpha_s^{(i)} v_s^{(i)} \right) (W_k^{(i)})^\top q \right] \in \mathbb{R}^{d_e} . \tag{6.14}$$

By substituting Eq. (6.14) into Eq. (6.13), the gradient-based explanations discussed in Section 6.5.1 can be reconstructed. Specifically, G-avg, G-l1, and G-l2 are the average, the $L^1$, and the $L^2$ norm of $\nabla_{e_t} f(x) \in \mathbb{R}^{d_e}$, respectively, while G×I is the dot product between the gradient and the embedding vector: $e_t^\top (\nabla_{e_t} f(x))$.

Several key points are noteworthy. (i) The gradient of $f$, at the first-order approximation, is **linear in** $\alpha$, which partly explains its correlation with the attention weights. Consequently, **$\alpha$-avg correlates with G-avg and G-l1**, aligning with desiderata (i) of Jain and Wallace [2019], while the same does not hold for $\alpha$-max and G-l2. However, this correlation may not necessarily hold true for deeper models.

(ii) **The gradient captures the influence of the linear layers** $W_\ell^{(i)}$, providing an insight disregarded by attention-based explanations. For instance, assuming $K = 1$ and $v_t \approx \sum_{s=1}^{T_{\max}} \alpha_s v_s$ (reflecting a situation where the value vector is "typical"), the only remaining part in Eq. (6.14) is $\alpha_t W_v^\top W_\ell^\top$. A positive $\alpha_t$ can yield **negative** explanations if the coordinates of $W_v^\top W_\ell^\top$ reflect the true behavior of the model.

## 6.6 Perturbation-based Explanations

This section focuses on perturbation-based explanations, specifically using LIME for text data. The following subsections will provide an overview of how LIME operates, introduce necessary additional notation, and present the main result.

### 6.6.1 Reminder on LIME

LIME for text data, as detailed in Mardaoui and Garreau [2021], operates by starting with the document $\xi$ to be explained and generating local perturbations $X_1, \ldots, X_n$. A local linear model is then trained on these perturbations to approximate the predictions of $f$. The linear coefficients of this model are provided as the explanation to the user.

**Sampling.** Let $X$ denote the distribution of the randomly perturbed documents. In this context, $X$ is generated as follows: first, pick $s$ uniformly at random from $[d]$ (the local dictionary), then choose a set $S \subseteq [d]$ of size $s$ uniformly at random. Finally, remove all occurrences of words appearing in $S$ from $\xi$, where removing means replacing with the `UNK` token. For simplicity, it is assumed that tokens and words coincide. The perturbed samples $X_1, \ldots, X_n$ are independent and identically distributed repetitions of this process.

Associated with the $X_i$ samples are vectors $Z_1, \ldots, Z_n \in \{0, 1\}^d$, indicating the presence or absence of a word in $X_i$. Specifically, $Z_{i,j} = 1$ if word $j$ is present in $X_i$ and 0 otherwise.

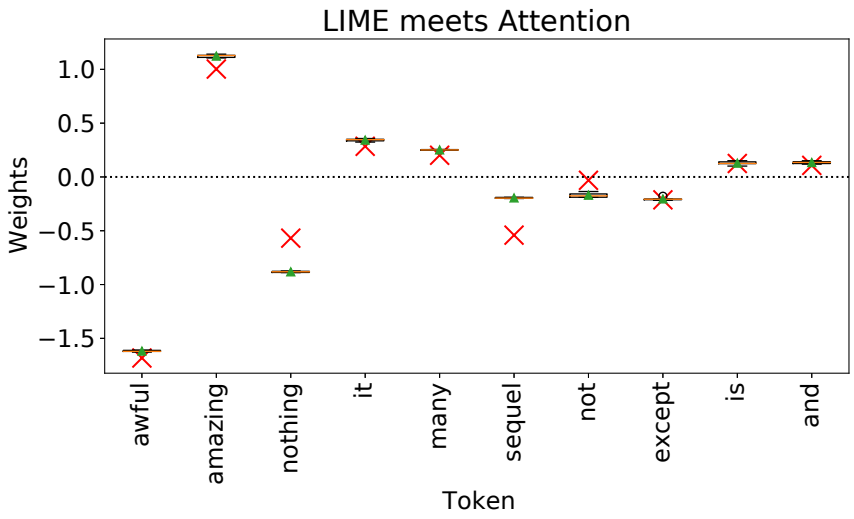


Figure 6.5 – **Illustration of the accuracy of Eq. (6.19).** The boxplots show the results from 5 runs of LIME with default parameters, while the red crosses indicate the predictions given by Theorem 6.6.1. The document $\xi$ contains $T = 99$ tokens and $d = 71$ distinct words and is classified as a negative review. Note that Theorem 6.6.1 holds true even with $T \neq d$ for reasonable word multiplicities, as discussed in Section 6.7.

**Weights.** Each new sample $X_i$ receives a positive weight $\pi_i$, defined as

$$\pi_i := \exp\left(\frac{-d(\mathbb{1}, Z_i)^2}{2\nu^2}\right), \tag{6.15}$$

where $d$ is the *cosine distance* and $\nu > 0$ is a bandwidth parameter. The intuition behind these weights is that $X_i$ can be far from $\xi$ if many words are removed. In the most extreme case, where $s = d$, all words from $\xi$ are removed. In such cases, $z_i$ has mostly 0 components and is distant from $\mathbb{1}$ in terms of cosine distance.

**Surrogate Model.** The next step involves training a surrogate model on $Z_1, \ldots, Z_n$, aiming to match the responses $Y_i := f(X_i)$. In the default implementation of LIME, this model is linear and is obtained through weighted ridge regression. Formally, LIME outputs

$$\widehat{\beta}_n^\lambda \in \operatorname*{arg\,min}_{\beta \in \mathbb{R}^{d+1}} \left\{ \sum_{i=1}^n \pi_i (y_i - \beta^\top z_i)^2 + \lambda \|\beta\|^2 \right\}, \tag{6.16}$$

where $\lambda > 0$ is a regularization parameter. The components of $\widehat{\beta}_n^\lambda$ are referred to as the *interpretable coefficients*, with the 0-th coordinate conventionally representing the intercept.

### 6.6.2 Limit Explanations

Under mild assumptions, Mardaoui and Garreau [2021, Theorem 1] demonstrate that LIME's coefficients converge to *limit coefficients* $\beta^\infty$. Specifically, this convergence occurs when the num-

ber of perturbed samples $n$ is large, the penalization in Eq. (6.16) is not too strong (which is typically the default case), and the bandwidth $\nu$ is also large.

The expression for the limit coefficient associated with word $j$ is

$$\beta_j^\infty = 3\mathbb{E}\left[f(X) \mid j \notin S\right] - \frac{3}{d}\sum_k \mathbb{E}\left[f(X) \mid k \notin S\right] . \tag{6.17}$$

This coefficient can be computed (exactly or approximately) as a function of the model parameters, providing precise insights into LIME's behavior in this context. This computation represents the main result of this section.

Before presenting Theorem 6.6.1, some additional notation is required. The index $h$ will denote the quantity corresponding to the `[UNK]` token. Specifically, $k_{h,t} := W_k h + W_k W_p(t) \in \mathbb{R}^{d_{\text{att}}}$ is the key vector associated with the `[UNK]` token at position $t \in [T_{\max}]$. For any $t \in [T_{\max}]$, we define:

$$g_{h,t} := \exp\left(\frac{q^\top k_{h,t}}{\sqrt{d_{\text{att}}}}\right) , \tag{6.18}$$

and

$$\alpha_{h,t} := \frac{g_{h,t}}{\sum_u g_{h,u}} .$$

Here, $\alpha_{h,t}$ represents the attention corresponding to the `[UNK]` token at position $t$, from the perspective of the query associated with the `[CLS]` token. Finally, set $v_{h,t} := W_v(h + W_p(t))$.

**Theorem 6.6.1 (LIME meets Attention).** *Assume that $d = T = T_{\max}^\varepsilon$, with $\varepsilon \in (0,1)$. Further assume the existence of positive constants $0 < c < C$ such that, as $T \to +\infty$, for all $t \in [T_{\max}]$, $\max(|v_t|, |v_{h,t}|) \leq C$, and $c \leq \min(g_t, g_{h,t}) \leq C$. Then,*

$$\beta_j^\infty = \frac{3}{2K}\sum_{i=1}^{K}\sum_{t=1}^{T_{\max}} W_\ell^{(i)}\left(\alpha_t^{(i)} v_t^{(i)} - \alpha_{h,t}^{(i)} v_{h,t}^{(i)}\right)\mathbb{1}_{\xi_t = j} + \mathcal{O}\left(T_{\max}^{(2-\varepsilon)\vee 3/2}\right) . \tag{6.19}$$

Theorem 6.6.1 is proved in Section C.2. The main challenge in the proof is to derive accurate approximations for Eq. (6.17), as the considered model, despite being single-layered, exhibits high non-linearity.

It is important to note that Theorem 6.6.1 assumes all tokens are distinct. While it is conjectured that this assumption can be relaxed (as discussed in Section 6.7), it is necessary for a rigorous comparison between attention and LIME explanations. However, the experiments were conducted without assuming distinct tokens.

Despite this, a very good match was observed between the theoretical approximation and the empirical outputs of LIME with default parameters. This is illustrated in Figure 6.5 on a specific example, and further quantified in Appendix C.2.

Several observations can be drawn:

(i) It is evident from Eq. (6.19) that LIME explanations are **quite different** from gradient-based explanations (Eq. (6.14)), except for the leading term (which is proportional to $\alpha_t W_\ell v_t$) found in both expressions.

(ii) It is apparent that **LIME explanations are approximately an affine transformation of the** $\alpha_t^{(i)}$.

(iii) As with gradient-based explanations, there is a significant difference compared to plain attention-based explanations: the final layer plays a crucial role in the explanation. To put it simply, assume $K = 1$ and $W_\ell v_t = 0$, then **the influence of $\alpha_t$ vanishes regardless of its value**. This is advantageous for LIME, as it corresponds to the model nullifying the influence of token $t$ in later stages, despite a positive attention score.

(iv) From Eq. (6.19), it is clear that **LIME explanations will be near zero** whenever $\alpha_t^{(i)} v_t^{(i)} \approx \alpha_{h,t}^{(i)} v_{h,t}^{(i)}$, indicating that the **attention $\times$ value of head $i$ for token $t$ is comparable to that for the `[UNK]` token**. This highlights the importance of carefully choosing the embedding for the replacement token.

## 6.7 Limitations

The model described in Section 6.3 differs from practical architectures primarily in the following aspects: (i) number of layers, (ii) skip connections, and (iii) non-linearities:

(i) Only a single layer is considered, which introduces non-linearity and presents a challenging analysis while providing good performance for the task.

(ii) Skip connections are not included as they did not enhance performance, but the analysis can easily adapt to include them since this operation is linear.

(iii) Following typical theoretical works [Gunasekar et al., 2018], additional non-linearities, such as ReLU activations or its variants, are not incorporated.

The main limitation of this analysis is its focus on a single-layer model. This allows for a deep theoretical exploration, yet its applicability to more complex architectures may not be straightforward. Extending this analysis to multi-layer architectures introduces additional theoretical challenges. In multi-layer transformers, approximation errors can accumulate and intensify as they propagate through the network. Moreover, assumptions valid for a single layer may not hold for deeper networks, especially those with non-linearities like ReLU activations. However, even for simple architectures, many questions remain unresolved, and the interpretability of these models has not been thoroughly studied in a formal manner.

Several theoretical studies on transformers share these limitations. For example, Jelassi et al. [2022] elucidates how Vision Transformers discern spatial patterns within a single-layer, single-head architecture. Similarly, Tarzanagh et al. [2023] establishes the correspondence between the optimization geometry of self-attention and an SVM problem. Von Oswald et al. [2023] suggests that training transformers with a specific objective can induce a form of meta-learning, exemplified on a linear single-layer model. Additionally, Edelman et al. [2022] investigate transformers in time series forecasting, showcasing their superiority over traditional methods by accurately capturing temporal dependencies. In a bid to enhance transformer efficiency and scalability, Fu et al. [2023] introduce sparse attention mechanisms and efficient training strategies by formalizing single-layer transformers. Furthermore, Makkuva et al. [2024] tackle transformer interpretability, developing tools to visualize and comprehend attention patterns within the models, aiming to bridge the gap between high performance and the imperative for transparency in critical applications. Appendix C.5 reports experiments on a multi-layer transformer.

Theorem 6.6.1 also has limitations. First, as in previous work, the approximation holds true for large document and window sizes. While this is expected, the results are experimentally satisfying for documents a few dozen tokens long. Second, Theorem 6.6.1 assumes all tokens are distinct. This simplification allows for a rigorous formalization of LIME's behavior using its de-

fault parameters as defined in the official implementation. Experimentally, Eq. (6.19) holds for documents containing repeated words. Theorem 6.6.1 was validated without this assumption by computing the norm-2 error between the official LIME weights and our approximation (illustrated in Figure 6.5). The average error over the full test set (described in Appendix C.6) is $0.808$, with a standard deviation of $0.219$. Theoretically, this assumption can be relaxed if the maximal multiplicity of tokens is small relative to $T$. If many tokens are identical, this is no longer true: consider, for instance, the extreme case of two groups of identical tokens. Further details are discussed in Appendix C.2.1.

## 6.8 Conclusion and Future Work

This chapter presents a theoretical analysis of how post-hoc explanations relate to a single-layer multi-head attention-based network. By providing exact and approximate expressions for post-hoc explanations on such a model, this work contributes to the ongoing debate in this area. These expressions highlight the fundamental differences between attention-based, gradient-based, and perturbation-based explanations. This deeper understanding not only enriches the discourse surrounding interpretability but also offers valuable insights for practitioners and researchers navigating the complexities of transformer interpretation.

The quest for perfect explanations remains elusive; no single method has emerged as entirely satisfactory. Current models employ attention scores in a non-intuitive manner to arrive at the final prediction. These scores undergo a series of transformations, often ignored when focusing solely on attention scores. Additionally, attention scores always provide a positive explanation, unlike most perturbation-based and gradient-based approaches. For these reasons, these approaches can extract more valuable insights than merely examining attention weights. This finding aligns with the assertions made by Bastings and Filippova [2020].

Future work will broaden the scope of this analysis by extending investigations to a diverse range of post-hoc interpretability methods, including Anchors, to understand model explanations across different methodologies. Similar statements (connecting explanations to the model's parameters) will be sought for more complex architectures, including skip connections, additional non-linearities, and multi-layer models. This will help discern the relationship between model parameters and different explanations. Additionally, the interplay between the sampling mechanism of perturbation-based methods (often replacing at the word level) and the tokenizer used by the model (tokens are often subwords) will be explored further.

The focus of this paper has been on text classification, leveraging well-established and broadly studied post-hoc explainers for a thorough theoretical analysis. However, the scope of applications for this analysis will be expanded. Specifically, while this study focused on token-level explanations, future research will extend these findings beyond text models to encompass other domains, such as computer vision.

# CHAPTER 7

# Conclusion and Perspectives

## 7.1 Conclusion

This PhD thesis has contributed to the field of Machine Learning interpretability by studying its foundations. Chapter 3 highlighted problems and unexpected behaviors in methods widely used by researchers and practitioners. Through empirical comparisons and rigorous mathematical analyses, the limitations of these methods have been brought to light, in particular about Anchors (Chapter 4) and attention-based explanations (Chapter 6). Additionally, Chapter 5 introduced FRED: a new method based on insights derived from the analysis of other methods, which aims to overcome some of their limitations. In particular, **this thesis has focused on local and post-hoc interpretability methods**, which explain individual predictions of machine learning models without intervening in their training or the design of their architecture. Additionally, except for Chapter 6, this thesis has examined model-agnostic methods: designed to explain any black-box model without knowing its nature or using any of its internal parameters. While text classification, particularly with a limited number of classes, has been extensively studied and developed through various models, tools, and research, the interpretability of these models remains a significant challenge. Therefore, this thesis has primarily focused on this task.

The starting point of the PhD has been noticing that **different explainers often produce diverse, sometimes even conflicting explanations.** This phenomenon occurs even with the most commonly used methods and on simple tasks such as sentiment analysis. The natural consequence of this observation has been to question which method produces better explanations. While metrics such as accuracy, precision, and recall are commonly employed to compare different models for text classification across well-known benchmark datasets, a similar standard for evaluating interpretability has been largely absent. This topic has been discussed in Section 1.5.2, and some options specifically for NLP have been explored in Section 2.4. This problem has been even more pronounced when comparing methods that produce different types of explanations, such as feature attribution and rule-based methods. In such case, **one would expect that the keywords identified as most crucial by a rule-based method have been highlighted as most important by a feature attribution method.** However, it has been shown in this thesis that this is not guaranteed, nor is it clear which explanation is the most faithful to the model.

To this end, Lopardo and Garreau [2022] presented in Chapter 3 has compared the explanations of LIME [Ribeiro et al., 2016] and Anchors [Ribeiro et al., 2018] on simple models, for which the decision-making process is precisely known. Qualitative experiments (Section 3.3.1) have shown that **Anchors exhibit unclear behavior that depends on the multiplicity of words in the text:**

**the explanation changes depending on how many times an important word appears in the text**. In contrast, quantitative experiments (Section 3.3.2) have shown that LIME has been much more faithful to a linear classification model compared to Anchors. This was expected, given that LIME uses a linear model as its local surrogate.

In essence, what has emerged from this comparison is that **different methods effectively explain different aspects of the prediction**, depending on their internal mechanisms. Therefore, the explanation itself must be interpreted, and to do so correctly, **it is essential to precisely understand the foundations of the method being used.**

LIME, with its pros and cons, has been extensively studied. In contrast, Anchors remained popular but insufficiently studied. For this reason, Lopardo et al. [2023a] has presented the first comprehensive study of Anchors, revealing its strengths and limitations. The analysis has included a detailed exploration of Anchors precision and coverage, the algorithm, and its sampling methods. The findings have provided a robust theoretical foundation for understanding and improving Anchors.

Subsequently, leveraging insights from these previous analyses, Lopardo et al. [2023b] has proposed FRED: a local, post-hoc, and model-agnostic method with clear and explicit definitions, which aims to overcome some of the limitations of LIME and Anchors. FRED provides explanations in the form of feature attribution (sentence highlighting), by identifying the minimal set of most important keywords for the prediction, and through example-based explanations, showcasing slightly perturbed versions of the original text that receive a different classification (counterfactuals) or the same classification (prototypes).

Chapter 5 has provided a theoretical analysis that illustrates the behavior of FRED when applied to intrinsically interpretable models, in the same spirit as Lopardo et al. [2023a]. Additionally, experiments comparing various explainability techniques, including LIME, SHAP, and Anchors, across different datasets and models, have validated the robustness and faithfulness of FRED's explanations.

The thesis has then shifted to the study of attention-based explanations in Chapter 6. The attention mechanism [Bahdanau et al., 2015], notably used by transformers [Vaswani et al., 2017], produces attention weights for each input token, leading to the natural idea of using these weights as explanations. These weights have been interpreted as a measure of how much the model focuses on different parts of the input text.

However, there has been a wide and animated debate on whether these attention weights can be used as explanations, summarized in Section 6.2. Lopardo et al. [2024] has entered this debate with a mathematical analysis that has highlighted the distinctions between attention-based and other post-hoc explanations. In summary, Section 6.3 defined a mathematically tractable and simplified version of a multi-head, single-layer transformer. **The analysis has demonstrated that post-hoc methods capture more useful insights than merely examining attention weights.** In fact, while attention weights undoubtedly provide useful information about a model's decision-making process, focusing solely on these weights ignores the way they are interconnected throughout the network.

## 7.2 Perspectives

Building on the findings and contributions of this thesis, several promising avenues for future research and development can be identified.

**Word replacement.** A first direction is to study the impact of different methods for perturbing text documents. It has been shown that the sampling scheme used by perturbation-based methods (Section 1.4.6) is central and has a direct impact on the explanations. Directly removing words to alter the text can be problematic for some models, as it changes the structure and length of the text, and likely disrupts the semantic and grammatical structure. Replacing with an out-of-dictionary mask token, as default several techniques, including LIME, SHAP, Anchors, is an improvement, but the resulting samples are out-of-distribution (OOD, Section 1.5.1) and still lack coherence.

The `pos-sampling` used by FRED (Section 5.2.2) is a significant improvement, as it replaces the selected word with one that retains the same *part-of-speech* tag. However, this does not guarantee that the generated samples are coherent (for example, the sentence "I love this movie" could become "I eat this movie"), and thus it is not immune to the OOD issue.

A promising approach could be to rely on vector representations such as Word2Vec [Mikolov et al., 2013] or GloVe [Pennington et al., 2014]. In this case, a concept of similarity between words can be defined based on the distance of their vector representations. *Similar* words should have *similar* vector representations in some sense. If trained on an appropriate corpus, these representations can be leveraged to replace a word with one that has a similar representation. Preliminary experiments (on small datasets) have shown that the generated samples are still largely nonsensical. However, using an adapted corpus designed to maximize the similarity between vector representations of words that are likely to belong to the same context can represent an effective (and computationally lightweight) method for addressing word replacement.

Replacing missing words using BERT [Devlin et al., 2019] is an alternative implemented as a second option in the Anchors repository [*]. However, empirical results [†] have shown that this option is not entirely satisfactory in the context of interpretability. BERT, in fact, is specifically designed to predict missing words, and it performs well in this task. If an example is moderately large and contains many positive words, for instance, BERT understands from the context that the text is positive and, when replacing a masked word, will most likely use another word with a positive connotation. As a result, the underlying model likely maintains the same prediction whether an important token is present in a sample or not, and thus the explainer fails to identify it as important.

Such a task seems straightforward for modern LLMs. However, a more significant issue could be the computational cost: generating 5000 perturbed samples (as LIME by default) using an LLM can be very resource-intensive, which is why this approach is not widely used. It would be interesting to quantitatively measure which method works best. This obviously depends on the underlying model and the specific task for which the replacement is needed. However, an intriguing research question is to identify one or a set of metrics to measure the quality of text perturbations.

**Evaluation of explanations.** Consequently, it would be possible to make progress in the challenging discourse of evaluation metrics for explanations (Section 1.5.2). The metrics introduced in Section 2.4) heavily rely on word replacement. The *faithfulness* metrics used in Chapter 5, namely the *sufficiency* and *comprehensiveness* as defined in DeYoung et al. [2020], measure the quality of a rule-based explanation (*i.e.*, the relevance of the identified keywords) when they are *removed* from the original document or when they are the only words present, respectively. To achieve this, other

*. https://github.com/marcotcr/anchor

†. https://github.com/marcotcr/anchor/blob/master/notebooks/Anchor%20for%20text.ipynb

words are masked with a predefined token (such as UNK). Similarly, the *AUCMorf* [Kakogeorgiou and Karantzalos, 2021], which measures the quality of feature-based explanations, evaluates the model's prediction deviation when the top $k$ tokens in order of importance are *removed*: in practice replaced by a token. Depending on the mask, one explainer may be favored over another, making comparisons using such metrics potentially unfair [Hsia et al., 2024]. Adopting an optimal mask token is therefore crucial.

**Adapting the analysis to different data types.** The analysis of Anchors [Lopardo et al., 2023a] discussed in Chapter 4 can be extended to its implementation for tabular data and images. In the case of tabular data, Anchors identifies the minimal set of input features that most significantly impact the prediction, maximizing *coverage* while ensuring high *precision*, exactly as in Eq. 4.3. Similarly, for image classifiers, the critical difference is that the input features are superpixels (*i.e.*, groups of pixels), as in the case of LIME. Anchors, therefore, identifies the minimal set of superpixels that satisfy Eq. 4.3.

Similarly, the analysis of attention-based explanations can be extended to vision models. The definitions for attention in Section 6.3.2 remain applicable, and the model in Section 6.3 can be adapted with minor adjustments for the task of image classification. In fact, explanations are considered at the token level, and whether the token is a word or a pixel does not change the analysis. The same applies to gradient-based explanations, meaning that the results from Section 6.5 are equally valid for image data.

The challenging part that requires further work is related to LIME. In practice, LIME operates differently on various types of data. For Section 6.6, insights from preious work have been adapted to the model of Section 6.3, in order to obtain the explicit LIME weights for each token. In the case of images, additional considerations are necessary. Fortunately, LIME on images has also been studied. The main challenge lies in adapting the model in exam to Garreau and Mardaoui [2021, Proposition 2], as the number of superpixels and the superpixel to which each pixel belongs are not known a priori.

As detailed in Section 5.5, FRED can also be extended to image classifiers and tabular data by adapting its sampling process. For images, common perturbation methods, such as pixel masking or blurring, risk creating out-of-distribution samples. Vision-language models can generate more realistic perturbations, but are resource-intensive. For tabular data, perturbation methods include replacing values with mean or median values or using domain-specific logic, with challenges in preserving feature distribution and relationships. The key challenge is developing a fast and effective sampling method that avoids systematically producing out-of-distribution samples, ensuring the reliability of FRED's explanations.

**Exploring the technical explainability requirements in regulation.** As introduced in Section 1.3.2, interpretability is one of the central points around which new artificial intelligence regulations revolve. The **AI Act of the European Union mandates that decisions made by *high-risk* AI applications must be explainable.** More specifically, Article 13 of on *Transparency and provision of information to deployers* mandates that

> High-risk AI systems shall be designed and developed in such a way as to ensure that their operation is sufficiently transparent to enable deployers to interpret a system's output and use it appropriately. An appropriate type and degree of transparency shall

> be ensured with a view to achieving compliance with the relevant obligations of the provider and deployer [...].

Similarly, Article 14 on *Human oversight* states that

> [...] the high-risk AI system shall be provided to the deployer in such a way that natural persons to whom human oversight is assigned are enabled, as appropriate and proportionate [...] to correctly interpret the high-risk AI system's output, taking into account, for example, the interpretation tools and methods available [...].

It is crucial to interpret the regulations to understand the necessary requirements for explanations. The law is not precise: Article 13 refers to "an appropriate type and degree of transparency," leaving the question of what this means open. Additionally, Article 14 says "taking into account the interpretation tools and methods available", implying that existing methods are compliant, but it has already been shown that these methods are often flawed.

Preliminary interpretations suggest that only self-explainable models (Section 1.4.4) may be suitable for compliance. Bordt et al. [2022] argue that post-hoc explanation algorithms, such as SHAP and LIME, fail to meet legal and regulatory transparency requirements, especially in adversarial contexts where the explanation provider and receiver have conflicting interests, proving that post-hoc explanations can be manipulated, lack clear standards, and do not reliably convey the true reasons behind decisions.

In the case of the GDPR, the conflict between the *right to be forgotten* and the need for actionable explanations has already proven problematic, as discussed in Pawelczyk et al. [2022b]. The study highlights how data deletion requests can significantly impact the validity of algorithmic recourse, emphasizing the need for careful consideration of these regulatory requirements. Additionally, a recent review [Nannini et al., 2024] reveals that many XAI research papers engage with ethical considerations only superficially, without substantial engagement to practically integrating ethical analysis into XAI development. This underscores the need for a deeper, more comprehensive integration of ethical considerations in XAI research and practice to ensure that transparency and explainability requirements are met effectively.

# Bibliography


Talal A. A. Abdullah, Mohd Soperi Mohd Zahid, and Waleed Ali. A Review of Interpretable ML in Healthcare: Taxonomy, Applications, Challenges, and Future Directions. *Symmetry*, 13(12): 2439, 2021.

Eldar D. Abraham, Karel D'Oosterlinck, Amir Feder, Yair Gat, Atticus Geiger, Christopher Potts, Roi Reichart, and Zhengxuan Wu. CEBaB: Estimating the Causal Effects of Real-World Concepts on NLP Model Behavior. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:17582–17596, 2022.

Carlo Abrate and Francesco Bonchi. Counterfactual Graphs for Explainable Classification of Brain Networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 2495–2504, 2021.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*, 2023.

Amina Adadi and Mohammed Berrada. Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). *IEEE access*, 6:52138–52160, 2018.

David Adam. Lethal AI weapons are here: how can we control them? *Nature*, 2024.

Chirag Agarwal, Nari Johnson, Martin Pawelczyk, Satyapriya Krishna, Eshika Saxena, Marinka Zitnik, and Himabindu Lakkaraju. Rethinking Stability for Attribution-based Explanations. In *ICLR International Workshop on Privacy, Accountability, Interpretability, Robustness, Reasoning on Structured Data (PAIR$^2$Struct)*, 2022a.

Chirag Agarwal, Satyapriya Krishna, Eshika Saxena, Martin Pawelczyk, Nari Johnson, Isha Puri, Marinka Zitnik, and Himabindu Lakkaraju. OpenXAI: Towards a Transparent Evaluation of Post hoc Model Explanations. *Advances in Neural Information Processing Systems (NeurIPS)*, 35: 15784–15799, 2022b.

Rishabh Agarwal, Levi Melnick, Nicholas Frosst, Xuezhou Zhang, Ben Lengerich, Rich Caruana, and Geoffrey E Hinton. Neural Additive Models: Interpretable Machine Learning with Neural Nets. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:4699–4711, 2021.

Daehwan Ahn, Abdullah Almaatouq, Monisha Gulabani, and Kartik Hosanagar. Impact of Model Interpretability and Outcome Feedback on Trust in AI. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 1–25, 2024.

Samir N Ajani, Prashant Khobragade, Mrunalee Dhone, Bireshwar Ganguly, Nilesh Shelke, and Namita Parati. Advancements in Computing: Emerging Trends in Computational Science with Next-Generation Computing. *International Journal of Intelligent Systems and Applications in Engineering*, 12(7s):546–559, 2024.

Selin Akgun and Christine Greenhow. Artificial intelligence in education: Addressing ethical challenges in K-12 settings. *AI and Ethics*, 2(3):431–440, 2022.

E. Albini, A. Rago, P. Baroni, F. Toni, et al. Relation-Based Counterfactual Explanations for Bayesian Network Classifiers. In *Proceeding of the International Joint Conference on Artificial Intelligence (IJCAI)*, volume 2021, pages 451–457, 2020.

David Alvarez-Melis and Tommi Jaakkola. A causal framework for explaining the predictions of black-box sequence-to-sequence models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 412–421, 2017.

David Alvarez Melis and Tommi Jaakkola. Towards robust interpretability with self-explaining neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 31, 2018.

Salim I. Amoukou and Nicolas J. B. Brunel. Consistent Sufficient Explanations and Minimal Local Rules for explaining regression and classification models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards Better Understanding of Gradient-based Attribution Methods for Deep Neural Networks. In *International Conference on Learning Representations (ICLR)*, 2018.

Sule Anjomshoae, Timotheus Kampik, and Kary Främling. Py-CIU: A Python Library for Explaining Machine Learning Predictions Using Contextual Importance and Utility. In *IJCAI-PRICAI 2020 Workshop on Explainable Artificial Intelligence (XAI), January 8, 2020*, 2020.

Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. "What is relevant in a text document?": An interpretable machine learning approach. *PloS one*, 12(8):e0181142, 2017.

Leila Arras, Ahmed Osman, Klaus-Robert Müller, and Wojciech Samek. Evaluating Recurrent Neural Network Explanations. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 113–126, 2019.

Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, , et al. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information fusion*, 58:82–115, 2020.

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. A Diagnostic Study of Explainability Techniques for Text Classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3256–3274, 2020.

Giuseppe Attanasio, Eliana Pastor, Chiara Di Bonaventura, and Debora Nozza. ferret: a Framework for Benchmarking Explainers on Transformers. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, 2023.

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations (ICLR)*, 2015.

Aram Bahrini, Mohammadsadra Khamoshifar, Hossein Abbasimehr, Robert J Riggs, Maryam Esmaeili, Rastin Mastali Majdabadkohne, and Morteza Pasehvar. ChatGPT: Applications, opportunities, and threats. In *2023 Systems and Information Engineering Design Symposium (SIEDS)*, pages 274–279. IEEE, 2023.

Pietro Barbiero, Gabriele Ciravegna, Francesco Giannini, Pietro Lió, Marco Gori, and Stefano Melacci. Entropy-based logic explanations of neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6046–6054, 2022.

Solon Barocas, Moritz Hardt, and Arvind Narayanan. *Fairness and machine learning: Limitations and opportunities*. MIT Press, 2023.

Jasmijn Bastings and Katja Filippova. The elephant in the interpretability room: Why use attention as explanation when we have saliency methods? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 149–155, 2020.

Jasmijn Bastings, Sebastian Ebert, Polina Zablotskaia, Anders Sandholm, and Katja Filippova. "Will You Find These Shortcuts?" A Protocol for Evaluating the Faithfulness of Input Salience Methods for Text Classification. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 976–991, 2022.

BBC News. Facebook apology as AI labels black men 'primates', September 6 2021.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623, 2021.

José Manuel Benítez, Juan Luis Castro, and Ignacio Requena. Are artificial neural networks black boxes? *IEEE Transactions on neural networks*, 8(5):1156–1164, 1997.

Richard Berk, Hoda Heidari, Shahin Jabbari, Michael Kearns, and Aaron Roth. Fairness in criminal justice risk assessments: The state of the art. *Sociological Methods & Research*, 50(1): 3–44, 2021.

Andrew C Berry. The accuracy of the Gaussian approximation to the sum of independent variates. *Transactions of the American Mathematical Society*, 49(1):122–136, 1941.

Jasmin Bharadiya. Artificial intelligence in transportation systems a critical review. *American Journal of Computing and Engineering*, 6(1):34–45, 2023.

Umang Bhatt, Adrian Weller, and José MF Moura. Evaluating and aggregating feature-based model explanations. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3016–3022, 2021.

Adrien Bibal, Rémi Cardon, David Alfter, Rodrigo Wilkens, Xiaoou Wang, Thomas François, and Patrick Watrin. Is attention explanation? An introduction to the debate. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3889–3900, 2022.

Blair Bilodeau, Natasha Jaques, Pang Wei Koh, and Been Kim. Impossibility theorems for feature attribution. *Proceedings of the National Academy of Sciences*, 121(2):e2304406120, 2024.

Christopher M. Bishop and Nasser M. Nasrabadi. *Pattern Recognition and Machine Learning*, volume 4. Springer, 2006.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet Allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

Francesco Bodria, Fosca Giannotti, Riccardo Guidotti, Francesca Naretto, Dino Pedreschi, and Salvatore Rinzivillo. Benchmarking and survey of explanation methods for black box models. *Data Mining and Knowledge Discovery*, 37(5):1719–1778, 2023.

Adam Bohr and Kaveh Memarzadeh. The rise of artificial intelligence in healthcare applications. In *Artificial Intelligence in healthcare*, pages 25–60. Elsevier, 2020.

Sebastian Bordt and Ulrike von Luxburg. From Shapley values to generalized additive models and back. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 709–745. PMLR, 2023.

Sebastian Bordt and Ulrike von Luxburg. Statistics without Interpretation: A Sober Look at Explainable Machine Learning. *arXiv preprint arXiv:2402.02870*, 2024.

Sebastian Bordt, Michèle Finck, Eric Raidl, and Ulrike von Luxburg. Post-hoc explanations fail to achieve their purpose in adversarial contexts. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, pages 891–905, 2022.

Christian Borgelt. An Implementation of the FP-growth Algorithm. In *Proceedings of the 1st international workshop on open source data mining: frequent pattern mining implementations*, pages 1–5, 2005.

Ali Borji. Generated Faces in the Wild: Quantitative Comparison of Stable Diffusion, Midjourney and DALL-E 2. *arXiv preprint arXiv:2210.00586*, 2022.

Philip Nicholas Boucher. Artificial intelligence: How does it work, why does it matter, and what can we do about it? *EPRS: European Parliamentary Research Service*, 2020.

Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press, 2013.

Steven Bramhall, Hayley Horn, Michael Tieu, and Nibhrat Lohia. Qlime-a quadratic local interpretable model-agnostic explanation approach. *SMU Data Science Review*, 3(1):4, 2020.

L. Breiman, J. Friedman, R. Olshen, and C. J. Stone. Classification and Regression Trees, 1983.

Andrei Z. Broder, Steven C. Glassman, Mark S. Manasse, and Geoffrey Zweig. Syntactic clustering of the web. *Computer networks and ISDN systems*, 29(8-13):1157–1166, 1997.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:1877–1901, 2020.

Gino Brunner, Yang Liu, Damian Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. On Identifiability in Transformers. In *International Conference on Learning Representations (ICLR)*, 2020.

Eleanor R. Burgess, Ivana Jankovic, Melissa Austin, Nancy Cai, Adela Kapuścińska, Suzanne Currie, J. Marc Overhage, Erika S. Poole, and Jofish Kaye. Healthcare AI treatment decision support: Design principles to enhance clinician adoption and trust. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–19, 2023.

Corinne Cath. Governing artificial intelligence: ethical, legal and technical opportunities and challenges. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 376(2133):20180080, 2018.

Simon Caton and Christian Haas. Fairness in machine learning: A survey. *ACM Computing Surveys*, 56(7):1–38, 2024.

Martin Charachon. *Designing Visual Explanations of Deep Learning Classifiers Decisions in Medical Image Analysis*. PhD thesis, Université Paris-Saclay, 2023.

Martin Charachon, Céline Hudelot, Paul-Henry Cournede, Camille Ruppli, and Roberto Ardon. Combining Similarity and Adversarial Learning to Generate Visual Explanation: Application to

Medical Image Classification. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 7188–7195. IEEE, 2021.

Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE winter conference on applications of computer vision (WACV)*, pages 839–847. IEEE, 2018.

Hila Chefer, Shir Gur, and Lior Wolf. Transformer interpretability beyond attention visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 782–791, 2021.

Saneem Chemmengath, Amar Prakash Azad, Ronny Luss, and Amit Dhurandhar. Let the CAT out of the bag: Contrastive Attributed explanations for Text. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7190–7206, 2022.

Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K. Su. This looks like that: deep learning for interpretable image recognition. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019.

Lijia Chen, Pingping Chen, and Zhijian Lin. Artificial intelligence in education: A review. *IEEE Access*, 8:75264–75278, 2020.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240): 1–113, 2023.

Gabriele Ciravegna, Pietro Barbiero, Francesco Giannini, Marco Gori, Pietro Lió, Marco Maggini, and Stefano Melacci. Logic Explained Networks. *arXiv preprint arXiv:2108.05149*, 2021.

Gabriele Ciravegna, Pietro Barbiero, Francesco Giannini, Marco Gori, Pietro Lió, Marco Maggini, and Stefano Melacci. Logic explained networks. *Artificial Intelligence*, 314:103822, 2023.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What Does BERT Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, 2019.

Ian Covert, Scott Lundberg, and Su-In Lee. Explaining by removing: A unified framework for model explanation. *Journal of Machine Learning Research*, 22(209):1–90, 2021.

Danilo Croce, Daniele Rossini, and Roberto Basili. Explaining non-linear classifier decisions within kernel-based deep architectures. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 16–24, 2018.

Riccardo Crupi, Alessandro Castelnovo, Daniele Regoli, and Beatriz San Miguel Gonzalez. Counterfactual explanations as interventions in latent space. *Data Mining and Knowledge Discovery*, pages 1–37, 2022.

Hugo Cui, Freya Behrens, Florent Krzakala, and Lenka Zdeborova. A Phase Transition between Positional and Semantic Learning in a Solvable Model of Dot-Product Attention. In *High-dimensional Learning Dynamics 2024: The Emergence of Structure and Reasoning*, 2024.

Piotr Dabkowski and Yarin Gal. Real time image saliency for black box classifiers. *Advances in Neural Information Processing Systems (NeurIPS)*, 30, 2017.

Marina Danilevsky, Kun Qian, Ranit Aharonov, Yannis Katsis, Ban Kawas, and Prithviraj Sen. A Survey of the State of Explainable AI for Natural Language Processing. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 447–459, 2020.

Anubrata Das, Chitrank Gupta, Venelin Kovatchev, Matthew Lease, and Junyi Jessy Li. ProtoTEx: Explaining Model Decisions with Prototype Tensors. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2986–2997, 2022.

Luca Deck, Jakob Schoeffer, Maria De-Arteaga, and Niklas Kühl. A Critical Survey on Fairness Benefits of Explainable AI. In *The 2024 ACM Conference on Fairness, Accountability, and Transparency*, pages 1579–1595, 2024.

Julien Delaunay, Luis Galárraga, and Christine Largouët. Improving anchor-based explanations. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 3269–3272, 2020.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255. Ieee, 2009.

Misha Denil, Alban Demiraj, and Nando De Freitas. Extraction of Salient Sentences from Labelled Documents. *arXiv preprint arXiv:1412.6815*, 2014.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C Wallace. ERASER: A Benchmark to Evaluate Rationalized NLP Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, 2020.

Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. *Advances in Neural Information Processing Systems (NeurIPS)*, 31, 2018.

Persi Diaconis and Sandy Zabell. Closed form summation for classical distributions: variations on a theme of de Moivre. *Statistical Science*, pages 284–302, 1991.

Jon Donnelly, Alina Jade Barnett, and Chaofan Chen. Deformable protopnet: An interpretable image classifier using deformable prototypes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10265–10275, 2022.

Finale Doshi-Velez and Been Kim. Towards A Rigorous Science of Interpretable Machine Learning. *arXiv preprint arXiv:1702.08608*, 2017.

Julia Dressel and Hany Farid. The accuracy, fairness, and limits of predicting recidivism. *Science advances*, 4(1):eaao5580, 2018.

Benjamin L. Edelman, Surbhi Goel, Sham Kakade, and Cyril Zhang. Inductive biases and variable creation in self-attention mechanisms. In *International Conference on Machine Learning (ICML)*, pages 5793–5831. PMLR, 2022.

Upol Ehsan, Q Vera Liao, Michael Muller, Mark O Riedl, and Justin D Weisz. Expanding explainability: Towards social transparency in ai systems. In *Proceedings of the 2021 CHI conference on human factors in computing systems*, pages 1–19, 2021.

Radwa ElShawi, Youssef Sherif, Mouaz Al-Mallah, and Sherif Sakr. ILIME: Local and Global Interpretable Model-Agnostic Explainer of Black-Box Decision. In *Advances in Databases and Information Systems: 23rd European Conference, ADBIS 2019, Bled, Slovenia, September 8–11, 2019, Proceedings 23*, pages 53–68. Springer, 2019.

Gabriel Erion, Joseph D Janizek, Pascal Sturmfels, Scott M Lundberg, and Su-In Lee. Learning explainable models using attribution priors, 2019.

Carl-Gustav Esseen. On the Liapunov limit error in the theory of probability. *Ark. Mat. Astr. Fys.*, 28:1–19, 1942.

Kawin Ethayarajh and Dan Jurafsky. Attention Flows are Shapley Value Explanations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 49–54, 2021.

European Commission. *Ethics guidelines for trustworthy AI*. European Commission and Directorate-General for Communications Networks, Content and Technology, 2019. doi/10 .2759/346720.

Usama Fayyad and Keki Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceeding of the International Joint Conference on Artificial Intelligence (IJCAI)*, 1993.

Zhili Feng, Michal Moshkovitz, Dotan Di Castro, and J. Zico Kolter. An Axiomatic Approach to Model-Agnostic Concept Explanations. *arXiv preprint arXiv:2401.06890*, 2024.

Hidde Fokkema, Rianne de Heide, and Tim van Erven. Attribution-based explanations that provide recourse cannot be robust. *Journal of Machine Learning Research*, 24(360):1–37, 2023.

Hidde Fokkema, Damien Garreau, and Tim van Erven. The Risks of Recourse in Binary Classification. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 550–558. PMLR, 2024.

Ruth C. Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3429–3437, 2017.

International Organization for Standardization. ISO/IEC TR 29119-11:2020 Software and systems engineering, Software testing, Part 11: Guidelines on the testing of AI-based systems, 2020.

Eric J. Friedman. Paths and consistency in additive cost sharing. *International Journal of Game Theory*, 32(4):501–518, 2004.

Jerome H. Friedman and Bogdan E. Popescu. Predictive Learning via Rule Ensembles. *The Annals of Applied Statistics*, pages 916–954, 2008.

Christopher Frye, Damien de Mijolla, Tom Begley, Laurence Cowton, Megan Stanley, and Ilya Feige. Shapley explainability on the data manifold. In *International Conference on Learning Representations (ICLR)*, 2020.

Hengyu Fu, Tianyu Guo, Yu Bai, and Song Mei. What can a Single Attention Layer Learn? A Study Through the Random Features Lens. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pages 11912–11951, 2023.

Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

Philip Gage. A new algorithm for data compression. *The C Users Journal*, 12(2):23–38, 1994.

Andrea Galassi, Marco Lippi, and Paolo Torroni. Attention in natural language processing. *IEEE Transactions on Neural Networks and Learning Systems*, 32(10):4291–4308, 2020.

Damien Garreau and Ulrike Luxburg. Explaining the explainer: A first theoretical analysis of LIME. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1287–1296. PMLR, 2020.

Damien Garreau and Dina Mardaoui. What does LIME really see in images? In *International Conference on Machine Learning (ICML)*, pages 3620–3629. PMLR, 2021.

Damien Garreau and Ulrike von Luxburg. Looking deeper into tabular LIME. *arXiv preprint arXiv:2008.11092, v3*, 2020.

Sara Gerke, Timo Minssen, and Glenn Cohen. Ethical and legal challenges of artificial intelligence-driven healthcare. In *Artificial intelligence in healthcare*, pages 295–336. Elsevier, 2020.

Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. Towards automatic concept-based explanations. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019.

Leilani H. Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pages 80–89. IEEE, 2018.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in Neural Information Processing Systems (NeurIPS)*, 27, 2014.

Riccardo Guidotti. Counterfactual explanations and how to find them: literature review and benchmarking. *Data Mining and Knowledge Discovery*, pages 1–55, 2022.

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Dino Pedreschi, Franco Turini, and Fosca Giannotti. Local Rule-Based Explanations of Black Box Decision Systems. *arXiv preprint arXiv:1805.10820*, 2018a.

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018b.

Riccardo Guidotti, Anna Monreale, Fosca Giannotti, Dino Pedreschi, Salvatore Ruggieri, and Franco Turini. Factual and counterfactual explanations for black box decision making. *IEEE Intelligent Systems*, 34(6):14–23, 2019.

Riccardo Guidotti, Anna Monreale, Stan Matwin, and Dino Pedreschi. Black box explanation by learning image exemplars in the latent feature space. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2019, Würzburg, Germany, September 16–20, 2019, Proceedings, Part I*, pages 189–205. Springer, 2020.

Suriya Gunasekar, Jason D Lee, Daniel Soudry, and Nati Srebro. Implicit bias of gradient descent on linear convolutional networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 31, 2018.

Laurie A Harris and Chris Jaikaran. Highlights of the 2023 Executive Order on Artificial Intelligence for Congress. *Congressional Research Service (CRS) Reports and Issue Briefs*, pages NA–NA, 2023.

Jochen Hartmann, Mark Heitmann, Christian Siebert, and Christina Schamp. More than a feeling: Accuracy and application of sentiment analysis. *International Journal of Research in Marketing*, 40(1):75–87, 2023.

Peter Hase, Harry Xie, and Mohit Bansal. The out-of-distribution problem in explainability and search methods for feature importance explanations. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:3650–3666, 2021.

Trevor Hastie, Robert Tibshirani, Jerome H. Friedman, and Jerome H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, volume 2. Springer, 2009.

Johannes Haug, Stefan Zürn, Peter El-Jiz, and Gjergji Kasneci. On Baselines for Local Feature Attributions. *AAAI-21 Workshop on Explainable Agency in AI*, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

Anna Hedström, Leander Weber, Daniel Krakowczyk, Dilyara Bareeva, Franz Motzkus, Wojciech Samek, Sebastian Lapuschkin, and Marina M-C Höhne. Quantus: An Explainable AI Toolkit for Responsible Evaluation of Neural Network Explanations and Beyond. *Journal of Machine Learning Research*, 24(34):1–11, 2023.

Katherine Hermann, Ting Chen, and Simon Kornblith. The origins and prevalence of texture bias in convolutional neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:19000–19015, 2020.

Daniel S. Hoadley and Nathan J. Lucas. Artificial intelligence and national security. *Congressional Research Service Washington, DC*, 2018.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

Robert C Holte. Very simple classification rules perform well on most commonly used datasets. *Machine learning*, 11:63–90, 1993.

Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. A benchmark for interpretability methods in deep neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019.

Jennifer Hsia, Danish Pruthi, Aarti Singh, and Zachary C Lipton. Goodhart's Law Applies to NLP's Explanation Benchmarks. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 1322–1335, 2024.

Cheng-Yu Hsieh, Chih-Kuan Yeh, Xuanqing Liu, Pradeep Ravikumar, Seungyeon Kim, Sanjiv Kumar, and Cho-Jui Hsieh. Evaluations and Methods for Explanation through Robustness Analysis. In *International Conference on Learning Representations (ICLR)*, 2021.

Xuanxiang Huang and Joao Marques-Silva. The inadequacy of Shapley values for explainability. *arXiv preprint arXiv:2302.08160*, 2023a.

Xuanxiang Huang and Joao Marques-Silva. A Refutation of Shapley Values for Explainability. *arXiv preprint arXiv:2309.03041*, 2023b.

Xuanxiang Huang and Joao Marques-Silva. On the failings of Shapley values for explainability. *International Journal of Approximate Reasoning*, page 109112, 2024.

Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

Matthew Hutson. TAUGHT TO THE TEST: AI software clears high hurdles on IQ tests but still makes dumb mistakes. Can better benchmarks help? *Science*, 376(6593), 2022. 10.1126/science.abq7853.

Lujain Ibrahim, Munib Mesinovic, Kai-Wen Yang, and Mohamad A Eid. Explainable prediction of acute myocardial infarction using machine learning and shapley values. *IEEE Access*, 8: 210410–210417, 2020.

Pavel Izmailov, Polina Kirichenko, Nate Gruver, and Andrew G Wilson. On feature learning in the presence of spurious correlations. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:38516–38532, 2022.

Alon Jacovi and Yoav Goldberg. Aligning faithful interpretations with their social attribution. *Transactions of the Association for Computational Linguistics*, 9:294–310, 2021.

Neham Jain, Vibhhu Sharma, and Gaurav Sinha. Counterfactual Explanations for Visual Recommender Systems. In *Companion Proceedings of the ACM on Web Conference 2024*, pages 674–677, 2024.

Sarthak Jain and Byron C. Wallace. Attention is not Explanation. In *Proceedings of NAACL-HLT*, pages 3543–3556, 2019.

Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, , et al. *An introduction to statistical learning*, volume 112. Springer, 2013.

Steve T.K. Jan, Vatche Ishakian, and Vinod Muthusamy. AI trust in business processes: the need for process-aware explanations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(08):13403–13404, 2020.

Mohammad Hossein Jarrahi. Artificial intelligence and the future of work: Human-AI symbiosis in organizational decision making. *Business horizons*, 61(4):577–586, 2018.

Samy Jelassi, Michael Sander, and Yuanzhi Li. Vision transformers provably learn spatial structure. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:37822–37836, 2022.

Neil Jethani, Mukund Sudarshan, Yindalon Aphinyanaphongs, and Rajesh Ranganath. Have We Learned to Explain? How Interpretability Methods Can Learn to Encode Predictions in their Interpretations. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1459–1467. PMLR, 2021.

Yichen Jiang, Nitish Joshi, Yen-Chun Chen, and Mohit Bansal. Explore, Propose, and Assemble: An Interpretable Model for Multi-Hop Reading Comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2714–2725, 2019.

Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21, 1972.

Kakogeorgiou and Karantzalos. Evaluating explainable artificial intelligence methods for multi-label deep learning classification tasks in remote sensing. *Int. Journal of Applied Earth Observation and Geoinformation*, 2021.

Emilie Kaufmann and Shivaram Kalyanakrishnan. Information complexity in bandit subset selection. In *Conference on Learning Theory*, pages 228–251. PMLR, 2013.

Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International Conference on Machine Learning (ICML)*, pages 2668–2677. PMLR, 2018.

Andreas Knoblauch. Closed-form expressions for the moments of the binomial probability distribution. *SIAM Journal on Applied Mathematics*, 69(1):197–204, 2008.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. Attention is Not Only a Weight: Analyzing Transformers with Vector Norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, 2020.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International Conference on Machine Learning (ICML)*, pages 5338–5348. PMLR, 2020.

Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, et al. Captum: A unified and generic model interpretability library for PyTorch. *arXiv preprint arXiv:2009.07896*, 2020.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 25, 2012.

Emanuele La Malfa, Agnieszka Zbrzezny, Rhiannon Michelmore, Nicola Paoletti, and Marta Kwiatkowska. On guaranteed optimal robust explanations for NLP models. *Proceeding of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2021.

Himabindu Lakkaraju, Stephen H. Bach, and Jure Leskovec. Interpretable decision sets: A joint framework for description and prediction. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1675–1684, 2016.

Orestis Lampridis, Riccardo Guidotti, and Salvatore Ruggieri. Explaining sentiment classification with synthetic exemplars and counter-exemplars. In *International Conference on Discovery Science*, pages 357–373. Springer, 2020.

Marguerita Lane and Anne Saint-Martin. The impact of Artificial Intelligence on the labour market: What do we know so far? *OECD Social, Employment and Migration Working Papers*, 2021.

Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International Conference on Machine Learning (ICML)*, pages 1188–1196. PMLR, 2014.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Min Kyung Lee and Katherine Rich. Who is included in human perceptions of AI?: Trust and perceived fairness around healthcare AI and cultural mistrust. In *Proceedings of the 2021 CHI conference on human factors in computing systems*, pages 1–14, 2021.

Tobias Leemann, Michael Kirchhof, Yao Rong, Enkelejda Kasneci, and Gjergji Kasneci. When are post-hoc conceptual explanations identifiable? In *Uncertainty in Artificial Intelligence*, pages 1207–1218. PMLR, 2023.

Tobias Leemann, Alina Fastowski, Felix Pfeiffer, and Gjergji Kasneci. Attention Mechanisms Don't Learn Additive Models: Rethinking Feature Importance for Transformers. *arXiv preprint arXiv:2405.13536*, 2024a.

Tobias Leemann, Martin Pawelczyk, Bardh Prenkaj, and Gjergji Kasneci. Towards Non-adversarial Algorithmic Recourse. In *World Conference on Explainable Artificial Intelligence*, pages 395–419. Springer, 2024b.

Benjamin Letham, Cynthia Rudin, Tyler H McCormick, and David Madigan. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics*, pages 1350–1371, 2015.

Robert Lewis, Yuanbo Liu, Matthew Groh, and Rosalind Picard. Shaping Habit Formation Insights with Shapley Values: Towards an Explainable AI-system for Self-understanding and Health Behavior Change. *Proc. Realizing AI Healthcare, Challenges Appearing Wild CHI*, 2021.

Jian-hua Li. Cyber security meets artificial intelligence: a survey. *Frontiers of Information Technology & Electronic Engineering*, 19(12):1462–1474, 2018.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. Visualizing and Understanding Neural Models in NLP. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 681–691, 2016.

Yu Li, Chao Huang, Lizhong Ding, Zhongxiao Li, Yijie Pan, and Xin Gao. Deep learning in bioinformatics: Introduction, application, and perspective in the big data era. *Methods*, 166: 4–21, 2019.

Yuchen Li, Yuanzhi Li, and Andrej Risteski. How do transformers learn topic structure: Towards a mechanistic understanding. In *International Conference on Machine Learning (ICML)*, pages 19689–19729. PMLR, 2023.

Brian Y. Lim, Anind K. Dey, and Daniel Avrahami. Why and why not explanations improve the intelligibility of context-aware intelligent systems. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 2119–2128, 2009.

Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. Explainable AI: A Review of Machine Learning Interpretability Methods. *Entropy*, 23(1):18, 2021.

Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*, 2019.

Yixin Liu, Kai Zhang, Yuan Li, Zhiling Yan, Chujie Gao, Ruoxi Chen, Zhengqing Yuan, Yue Huang, Hanchi Sun, Jianfeng Gao, et al. Sora: A Review on Background, Technology, Limitations, and Opportunities of Large Vision Models. *arXiv preprint arXiv:2402.17177*, 2024.

Shayne Longpre, Marcus Storm, and Rishi Shah. Lethal autonomous weapons systems & artificial intelligence: Trends, challenges, and policies. *MIT Science Policy Review*, 2022. 10.38105/spr.360apm5typ.

Gianluigi Lopardo. Explainable AI for business decision-making. Master's thesis, Politecnico di Torino, 2021.

Gianluigi Lopardo and Damien Garreau. Comparing Feature Importance and Rule Extraction for Interpretability on Text Data. In *2-nd Workshop on Explainable and Ethical AI-26TH International Conference on Pattern Recognition (XAIE@ ICPR) 2022*, 2022.

Gianluigi Lopardo, Damien Garreau, Frédéric Precioso, and Greger Ottosson. SMACE: A New Method for the Interpretability of Composite Decision Systems. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 325–339. Springer, 2022.

Gianluigi Lopardo, Frederic Precioso, and Damien Garreau. A Sea of Words: An In-Depth Analysis of Anchors for Text Data. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2023a.

Gianluigi Lopardo, Frederic Precioso, and Damien Garreau. Faithful and Robust Local Interpretability for Textual Predictions. *arXiv preprint arXiv:2311.01605*, 2023b.

Gianluigi Lopardo, Frederic Precioso, and Damien Garreau. Attention Meets Post-hoc Interpretability: A Mathematical Perspective. In *International Conference on Machine Learning (ICML)*. PMLR, 2024.

Hans Peter Luhn. A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of research and development*, 1(4):309–317, 1957.

Scott M. Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems (NeurIPS)*, 30:4765–4774, 2017.

Scott M. Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M. Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, 2(1):2522–5839, 2020.

Ronny Luss, Pin-Yu Chen, Amit Dhurandhar, Prasanna Sattigeri, Yunfeng Zhang, Karthikeyan Shanmugam, and Chun-Chen Tu. Leveraging latent features for local explanations. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 1139–1149, 2021.

Shuai Ma, Ying Lei, Xinru Wang, Chengbo Zheng, Chuhan Shi, Ming Yin, and Xiaojuan Ma. Who should I trust: AI or myself? Leveraging human and AI correctness likelihood to promote appropriate trust in AI-assisted decision-making. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–19, 2023.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150. Association for Computational Linguistics, 2011.

Florence Jessie MacWilliams and Neil James Alexander Sloane. *The theory of error correcting codes*, volume 16. Elsevier, 1977.

Ashok Vardhan Makkuva, Marco Bondaschi, Adway Girish, Alliot Nagle, Martin Jaggi, Hyeji Kim, and Michael Gastpar. Attention with Markov: A Framework for Principled Analysis of Transformers via Markov Chains. *arXiv preprint arXiv:2402.04161*, 2024.

Karl Manheim and Lyric Kaplan. Artificial intelligence: Risks to privacy and democracy. *Yale JL & Tech.*, 21:106, 2019.

Dina Mardaoui and Damien Garreau. An analysis of LIME for text data. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 3493–3501. PMLR, 2021.

Vincent Margot and George Luta. A new method to compare the interpretability of rule-based algorithms. *AI*, 2(4):621–635, 2021.

Joao Marques-Silva and Xuanxiang Huang. Explainability is NOT a Game. *arXiv preprint arXiv:2307.07514*, 2023.

Joao Marques-Silva and Alexey Ignatiev. Delivering trustworthy AI through formal XAI. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(11):12342–12350, 2022.

Chris Marsden and Trisha Meyer. *Regulating disinformation with artificial intelligence: effects of disinformation initiatives on freedom of expression and media pluralism*. European Parliament, 2019.

David Martens and Foster Provost. Explaining data-driven document classifications. *MIS quarterly*, 38(1):73–100, 2014.

Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *ACM computing surveys (CSUR)*, 54(6):1–35, 2021.

Joseph Meixner. Orthogonale Polynomsysteme mit einer besonderen Gestalt der erzeugenden Funktion. *Journal of the London Mathematical Society*, 1(1):6–13, 1934.

Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 746–751, 2013.

Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267:1–38, 2019.

Riccardo Miotto, Fei Wang, Shuang Wang, Xiaoqian Jiang, and Joel T. Dudley. Deep learning for healthcare: review, opportunities and challenges. *Briefings in bioinformatics*, 19(6):1236–1246, 2018.

Yisroel Mirsky and Wenke Lee. The creation and detection of deepfakes: A survey. *ACM computing surveys (CSUR)*, 54(1):1–41, 2021.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *NIPS Deep Learning Workshop*, 2013.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, , et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Christoph Molnar. Interpretable machine learning, 2020.

Forrest E. Morgan, Benjamin Boudreaux, Andrew J. Lohn, Mark Ashby, Christian Curriden, Kelly Klima, and Derek Grossman. Military applications of artificial intelligence. *Santa Monica: RAND Corporation*, 2020.

Ramaravind K. Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 607–617, 2020.

Mohammed Bany Muhammad and Mohammed Yeasin. Eigen-CAM: Class activation map using principal components. In *2020 international joint conference on neural networks (IJCNN)*, pages 1–7. IEEE, 2020.

Blake Murdoch. Privacy and artificial intelligence: challenges for protecting health information in a new era. *BMC Medical Ethics*, 22:1–5, 2021.

Nikolaos Mylonas, Ioannis Mollas, and Grigorios Tsoumakas. An attention matrix for every decision: Faithfulness-based arbitration among multiple attention-based interpretations of transformers in text classification. *Data Mining and Knowledge Discovery*, pages 1–26, 2023.

Nikolaos Mylonas, Ioannis Mollas, and Grigorios Tsoumakas. An attention matrix for every decision: Faithfulness-based arbitration among multiple attention-based interpretations of transformers in text classification. *Data Mining and Knowledge Discovery*, 38(1):128–153, 2024.

Luca Nannini, Marta Marchiori Manerba, and Isacco Beretta. Mapping the landscape of ethical considerations in explainable AI research. *Ethics and Information Technology*, 26, 2024.

Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, et al. Efficient large-scale language model training on GPU clusters using megatron-LM. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–15, 2021.

Meike Nauta, Ron Van Bree, and Christin Seifert. Neural prototype trees for interpretable fine-grained image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14933–14943, 2021.

Noella Nazareth and Yeruva Venkata Ramana Reddy. Financial applications of machine learning: A literature review. *Expert Systems with Applications*, 219:119640, 2023.

Michael Neely, Stefan F. Schouten, Maurits J. R. Bleeker, and Ana Lucic. Order in the court: Explainable AI methods prone to disagreement. *arXiv preprint arXiv:2105.03287*, 2021.

Michael Neely, Stefan F. Schouten, Maurits Bleeker, and Ana Lucic. A Song of (Dis)agreement: Evaluating the Evaluation of Explainable Artificial Intelligence in Natural Language Processing. In *HHAI2022: Augmenting Human Intellect*, pages 60–78. IOS Press, 2022.

Dong Nguyen. Comparing automatic and human evaluation of local explanations for text classification. In *16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1069–1078. Association for Computational Linguistics, 2018.

Harsha Nori, Samuel Jenkins, Paul Koch, and Rich Caruana. InterpretML: A Unified Framework for Machine Learning Interpretability. *arXiv preprint arXiv:1909.09223*, 2019.

Jonas Oppenlaender. The creativity of text-to-image generation. In *Proceedings of the 25th International Academic Mindtrek Conference*, pages 192–202, 2022.

Cecilia Panigutti, Ronan Hamon, Isabelle Hupont, David Fernandez Llorca, Delia Fano Yela, Henrik Junklewitz, Salvatore Scalzo, Gabriele Mazzini, Ignacio Sanchez, Josep Soler Garrido, et al. The role of explainable AI in the context of the AI Act. In *Proceedings of the 2023 ACM conference on fairness, accountability, and transparency*, pages 1139–1150, 2023.

David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. Carbon Emissions and Large Neural Network Training. *arXiv preprint arXiv:2104.10350*, 2021.

Martin Pawelczyk. *On the Generation of Realistic and Robust Counterfactual Explanations for Algorithmic Recourse*. PhD thesis, Universität Tübingen, 2024.

Martin Pawelczyk, Klaus Broelemann, and Gjergji Kasneci. On counterfactual explanations under predictive multiplicity. In *Conference on Uncertainty in Artificial Intelligence*, pages 809–818. PMLR, 2020a.

Martin Pawelczyk, Klaus Broelemann, and Gjergji Kasneci. Learning model-agnostic counterfactual explanations for tabular data. In *Proceedings of The Web Conference 2020*, pages 3126–3132, 2020b.

Martin Pawelczyk, Sascha Bielawski, Johan Van den Heuvel, Tobias Richter, and Gjergji Kasneci. CARLA: A Python Library to Benchmark Algorithmic Recourse and Counterfactual Explanation Algorithms. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

Martin Pawelczyk, Chirag Agarwal, Shalmali Joshi, Sohini Upadhyay, and Himabindu Lakkaraju. Exploring counterfactual explanations through the lens of adversarial examples: A theoretical and empirical analysis. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 4574–4594. PMLR, 2022a.

Martin Pawelczyk, Tobias Leemann, Asia Biega, and Gjergji Kasneci. On the trade-off between actionable explanations and the right to be forgotten. In *International Conference on Learning Representations (ICLR)*, 2022b.

Martin Pawelczyk, Himabindu Lakkaraju, and Seth Neel. On the privacy risks of algorithmic recourse. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 9680–9696. PMLR, 2023.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Billy Perrigo. Exclusive: OpenAI Used Kenyan Workers on Less Than $2 Per Hour to Make ChatGPT Less Toxic. *TIME*, January 18 2023.

Slav Petrov, Dipanjan Das, and Ryan McDonald. A Universal Part-of-Speech Tagset. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2089–2096, 2012.

Vitali Petsiuk, Abir Das, and Kate Saenko. RISE: Randomized Input Sampling for Explanation of Black-box Models. *arXiv preprint arXiv:1806.07421*, 2018.

Mary Phuong and Marcus Hutter. Formal Algorithms for Transformers. *arXiv preprint arXiv:2207.09238*, 2022.

Gregory Plumb, Denali Molitor, and Ameet S Talwalkar. Model agnostic supervised local explanations. *Advances in Neural Information Processing Systems (NeurIPS)*, 31, 2018.

Nina Poerner, Hinrich Schütze, and Benjamin Roth. Evaluating neural network explanation methods using hybrid documents and morphosyntactic agreement. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 340–350, 2018.

Eleonora Poeta, Gabriele Ciravegna, Eliana Pastor, Tania Cerquitelli, and Elena Baralis. Concept-based explainable artificial intelligence: A survey. *arXiv preprint arXiv:2312.12936*, 2023.

Rafael Poyiadzi, Kacper Sokol, Raul Santos-Rodriguez, Tijl De Bie, and Peter Flach. FACE: feasible and actionable counterfactual explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 344–350, 2020.

Isha Puri, Amit Dhurandhar, Tejaswini Pedapati, Karthikeyan Shanmugam, Dennis Wei, and Kush R Varshney. CoFrNets: interpretable neural architecture inspired by continued fractions. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:21668–21680, 2021.

Luyu Qiu, Yi Yang, Caleb Chen Cao, Jing Liu, Yueyuan Zheng, Hilary Hei Ting Ngai, Janet Hsiao, and Lei Chen. Resisting Out-of-Distribution Data Problem in Perturbation of XAI. *arXiv preprint arXiv:2107.14000*, 2021.

Owen Queen, Tom Hartvigsen, Teddy Koker, Huan He, Theodoros Tsiligkaridis, and Marinka Zitnik. Encoding time-series explanations through self-supervised model behavior consistency. *Advances in Neural Information Processing Systems (NeurIPS)*, 36, 2024.

Harish Guruprasad Ramaswamy, , et al. Ablation-CAM: Visual explanations for deep convolutional network via gradient-free localization. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 983–991, 2020.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why should I trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Mattia Rigotti, Christoph Miksovic, Ioana Giurgiu, Thomas Gschwind, and Paolo Scotton. Attention-based interpretability with concept transformers. In *International Conference on Learning Representations (ICLR)*, 2021.

Rockafellar. *Convex analysis*. Princeton university press, 1997.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, 2022.

Yao Rong, Tobias Leemann, Thai-Trang Nguyen, Lisa Fiedler, Peizhu Qian, Vaibhav Unhelkar, Tina Seidel, Gjergji Kasneci, and Enkelejda Kasneci. Towards human-centered explainable ai: A survey of user studies for model explanations. *IEEE transactions on pattern analysis and machine intelligence*, 2023.

Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958a.

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958b.

Andrew Slavin Ross, Michael C Hughes, and Finale Doshi-Velez. Right for the right reasons: training differentiable models by constraining their explanations. In *Proceeding of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2662–2670, 2017.

Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *International Conference on Learning Representations (ICLR)*, 2020.

Haşim Sak, Andrew Senior, and Françoise Beaufays. Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition. *arXiv preprint arXiv:1402.1128*, 2014.

Michele Salvagno, Fabio Silvio Taccone, and Alberto Giovanni Gerli. Can artificial intelligence help for scientific writing? *Critical care*, 27(1):75, 2023.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *5th Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS 2019*, 2019.

Adriel Saporta, Xiaotong Gui, Ashwin Agrawal, Anuj Pareek, Steven QH Truong, Chanh DT Nguyen, Van-Doan Ngo, Jayne Seekins, Francis G Blankenberg, Andrew Y Ng, et al. Benchmarking saliency methods for chest X-ray interpretation. *Nature Machine Intelligence*, 4(10): 867–878, 2022.

Marcel Paul Schützenberger. On context-free languages and push-down automata. *Information and control*, 6(3):246–264, 1963.

Patrick Schwab and Walter Karlen. Cxplain: Causal explanations for model interpretation under uncertainty. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019.

Reva Schwartz, Reva Schwartz, Apostol Vassilev, Kristen Greene, Lori Perine, Andrew Burt, and Patrick Hall. *Towards a standard for identifying and managing bias in artificial intelligence*, volume 3. US Department of Commerce, National Institute of Standards and Technology, 2022.

Andrew D. Selbst and Solon Barocas. The intuitive appeal of explainable machines. *Fordham L. Rev.*, 87:1085, 2018.

Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 618–626, 2017.

Hugo Henri Joseph Senetaire, Damien Garreau, Jes Frellsen, and Pierre-Alexandre Mattei. Explainability as statistical inference. In *International Conference on Machine Learning (ICML)*, pages 30584–30612. PMLR, 2023.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, 2016.

Sofia Serrano and Noah A Smith. Is Attention Interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, 2019.

Mattia Setzu, Riccardo Guidotti, Anna Monreale, and Franco Turini. Global explanations with local scoring. In *Machine Learning and Knowledge Discovery in Databases: International Workshops of ECML PKDD 2019, Würzburg, Germany, September 16–20, 2019, Proceedings, Part I*, pages 159–171. Springer, 2020.

Mattia Setzu, Riccardo Guidotti, Anna Monreale, Franco Turini, Dino Pedreschi, and Fosca Giannotti. Glocalx-from local to global explanations of black box AI models. *Artificial Intelligence*, 294:103457, 2021.

Sharath M. Shankaranarayana and Davor Runje. ALIME: Autoencoder based approach for local interpretability. In *Intelligent Data Engineering and Automated Learning–IDEAL 2019: 20th International Conference, Manchester, UK, November 14–16, 2019, Proceedings, Part I 20*, pages 454–463. Springer, 2019.

Claude E Shannon. The redundancy of English. In *Cybernetics; Transactions of the 7th Conference, New York: Josiah Macy, Jr. Foundation*, pages 248–272, 1951.

Lloyd S. Shapley. A value for $n$-person games. *Contributions to the Theory of Games, number 28 in Annals of Mathematics Studies, pages 307–317*, II, 1953.

Irina G. Shevtsova. An improvement of convergence rate estimates in the Lyapunov theorem. *Doklady Mathematics*, 82(3):862–864, 2010.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International Conference on Machine Learning (ICML)*, pages 3145–3153. PMLR, 2017.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *nature*, 529(7587): 484–489, 2016.

K Simonyan, A Vedaldi, and A Zisserman. Deep inside convolutional networks: visualising image classification models and saliency maps. In *Proceedings of the International Conference on Learning Representations (ICLR)*. ICLR, 2014.

Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling LIME and SHAP: Adversarial attacks on post hoc explanation methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 180–186, 2020.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. SmoothGrad: removing noise by adding noise. *ICML Workshop on Visualization for Deep Learning*, 2017.

Gregor Stiglic, Primoz Kocbek, Nino Fijacko, Marinka Zitnik, Katrien Verbert, and Leona Cilar. Interpretability of machine learning-based prediction models in healthcare. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(5):e1379, 2020.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and Policy Considerations for Modern Deep Learning Research. *Proceedings of the AAAI conference on artificial intelligence*, 34(09):13693–13696, 2020.

Simone Stumpf, Vidya Rajaram, Lida Li, Margaret Burnett, Thomas Dietterich, Erin Sullivan, Russell Drummond, and Jonathan Herlocker. Toward harnessing user feedback for machine learning. In *Proceedings of the 12th international conference on Intelligent user interfaces*, pages 82–91, 2007.

Pascal Sturmfels, Scott Lundberg, and Su-In Lee. Visualizing the impact of feature attribution baselines. *Distill*, 5(1):e22, 2020.

Kaiser Sun and Ana Marasović. Effective Attention Sheds Light On Interpretability. In *Findings of the Association for Computational Linguistics (ACL-IJCNLP)*, pages 4126–4135, 2021.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning (ICML)*, pages 3319–3328. PMLR, 2017a.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning (ICML)*, pages 3319–3328. PMLR, 2017b.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, 2016.

Magamed Taimeskhanov, Ronan Sicre, and Damien Garreau. CAM-Based Methods Can See through Walls. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2024.

A Tallón-Ballesteros and C Chen. Explainable AI: Using Shapley value to explain complex anomaly detection ML-based systems. *Machine learning and artificial intelligence*, 332:152, 2020.

Davoud Ataee Tarzanagh, Yingcong Li, Christos Thrampoulidis, and Samet Oymak. Transformers as Support Vector Machines. In *NeurIPS 2023 Workshop on Mathematics of Modern Machine Learning*, 2023.

Neil Thompson, Kristjan Greenewald, Keeheon Lee, and Gabriel F. Manso. The Computational Limits of Deep Learning. In *Ninth Computing within Limits*. LIMITS, 2023.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. Generating Token-Level Explanations for Natural Language Inference. In *NAACL 2019*, 2019.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996.

Suzanne Tolmeijer, Markus Christen, Serhiy Kandul, Markus Kneer, and Abraham Bernstein. Capable but amoral? Comparing AI and human expert collaboration in ethical decision making. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pages 1–17, 2022.

Davy van de Sande, Michel E van Genderen, Joost Huiskens, Diederik Gommers, and Jasper van Bommel. Moving from bytes to bedside: a systematic review on the use of artificial intelligence in the intensive care unit. *Intensive care medicine*, 47:750–760, 2021.

Vladimir Naumovich Vapnik. *Statistical learning theory*. wiley New York, 1998.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems (NeurIPS)*, 30, 2017.

Andrea Vedaldi and Stefano Soatto. Quick shift and kernel methods for mode seeking. In *Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part IV 10*, pages 705–718. Springer, 2008.

Alfredo Vellido. The importance of interpretability and visualization in machine learning for applications in medicine and health care. *Neural computing and applications*, 32(24):18069–18083, 2020.

Luisa Verdoliva. Media forensics and deepfakes: an overview. *IEEE Journal of Selected Topics in Signal Processing*, 14(5):910–932, 2020.

Himanshu Verma, Jakub Mlynar, Roger Schaer, Julien Reichenbach, Mario Jreige, John Prior, Florian Evéquoz, and Adrien Depeursinge. Rethinking the role of AI with physicians in oncology: revealing perspectives from clinical and research workflows. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–19, 2023.

Jesse Vig and Yonatan Belinkov. Analyzing the Structure of Attention in a Transformer Language Model. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 63–76, 2019.

Georgios Vlassopoulos, Tim van Erven, Henry Brighton, and Vlado Menkovski. Explaining Predictions by Approximating the Local Decision Boundary. *arXiv preprint arXiv:2006.07985*, 2020.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning (ICML)*, pages 35151–35174. PMLR, 2023.

Sandra Wachter, Brent Mittelstadt, and Luciano Floridi. Why a Right to Explanation of Automated Decision-Making Does Not Exist in the General Data Protection Regulation. *International data privacy law*, 7(2):76–99, 2017a.

Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harv. JL & Tech.*, 31:841, 2017b.

Eric Wallace, Shi Feng, and Jordan Boyd-Graber. Interpreting Neural Networks with Nearest Neighbors. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 136–144, 2018.

Bowen Wang, Liangzhi Li, Yuta Nakashima, and Hajime Nagahara. Learning bottleneck concepts in image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10962–10971, 2023.

Fulton Wang and Cynthia Rudin. Falling rule lists. In *Artificial intelligence and statistics*, pages 1013–1022. PMLR, 2015.

Haofan Wang, Zifan Wang, Mengnan Du, Fan Yang, Zijian Zhang, Sirui Ding, Piotr Mardziel, and Xia Hu. Score-CAM: Score-weighted visual explanations for convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24–25, 2020.

Gail Weiss, Yoav Goldberg, and Eran Yahav. Thinking like transformers. In *International Conference on Machine Learning (ICML)*, pages 11080–11090. PMLR, 2021.

Kaiyue Wen, Yuchen Li, Bingbin Liu, and Andrej Risteski. Transformers are uninterpretable with myopic methods: a case study with bounded Dyck grammars. *Advances in Neural Information Processing Systems (NeurIPS)*, 36, 2024.

Sarah Wiegreffe and Yuval Pinter. Attention is not not Explanation. In *2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 11–20. Association for Computational Linguistics, 2019.

Qian Yang, Yuexing Hao, Kexin Quan, Stephen Yang, Yiran Zhao, Volodymyr Kuleshov, and Fei Wang. Harnessing biomedical literature to calibrate clinicians' trust in AI decision support systems. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–14, 2023.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019.

Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On completeness-aware concept-based explanations in deep neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:20554–20565, 2020.

Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing. *IEEE Computational intelligence magazine*, 13(3): 55–75, 2018.

Muhammad Rehman Zafar and Naimul Mefraz Khan. DLIME: A Deterministic Local Interpretable Model-Agnostic Explanations Approach for Computer-Aided Diagnosis Systems. *arXiv preprint arXiv:1906.10263*, 2019.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pages 818–833. Springer, 2014.

Hanwei Zhang, Felipe Torres, Ronan Sicre, Yannis Avrithis, and Stephane Ayache. Opti-CAM: Optimizing saliency maps for interpretability. *arXiv preprint arXiv:2301.07002*, 2023a.

Qiaoning Zhang, Matthew L. Lee, and Scott Carter. You complete me: Human-AI teams and complementary expertise. In *Proceedings of the 2022 CHI conference on human factors in computing systems*, pages 1–28, 2022.

Ruihan Zhang, Prashan Madumal, Tim Miller, Krista A. Ehinger, and Benjamin IP Rubinstein. Invertible concept-based explanations for cnn models with non-negative concept activation vectors. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(13):11682–11690, 2021.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. Siren's Song in the AI Ocean: A Survey on Hallucination in Large Language Models. *arXiv preprint arXiv:2309.01219*, 2023b.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. A Survey of Large Language Models, 2023.

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2921–2929, 2016.

## Media References

Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine Bias. *ProPublica*, 2016. URL https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing.

Gretchen Bueermann and Natasa Perucica. How can we combat the worrying rise in deepfake content? *World Economic Forum*, May 2023. URL https://www.weforum.org/agenda/2023/05/how-can-we-combat-the-worrying-rise-in-deepfake-content/.

Jeffrey Dastin. Insight - Amazon scraps secret AI recruiting tool that showed bias against women. *Reuters*, 2018. URL https://www.reuters.com/article/idUSKCN1MK0AG/.

Attilio Di Battista, Sam Grayling, Else Hasselaar, T Leopold, Ricky Li, Mark Rayner, and Saadia Zahidi. Future of jobs report 2023. In *World Economic Forum, Geneva, Switzerland.*, 2023. URL https://www.weforum.org/reports/the-future-of-jobs-report-2023.

Jake Okechukwu Effoduh. Weapons powered by artificial intelligence need to be regulated. *World Economic Forum*, June 2021. URL https://www.weforum.org/agenda/2021/06/the-accelerating-development-of-weapons-powered-by-artificial-risk-is-a-risk-to-humanity/.

AI Epoch. Parameter, compute and data trends in machine learning. *Published online at https://epochai. org/data/notable-ai-models-documentation# inclusion*, 2024.

Epoch AI. Parameter, Compute and Data Trends in Machine Learning. *Epoch AI*, 2024. URL https://epochai.org/data/epochdb/visualization.

EU. General Data Protection Regulation (GDPR). *Official Journal of the European Union*, 679, 2016. URL https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32016R0679.

EU. Artificial Intelligence Act. *Official Journal of the European Union*, 2024. URL https://eur-lex.europa.eu/legal-content/en/TXT/HTML/?uri=OJ%3AL_202401689#d1e2918-1-1.

Europol. Facing Reality? Law Enforcement and the Challenge of Deepfakes. *Europol Innovation Lab*, 2022. URL https://www.europol.europa.eu/cms/sites/default/files/documents/Europol_Innovation_Lab_Facing_Reality_Law_Enforcement_And_The_Challenge_Of_Deepfakes.pdf.

Walter Frick. AI Is Making Economists Rethink the Story of Automation. *Harvard Business Review*, 2024. URL https://hbr.org/2024/05/ai-is-making-economists-rethink-the-story-of-automation.

Kristalina Georgieva. AI Will Transform the Global Economy. Let's Make Sure it Benefits Humanity. *International Monetary Fund (IMF) Blog*, 2024. URL https://www.imf.org/en/Blogs/Articles/2024/01/14/ai-will-transform-the-global-economy-lets-make-sure-it-benefits-humanity.

Michele Gilman. AI algorithms intended to root out welfare fraud often end up punishing the poor instead. *The Conversation*, Feb 2020. URL https://theconversation.com/ai-algorithms-intended-to-root-out-welfare-fraud-often-end-up-punishing-the-poor-instead-131625.

Karen Hao. Algorithms in the Criminal Justice System: The Pros and Cons of AI. *MIT Technology Review*, January 2019. URL https://www.technologyreview.com/2019/01/21/137783/algorithms-criminal-justice-ai/.

Matthew Kaminski. The Coming War Between DC and Silicon Valley. *POLITICO*, 2024. URL https://www.politico.com/news/magazine/2024/05/20/artificial-intelligence-tech-antitrust-dc-silicon-valley-00158479.

Douwe Kiela, Tristan Thrush, Kawin Ethayarajh, and Amanpreet Singh. Plotting Progress in AI. *Contextual AI Blog*, 2023. URL https://contextual.ai/blog/plotting-progress.

Bernard Marr. AI and Jobs: The Good and Bad News. *Forbes*, May 2024. URL https://www.forbes.com/sites/bernardmarr/2024/05/29/ai-and-jobs-the-good-and-bad-news/?sh=55bafe5872a3.

Cade Metz. A.I. Is Mastering Language. Should We Trust What It Says? *The New York Times*, 2022. URL https://www.nytimes.com/2022/04/15/magazine/ai-language.html.

Max Roser. The brief history of artificial intelligence: the world has changed fast — what might be next? *Our World in Data*, 2022. URL https://ourworldindata.org/brief-history-of-ai.

Ron Schmelzer. How AI is Shaping the Future of Education. *Forbes*, May 2024. URL https://www.forbes.com/sites/ronschmelzer/2024/05/28/how-ai-is-shaping-the-future-of-education/.

Miazia Schueler, Salvatore Romano, Natalia Stanusch, Raziye Buse Çetin, Sonia Tabti, Marc Faddoul, and Ibis Lilley. Artificial Elections: Exposing the Use of Generative AI Imagery in the Political Campaigns of the 2024 French Elections. *AI Forensics*, 2024. URL https://aiforensics.org/uploads/Report_Artificial_Elections_81d14977e9.pdf.

Craig S. Smith. A.I. Here, There, Everywhere. *The New York Times*, 2021. URL https://www.nytimes.com/2021/02/23/technology/ai-innovation-privacy-seniors-education.html.

The Economist. What will artificial intelligence mean for your pay? *The Economist*, 2023. URL https://www.economist.com/finance-and-economics/2023/11/16/what-will-artificial-intelligence-mean-for-your-pay.

The Economist. What happened to the artificial-intelligence investment boom? *The Economist*, 2024a. URL https://www.economist.com/finance-and-economics/2024/01/07/what-happened-to-the-artificial-intelligence-investment-boom.

The Economist. Researchers are Figuring Out How Large Language Models Work. *The Economist*, 2024b. URL https://www.economist.com/science-and-technology/2024/07/11/researchers-are-figuring-out-how-large-language-models-work.

TIME. 2023 PERSON OF THE YEAR: Taylor Swift. *TIME*, 2023. URL https://time.com/6342806/person-of-the-year-2023-taylor-swift.

UC Berkeley. What is Machine Learning? *UC Berkeley School of Information Blog*, 2020. URL https://ischoolonline.berkeley.edu/blog/what-is-machine-learning/.

Rhiannon Williams. AI systems are getting better at tricking us. *MIT Technology Review*, May 10 2024. URL https://www.technologyreview.com/2024/05/10/1092293/ai-systems-are-getting-better-at-tricking-us/.

# Appendix

# APPENDIX A

# Appendix for Chapter 4: *An In-Depth Analysis of Anchors for Text Data*

**Organization of the Appendix.** The proofs of all results presented in Chapter 4 are provided in Section A.1. Specifically, Section A.1.9 demonstrates the validity of all findings for a normalized TF-IDF vectorization. Section A.2 compiles all required technical results. Additionally, Section A.3 contains further experiments for empirical validation of the findings.

## A.1 Proofs

In this section, all proofs omitted from the Chapter 4 are collected. Section A.1.1 contains the proof of Proposition 4.2.1. Sections A.1.2 and A.1.3 refer to Section 4.3. Sections A.1.4 to A.1.7 contain the proofs for Section 4.4. Section A.1.8 provides an additional result to those presented in Section 4.4.2. Section A.1.9 proves that the statements for TF-IDF vectorization remain valid for a normalized TF-IDF vectorization.

In all proofs where there is only one tolerance threshold $\varepsilon$ and where the selection function $\mathrm{p}$ is clear from context, $\mathcal{A}_k$ is written instead of $\mathcal{A}_k^{\mathrm{p}}(\varepsilon)$ for $k \in [3]$.

### A.1.1 Proof of Proposition 4.2.1: Equivalent sampling

Let $i \in [n]$ be fixed and let $R \subseteq [b]$ be the (random) set of replaced indices for this specific perturbed example. Let us first compute the probability that a given word with index $k \in [b] \setminus A$ is removed:

$$\begin{aligned}
\mathbb{P}(k \in R) &= \mathbb{P}(i \in R_k) && \text{(definition of } R_k\text{)} \\
&= \sum_{\ell=0}^{n} \mathbb{P}(i \in R_k \mid B_k = \ell) \cdot \mathbb{P}(B_k = \ell) && \text{(law of total probability)} \\
&= \sum_{\ell=0}^{n} \binom{n-1}{\ell-1} \binom{n}{\ell}^{-1} \cdot \binom{n}{\ell} \frac{1}{2^n} = \frac{2^{n-1}}{2^n} && \text{(uniform distribution among all subsets)}
\end{aligned}$$

$$\mathbb{P}(k \in R) = \frac{1}{2}\,.$$

The independence of the removals is now demonstrated. First, the column-wise independence is verified by definition: for each word with index $k \in [b]$, the number $B_k$ of copies to perturb is drawn independently by construction. Next, for a given column $k \in [b]$, the removals from example to example are shown to be independent.

Let $i, j \in [n]$, with $i \neq j$. Write

$$\begin{aligned}
\mathbb{P}\left(i, j \in R_k\right) &= \sum_{\ell=0}^{n} \mathbb{P}\left(i, j \in R_k \mid B_k = \ell\right) \mathbb{P}\left(B_k = \ell\right) && \text{(law of total probability)} \\
&= \sum_{\ell=0}^{n} \binom{n-2}{\ell-2} \binom{n}{\ell}^{-1} \binom{n}{\ell} \frac{1}{2^n} = \frac{2^{n-2}}{2^n} && \text{(uniform distribution among all subsets)} \\
\mathbb{P}\left(i, j \in R_k\right) &= \frac{1}{4}\,.
\end{aligned}$$

According to the first part of the proof, this is exactly $\mathbb{P}\left(i \in R_k\right) \cdot \mathbb{P}\left(j \in R_k\right)$, allowing us to conclude. □

### A.1.2 Proof of Proposition 4.3.1: Stability of exhaustive p-Anchors

Recall that $A^\star$ is set as $A^{\mathrm{p}}(\varepsilon)$. First, it is shown that $\mathcal{A}_1^{\mathrm{q}}(\varepsilon - \delta) \subseteq \mathcal{A}_1^{\mathrm{p}}(\varepsilon)$. Let $A \in \mathcal{A}_1^{\mathrm{q}}(\varepsilon - \delta)$. Using Eq. (4.4), the following is obtained:

$$\mathrm{p}(A) \geq \mathrm{q}(A) - \delta \geq 1 - (\varepsilon - \delta) - \delta = 1 - \varepsilon\,.$$

Therefore, $A \in \mathcal{A}_1^{\mathrm{p}}(\varepsilon)$. It is directly deduced that $\mathcal{A}_2^{\mathrm{q}}(\varepsilon - \delta) \subseteq \mathcal{A}_2^{\mathrm{p}}(\varepsilon)$. Next, it is shown that $\mathcal{A}_2^{\mathrm{q}}(\varepsilon - \delta)$ is non-empty, specifically containing $A^\star$. Indeed,

$$\mathrm{q}(A^\star) \geq \mathrm{p}(A^\star) - \delta \geq 1 - \varepsilon/4 - \delta \geq 1 - (\varepsilon - \delta)\,,$$

since $\delta < \varepsilon/4 < 3\varepsilon/8$. At this point, it suffices to show that $A^\star$ has (strict) maximal $q$ among $\mathcal{A}_2^{\mathrm{q}}(\varepsilon - \delta)$. Let us pick any $A \in \mathcal{A}_2^{\mathrm{q}}(\varepsilon - \delta)$ such that $A \neq A^\star$. Then,

$$\begin{aligned}
\mathrm{q}(A^\star) - \mathrm{q}(A) &= \mathrm{q}(A^\star) - \mathrm{p}(A^\star) + \mathrm{p}(A^\star) - \mathrm{p}(A) + \mathrm{p}(A) - \mathrm{q}(A) \\
&\geq -\delta + 1 - \varepsilon/4 - (1 - 3\varepsilon/4) - \delta \geq \varepsilon/4 > 0\,,
\end{aligned}$$

since $\delta < \varepsilon/4$. Finally, there is no uniform random draw (last step of Algorithm 1), since $\mathcal{A}_3^{\mathrm{p}}(\varepsilon) = \mathcal{A}_3^{\mathrm{q}}(\varepsilon - \delta) = \{A^\star\}$. □

### A.1.3 Proof of Proposition 4.3.2: $\widehat{\mathrm{Prec}}_n(A)$ uniformly approximates $\mathrm{Prec}$

Let $A \in \mathcal{A}$ be any anchor and let $x^{(1)}, \ldots, x^{(n)}$ be perturbed examples associated to this anchor. For all $i \in [n]$, the random variables $Y_i := \mathbb{1}_{f(x^{(i)})=1} \in \{0, 1\}$ are independent and bounded by construction. Since $\mathrm{Prec}\left(A\right) = \mathbb{E}\left[\widehat{\mathrm{Prec}}_n(A)\right]$, we can apply Hoeffding's inequality to the $Y_i$s [Boucheron et al., 2013, Theorem 2.8]. We obtain

$$\mathbb{P}\left(\left|\widehat{\mathrm{Prec}}_n(A) - \mathrm{Prec}\left(A\right)\right| > \delta\right) = \mathbb{P}\left(n\left|\widehat{\mathrm{Prec}}_n(A) - \mathrm{Prec}\left(A\right)\right| > n\delta\right) \leq 2\mathrm{e}^{-2n\delta^2}\,. \tag{A.1}$$

There are less than $2^b$ anchors (since we are not considering the empty anchor as a valid anchor), and we can conclude *via* a union bound argument. □

### A.1.4 Proof of Proposition 4.4.1: Dummy features

Let $A \in \mathcal{A}_3^{\text{Prec}}(\varepsilon)$. If $a_j = 0$, there is nothing to prove. Thus let us assume that $a_j > 0$ and come to a contradiction. Let us set $A^j$ the anchor identical to $A$ but with coordinate $j$ to zero. The precision of $A$ is given by

$$\text{Prec}\,(A) = \mathbb{P}_A\left(g(\varphi(x)) \in R\right) = \mathbb{P}_A\left(g(\ldots, \varphi(x)_j, \ldots) \in R\right) .$$

According to the discussion preceding Proposition 4.4.1, $\varphi(x)_j = M_j v_j$, where $M_j \sim a_j + B_j$, with $B_j \sim \text{Bin}(m_j - a_j, 1/2)$ (this is Eq. (4.6)). Since $g$ does not depend on coordinate $j$ and the sampling is independent, $g(\ldots, v_j(a_j + B_j), \ldots)$ is equal in distribution to $g(\ldots, v_j \text{Bin}(m_j, 1/2), \ldots)$. In particular, $\text{Prec}\,(A^j) = \text{Prec}\,(A)$. Since $|A^j| < |A|$, we can conclude. □

### A.1.5 Proof of Proposition 4.4.2: Presence of a set of words

Let $p_i := 1 - \frac{1}{2^{m_i}}$. Since the model only depends on the coordinates belonging to $J$, according to Proposition 4.4.1, we can restrict ourselves to anchors such that $a_j \neq 0$ if $j \in J$. Let us start by computing the precision of any candidate anchor $A \in \mathcal{A}$. We write

$$\begin{aligned}
\text{Prec}\,(A) &= \mathbb{E}_A\left[\mathbb{1}_{f(x)=f(\xi)}\right] \\
&= \mathbb{P}_A\left(f(x) = 1\right) && \text{(since } f(\xi) = 1\text{)} \\
&= \mathbb{P}_A\left(w_j \in x,\, \forall j \in J\right) \\
&= \prod_{j \in J} \mathbb{P}_A\left(w_j \in x\right) && \text{(by independence)} \\
&= \prod_{j \in J} \mathbb{P}_A\left(v_j M_j > 0\right) \\
&= \prod_{j \in J} \mathbb{P}_A\left(a_j + B_j > 0\right) \\
\text{Prec}\,(A) &= \prod_{j \in J} p_j^{\mathbb{1}_{a_j = 0}} .
\end{aligned}$$

Let us now apply Algorithm 1 step by step in each of the cases outlined in the statement of the result.

**Case (I):** $\max_{j \in J} m_j \leq B$**.** If for all $j \in J,\ a_j > 0$, then, according to the previous discussion,

$$\text{Prec}\,(A) = \prod_{j \in J} p_j^{\mathbb{1}_{a_j = 0}} = 1 .$$

Therefore, the anchor $(1, \dots, 1)$ belongs to $\mathcal{A}_1$. If, instead, there exists $j \in J$ such that $a_j = 0$, then $\sum_{j \in J} \mathbb{1}_{a_j=0} \geq 1$ and

$$\begin{aligned}
\text{Prec}\,(A) &= \prod_{j \in J} p_j^{\mathbb{1}_{a_j=0}} \\
&\leq \prod_{j \in J} \left(1 - \frac{1}{2^B}\right)^{\mathbb{1}_{a_j=0}} && \text{(since } m_j \leq B \text{ for all } j \in J\text{)} \\
&= \left(1 - \frac{1}{2^B}\right)^{\sum_{j \in J} \mathbb{1}_{a_j=0}} && \text{(since } \textstyle\sum_{j \in J} \mathbb{1}_{a_j=0} \geq 1\text{)} \\
&\leq 1 - \frac{1}{2^B} \\
\text{Prec}\,(A) &< 1 - \varepsilon\,. && \text{(since } 1 - \tfrac{1}{2^B} \leq 1 - \varepsilon \text{ by definition of } B\text{)}
\end{aligned}$$

Thus $\mathcal{A}_1$ consists of anchors having at least one occurrence of each word of $J$, and these anchors only. In $\mathcal{A}_2$, Algorithm 1 will select the anchor $A_J = \{w_j,\ j \in J\}$, such that $a_j = 1$ for all $j \in J$ and $a_j = 0$ if $j \notin J$, which is the shortest anchor satisfying the precision condition. Since there are no equality cases, $\mathcal{A}_3$ is a singleton and we can conclude.

**Case (II):** $\max_{j \in J} m_j > B$**.** We first make two claims:

**Claim 1.** *We can restrict our analysis to anchors $A \in \mathcal{A}$ such that $a_j \in \{0, 1\}$ for $j \in J$.*

Indeed, for any $j \in J$, the model $f$ is checking the presence of the word $w_j$ in a document $\delta$, *i.e.*, that $\varphi(\delta)_j > 0$, disregarding of its multiplicity. As said before, the anchor $A_J$ (such that $a_j = 1$ for all $j \in J$) has precision 1. Any other anchor $A = (a_1, \dots, a_d)$ such that $a_i \geq 2$ has the same precision, but higher length.

Now let us consider two indices $j$ and $j'$ in $J$ such that $j > j'$ (implying, $p_j < p_{j'}$) and an anchor $A$ such that $a_j = a_{j'} = 0$. We set $A^j$ (resp. $A^{j'}$) the anchor identical to $A$ except coordinate $j$ (resp. $j'$) put to 1. Let $I_A := \{j \in [d],\ a_j > 0\}$.

**Claim 2.** *If $j > j'$,* $\text{Prec}\,(A^j) > \text{Prec}\left(A^{j'}\right)$.

Indeed,

$$\begin{aligned}
\text{Prec}\left(A^j\right) &= \prod_{\ell \in J,\, a_\ell = 0} p_\ell \\
&= \prod_{\ell \in J \setminus (I_A \cup \{j\})} p_\ell \\
&= p_{j'} \cdot \prod_{\ell \in J \setminus (I_A \cup \{j\} \cup \{j'\})} p_\ell \\
&> p_j \cdot \prod_{\ell \in J \setminus (I_A \cup \{j\} \cup \{j'\})} p_\ell && \text{(since } p_j < p_{j'}\text{)} \\
\text{Prec}\left(A^j\right) &= \text{Prec}\left(A^{j'}\right)\,.
\end{aligned}$$

As a consequence, for any anchor of fixed length, we can get higher precision by moving indices to the right. Therefore, the anchor $A_{c_0}$ will be selected by Algorithm 1, see Figure A.1 for an illustration. □

| $a_1$ | $a_2$ | $\dots$ | $a_{k-c_0}$ | $a_{k-c_0+1}$ | $a_{k-c_0+2}$ | $\dots$ | $a_{k-1}$ | $a_k$ | $\text{Prec}(A) = \prod_{\ell=1}^{k} p_\ell^{\mathbb{1}_{a_\ell=0}}$ |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | … | 0 | 0 | 0 | … | 0 | 0 | $\prod_{\ell=1}^{k} p_j < 1-\varepsilon$ |
| 0 | 0 | … | 0 | 0 | 0 | … | 0 | 1 | $\prod_{\ell=1}^{k-1} p_\ell < 1-\varepsilon$ |
| 0 | 0 | … | 0 | 0 | 0 | … | 1 | 1 | $\prod_{\ell=1}^{k-2} p_\ell < 1-\varepsilon$ |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | 1 | 1 | $\text{Prec}(A) < 1-\varepsilon$ |
| 0 | 0 | … | 0 | 0 | 1 | … | 1 | 1 | $\prod_{\ell=1}^{k-c_0+1} p_\ell < 1-\varepsilon$ |
| 0 | 0 | … | 0 | 1 | 1 | … | 1 | 1 | $\prod_{\ell=1}^{k-c_0} p_\ell \geq 1-\varepsilon$ |
| 0 | 0 | … | 1 | 1 | 1 | … | 1 | 1 | $\prod_{\ell=1}^{k-c_0-1} p_\ell > 1-\varepsilon$ |
| ⋮ | ⋮ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | $\text{Prec}(A) > 1-\varepsilon$ |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | $1 > 1-\varepsilon$ |

Figure A.1 – **Illustration of Proposition 4.4.2.** Anchors extraction for a model classifying words according to the presence or absence of words $w_j$, $j \in J = \{1, \dots, k\}$, ranked such that $m_1 > m_2 > \cdots > m_k$. The anchor such that $a_\ell = 1$ for $\ell \in [\![k - c_0 + 1, k]\!]$ and $a_\ell = 0$ otherwise, is the minimal anchor satisfying the precision condition.

### A.1.6 Proof of Proposition 4.4.3: Precision of a linear classifier

Let us set

$$Z_d := \lambda_0 + \sum_{j=1}^{d} \lambda_j v_j M_j \,,$$

where $M_j = a_j + B_j$ are the random multiplicities, that is, $a_j$ is the number of anchored words for $j$ and, as before, $B_j \sim \mathrm{Bin}(m_j - a_j, 1/2,)$. In our notation, the problem of evaluating the precision of an anchor $A$ is that of evaluating accurately $\mathbb{P}(Z_d > 0)$. From Section A.2.1, we see that, for all $j \in [d]$,

$$\mathbb{E}[M_j] = \frac{1}{2}(m_j + a_j)\,, \qquad \text{and} \qquad \mathrm{Var}(M_j) = \frac{1}{4}(m_j - a_j)\,.$$

We deduce that

$$\mathbb{E}[Z_d] = \frac{1}{2}\sum_{j=1}^{d} \lambda_j v_j (m_j + a_j) \quad \text{and} \quad \mathrm{Var}(Z_d) = \frac{1}{4}\sum_{j=1}^{d} \lambda_j^2 v_j^2 (m_j - a_j)\,. \tag{A.2}$$

By the Berry-Esseen theorem for non-identically distributed version [Shevtsova, 2010], uniformly in $t \in \mathbb{R}$, it holds that

$$\left| \mathbb{P}(Z_d \le t) - \Phi\left( \frac{t - \mathbb{E}[Z_d]}{\sqrt{\mathrm{Var}(Z_d)}} \right) \right| \le C \cdot \frac{\sum_{j=1}^{d} \mathbb{E}\left[ |\lambda_j v_j M_j - \mathbb{E}[\lambda_j v_j M_j]|^3 \right]}{\mathrm{Var}(Z_d)^{3/2}}\,, \tag{A.3}$$

where $C \approx 7.15$ is a numerical constant. Setting $t = -\lambda_0$ in the previous display, we recognize $1-$ the definition of the precision. Using Eq. (A.2), we have obtained the left-hand side of Eq. (4.9). The numerator is upper bounded by first writing

$$\begin{aligned}
\mathbb{E}\left[ |\lambda_j v_j M_j - \mathbb{E}[\lambda_j v_j M_j]|^3 \right] &\le \max_j |\lambda_j v_j|^3 \cdot \mathbb{E}\left[ |B_j - \mathbb{E}[B_j]|^3 \right] && \text{(definition of } M_j\text{)} \\
&\le \max_j |\lambda_j v_j|^3 \cdot \frac{1}{\sqrt{8\pi}} \left( \frac{m_j - a_j}{2} \right)^{3/2} && \text{(Eq. (A.11))} \\
\mathbb{E}\left[ |\lambda_j v_j M_j - \mathbb{E}[\lambda_j v_j M_j]|^3 \right] &\le \frac{1}{\sqrt{\pi}} \cdot \max_j |\lambda_j v_j|^3 \cdot (\max_j m_j)^{3/2}\,.
\end{aligned}$$

We deduce that

$$\sum_{j=1}^{d} \mathbb{E}\left[ |\lambda_j v_j M_j - \mathbb{E}[\lambda_j v_j M_j]|^3 \right] \le \frac{1}{\sqrt{\pi}} \cdot \left( \max_j \lambda_j^2 v_j^2 \right)^{3/2} \cdot (\max_j m_j)^{3/2} \cdot d\,. \tag{A.4}$$

Regarding the denominator, we have

$$\begin{aligned}
\mathrm{Var}(Z_d) &= \frac{1}{4}\sum_{j=1}^{d} \lambda_j^2 v_j^2 (m_j - a_j) && \text{(Eq. (A.2))} \\
&\ge \frac{1}{4} \cdot \left( \min_j \lambda_j^2 v_j^2 \right) \cdot (b - |A|) && \text{(definition of } b \text{ and } |A|\text{)} \\
\mathrm{Var}(Z_d) &\ge \frac{1}{8} \cdot \left( \min_j \lambda_j^2 v_j^2 \right) \cdot \min_j m_j \cdot d\,.
\end{aligned}$$

Thus

$$\mathrm{Var}\left(Z_d\right)^{3/2} \geq \frac{1}{16\sqrt{2}} \cdot \left(\min_j \lambda_j^2 v_j^2\right)^{3/2} \cdot \left(\min_j\right)^{3/2} \cdot d^{3/2}\,. \tag{A.5}$$

In particular, this is a positive quantity. Coming back to Eq. (A.3), we see that

$$\left|\mathbb{P}\left(Z_d \leq t\right) - \Phi\left(\frac{t - \mathbb{E}\left[Z_d\right]}{\sqrt{\mathrm{Var}\left(Z_d\right)}}\right)\right| \leq \frac{16C\sqrt{2}}{\sqrt{\pi}} \cdot \left(\frac{\max_j \lambda_j^2 v_j^2}{\min_j \lambda_j^2 v_j^2}\right)^{3/2} \cdot \left(\frac{\max_j m_j}{\min_j m_j}\right)^{3/2} \cdot \frac{1}{\sqrt{d}}\,,$$

as announced. □

### A.1.7 Proof of Proposition 4.4.4: Approximate precision maximization

We start by proving two lemmas. The first shows that we can restrict ourselves to $\mathcal{A}_+$ when considering the minimization of $L$.

**Lemma A.1.1 (Restriction to positive anchors).** *Let $A \in \mathcal{A}$ be such that $a_j > 0$ whereas $\lambda_j < 0$. Then*

$$L\left(\ldots, 0, \ldots\right) < L\left(\ldots, a_j, \ldots\right)\,.$$

*Proof.*

Keeping in mind that $\lambda_j > 0$, we notice that $-\lambda_j v_j (m_j + a_j) > -\lambda_j v_j m_j$, and that $\lambda_j^2 v_j^2 (m_j - a_j) < \lambda_j^2 v_j^2 m_j$. In other terms, removing the word $w_j$ from the anchor $A$ both decreases the numerator and increases the numerator of $L$.

□

The second shows that the minimization of $L$ on $\mathcal{A}_+$ is straightforward, modulo a technical assumption on the size of the intercept.

**Lemma A.1.2 (Minimization of $L$).** *Assume that $\lambda_0 > -\gamma/2$. Assume further that the indices of the local dictionary are ordered such that the $\lambda_j v_j$s are strictly decreasing. Then, for any $k < \ell$ such that $\lambda_k, \lambda_\ell > 0$,*

$$L\left(a_1, \ldots, a_k + 1, \ldots, a_\ell, \ldots, a_d\right) < L\left(a_1, \ldots, a_k, \ldots, a_\ell + 1, \ldots, a_d\right)\,. \tag{A.6}$$

*Proof.*

First, since $A \in \mathcal{A}_+$ and $\lambda_0 \geq -\gamma/2$, we deduce that

$$\lambda_0 + \frac{1}{2}\sum_j \lambda_j v_j m_j + \frac{1}{2}\sum_j \lambda_j v_j a_j \geq 0\,. \tag{A.7}$$

Now, we will prove both inequalities by a function study. For any $j \in [d]$, let us set $\alpha_j := \lambda_j v_j$ . We also set $\Omega_1 := \sum_j \alpha_j (m_j - a_j)$, and $\Omega_2 := \sum_j \alpha_j^2 (m_j - a_j)$. Further, for any $a, b \in \mathbb{R}$, let us define the mapping

$$f_{a,b}(t) := \frac{a + t}{\sqrt{b - t^2}}\,.$$

With this notation in hand, Eq. (A.6) becomes

$$\frac{\gamma - \frac{1}{2}(\Omega_1 - \alpha_k)}{\sqrt{\Omega_2 - \alpha_k^2}} > \frac{\gamma - \frac{1}{2}(\Omega_1 - \alpha_\ell)}{\sqrt{\Omega_2 - \alpha_\ell^2}}$$

which is simply

$$f_{2\gamma-\Omega_1,\Omega_2}(\alpha_k) > f_{2\gamma-\Omega_1,\Omega_2}(\alpha_\ell)\,.$$

Observe that, by our assumptions, $2\gamma - \Omega_1 > 0$, and $\Omega_2 > 0$. It is straightforward to show that $f'_{a,b}(t) = \frac{at+b}{(b-t^2)^{3/2}}$, and therefore $f_{2\gamma-\Omega_1,\Omega_2}(\alpha_k)$ is a *strictly increasing* mapping on $[0, \sqrt{b}]$. Since $\sqrt{\Omega_2} \geq \alpha_k > \alpha_\ell > 0$, we can conclude.

□

**Proof of Proposition 4.4.4.** In this proof, we set $p = \overline{\Phi} \circ L$. Let $A^F$ be the anchor containing all words of $\xi$. In our notation,

$$A^F = (m_1, \ldots, m_d)\,.$$

We first notice that $\mathcal{A}_1^{\mathrm{p}}$ is non-empty since $\overline{\Phi}\left(L\left(A^F\right)\right) = 1$. By construction, $\mathcal{A}_2^{\mathrm{p}}$, consisting of anchors of $\mathcal{A}_1^{\mathrm{p}}$ of minimal length, is non-empty as well. Lemma A.1.1 ensures that $\mathcal{A}_3^{\mathrm{p}}$, the anchors of $\mathcal{A}_2^{\mathrm{p}}$ with the highest $p$ value, is a non-empty subset of $\mathcal{A}_+$. Indeed, one can remove the anchors corresponding to the indices $j \in [d]$ such that $\lambda_j v_j < 0$, and increase the value of $p$. Since we assumed that at least one $\lambda_j$ is positive, it is always possible to do this removal. Finally, let $\ell$ be the common length of the anchors belonging to $\mathcal{A}_3^{\mathrm{p}}$. Since we satisfy the assumptions of Lemma A.1.2, we see that the $p$ value of any anchor of length $\ell$ is *strictly* increasing if we swap indices towards the lower indices. We deduce the result. □

### A.1.8 Additional result for Section 4.4.2: Simple if-then rules

We present here a further result on a simple if-then classifier based on the presence of disjoint subsets of words.

**Proposition A.1.3 (Small decision tree).** *Let the (binary) classifier $f$ be defined as follows:*

$$f(z) = \mathbb{1}_{(w_1 \in z \text{ and } w_2 \in z) \text{ or } w_3 \in z}$$

*Then, for any $\varepsilon > 0$, the anchor $A = (0, 0, 1)$, will be selected by exhaustive Anchors.*

For example, consider the sentiment analysis task, and $f$ the model returning 1 (a positive prediction) if words "not" *and* "bad" *or* the word "good" are present in the document. Proposition A.1.3 implies that only the word "good" will be selected as an anchor. This is a satisfying property and corresponds to the intuition that we have from Anchors: in this class of examples, **the smallest rule is provably selected by Anchors.** We prove Proposition A.1.3 in Section A.1.8 of the Appendix.

Of course, the scope of Proposition A.1.3 is limited. It is possible to obtain similar results for other simple sets of rules, though challenging to present these results with a sufficient amount of generality.

**Proof of Proposition A.1.3** Let us start by computing $\mathrm{Prec}\,(A)$ for any candidate anchor $A \in \mathcal{A}$. Since the model only depends on the first three coordinates, according to Proposition 4.4.1, we can restrict ourselves to $A = (a_1, a_2, a_3)$. In this proof, we set $p_i := 1 - \frac{1}{2^{m_i}}$ the probability of keeping

the word $w_i$ while sampling and $B_k \sim \mathrm{Bin}(m_j - a_j, 1/2)$. The precision of a candidate anchor $A \in \mathcal{A}$ (Eq. (4.2)) associated to $f$ is

$$\begin{aligned}
\mathrm{Prec}\,(A) &= \mathbb{E}_A\left[\mathbb{1}_{f(x)=f(\xi)}\right] \\
&= \mathbb{P}_A\left(f(x)=1\right) \\
&= \mathbb{P}_A\left(\varphi(x)_1 > 0\right)\cdot \mathbb{P}_A\left(\varphi(x)_2 > 0\right)\cdot\left(1-\mathbb{P}_A\left(\varphi(x)_3>0\right)\right)+\mathbb{P}_A\left(\varphi(x)_3>0\right) \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(by independence)} \\
&= \mathbb{P}_A\left(M_1 > 0\right)\cdot \mathbb{P}_A\left(M_2 > 0\right)\cdot\left(1-\mathbb{P}_A\left(M_3>0\right)\right)+\mathbb{P}_A\left(M_3>0\right) \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(since } v_j > 0 \text{ for all } j \in [d]) \\
&= \mathbb{P}_A\left(a_1+B_1 > 0\right)\cdot \mathbb{P}_A\left(a_2+B_2 > 0\right)\cdot\left(1-\mathbb{P}_A\left(a_3+B_3>0\right)\right)+\mathbb{P}_A\left(a_3+B_3>0\right) \\
&= \begin{cases}
1\,, & \text{if } a_3 > 0\,, \\
1\,, & \text{if } a_1, a_2 > 0 \text{ and } a_3 = 0\,, \\
p_2(1-p_3)+p_3 = 1-\frac{1}{2^{m_2+m_3}}\,, & \text{if } a_1 > 0 \text{ and } a_2, a_3 = 0\,, \\
p_1(1-p_3)+p_3 = 1-\frac{1}{2^{m_1+m_3}}\,, & \text{if } a_2 > 0 \text{ and } a_1, a_3 = 0\,, \\
p_1p_2(1-p_3)+p_3 = 1-\frac{2^{m_1}+2^{m_2}-1}{2^{m_1+m_2+m_3}}\,, & \text{if } a_1 = a_2 = a_3 = 0\,.
\end{cases}
\end{aligned}$$

Now let us follow Algorithm 1 step by step. According to the previous discussion, any anchor such that $a_1$ and $a_2$ are positive and/or $a_3 > 0$ has precision 1, and thus belongs to $\mathcal{A}_1$. In particular, the anchor $(0,0,1)$ belongs to $\mathcal{A}_1$. We note that it also has minimal length 1, and therefore belongs to $\mathcal{A}_2$. Finally, any other anchor with the same length will have a smaller precision, since $p_2(1-p_3)+p_3 < 1$, $p_1(1-p_3)+p_3 < 1$, and $p_1p_2(1-p_3)+p_3 < 1$. In conclusion, $\mathcal{A}_3$ is reduced to a singleton and the anchor $A = (0,0,1)$ will be selected by exhaustive Anchors. □

### A.1.9 Normalized TF-IDF

In this section we show that our theoretical results demonstrated considering a TF-IDF vectorization as defined in Definition 2.2.1 still hold for the $\ell_2$-normalized TF-IDF vectorization, defined as

$$\forall j \in [D], \qquad \phi(z)_j := \frac{m_j(z)v_j}{\sqrt{\sum_{j=1}^{D} m_j(z)^2 v_j^2}}\,,$$

that is, the default normalization in the `scikit-learn` implementation of TF-IDF. The main result of this section is the following:

**Proposition A.1.4 (Normalized-TF-IDF, Berry-Esseen).** *Assume that $0 < v_{min} \le v_j \le v_{max}$ and $m \le m_j \le M$ for all $j \in [d]$. Assume further that $A$ is not the empty anchor, that $|A| \le b/2$ and that $a_j < m_j$ for all $j$. Finally, assume that $\|\lambda\| = 1$ as $d \to +\infty$. For all $t \in \mathbb{R}$, define*

$$P(t) := \Phi\left(\frac{t\sqrt{\sum_j\{(m_j+a_j)^2+m_j-a_j\}v_j^2} - \sum_j \lambda_j(m_j+a_j)v_j}{\sqrt{\sum_j \lambda_j^2(m_j-a_j)v_j^2}}\right).$$

*Then*

$$\left|\mathbb{P}\left(\lambda^\top\phi(x) \le t\right) - P(t)\right| \le \frac{2\sqrt{M}v_{min}^{-3/2}}{d^{3/4-\varepsilon}} + 2\exp\left(-\frac{d^{1+2\varepsilon}v_{min}^6}{4v_{max}^4M^4}\right). \tag{A.8}$$

As a direct consequence of Proposition A.1.4, we know that a good approximation of $\mathrm{Prec}\,(A)$ in the normalized TF-IDF case is $\overline{\Phi}\,(L(A))$, with

$$L(A) = \frac{-\lambda_0\sqrt{\sum_j\{(m_j+a_j)^2+m_j-a_j\}v_j^2} - \sum_j \lambda_j(m_j+a_j)v_j}{\sqrt{\sum_j \lambda_j^2(m_j-a_j)v_j^2}}\,. \tag{A.9}$$

This is reminiscent of Eq. (4.9) in the non-normalized case. When $\lambda_0 = 0$, the analysis of the maximization problem is a subcase of the non-normalized case, and we recover the same result. Although $\lambda_0 = 0$ can be a reasonable assumption (assuming centered data and no intercept), we conjecture that the result is true for a larger range of $\lambda_0$, similarly to the unnormalized case.

Let us now prove Proposition A.1.4. Let us set

$$Z_d := \frac{\sum_{j=1}^d \lambda_j v_j M_j}{\sqrt{\sum_{j=1}^d v_j^2 M_j^2}} \qquad \text{and} \qquad \tilde{Z}_d := \frac{\sum_{j=1}^d \lambda_j v_j M_j}{\sqrt{\frac{1}{4}\sum_{j=1}^d\{(m_j+a_j)^2+m_j-a_j\}}}\,.$$

Intuitively, when $d$ is large enough, both these quantities are close with high probability, and $\tilde{Z}_d$ has the same structure as the linear form studied in the normalized case, up to a constant. Thus, the analysis boils down to the previous case, modulo the following:

**Proposition A.1.5** ($Z_d$ **and** $\tilde{Z}_d$ **are close with high probability**). *Let $\varepsilon \in (0, 1/2)$. Assume that $0 < v_{min} \leq v_j \leq v_{max}$ and $m \leq m_j \leq M$ for all $j \in [d]$. Assume further that $A$ is not the empty anchor. Finally, assume that $\|\lambda\| = 1$ as $d \to +\infty$. Then*

$$\mathbb{P}\left(\left|Z_d - \tilde{Z}_d\right| > \frac{c}{d^{1/2-\varepsilon}}\right) \leq 2\exp\left(-\frac{c^2 d^{1+2\varepsilon} v_{min}^6}{4v_{max}^4 M^4}\right)\,,$$

*for any small positive constant $c$.*

**Proof of Proposition A.1.5.** In this proof, we write $D := \sum_{j=1}^d v_j^2 M_j^2$. We begin by computing the expectation of $D$. We know that $M_j = a_j + B_j$, where $B_j \sim \mathrm{Bin}(m_j - a_j, 1/2)$. Therefore,

$$\begin{aligned}
\mathbb{E}\left[M_j^2\right] &= \mathbb{E}\left[a_j^2 + 2a_jB_j + B_j^2\right] \\
&= a_j^2 + a_j(m_j - a_j) + \frac{1}{4}(m_j-a_j)^2 + \frac{1}{4}(m_j - a_j) \\
\mathbb{E}\left[M_j^2\right] &= \frac{1}{4}(m_j+a_j)^2 + \frac{1}{4}(m_j-a_j)\,,
\end{aligned}$$

where we used Lemma A.2.1 to compute $\mathbb{E}\left[B_j^2\right]$. By linearity, we deduce that

$$\mathbb{E}\left[D\right] = \frac{1}{4}\sum_{j=1}^d\{(m_j+a_j)^2 + m_j - a_j\}\,.$$

Note that, with this notation in hand,

$$Z_d = \frac{\sum_{j=1}^d \lambda_j M_j v_j}{\sqrt{D}} \qquad \text{and} \qquad \tilde{Z}_d = \frac{\sum_{j=1}^d \lambda_j M_j v_j}{\sqrt{\mathbb{E}\left[D\right]}}\,.$$

We need to prove the following, which shows that $D$ is concentrated around its expectation:

**Lemma A.1.6** (**Concentration of** $D$). *Assume that* $0 < v_{min} \leq v_j \leq v_{max}$ *and that* $m \leq m_j \leq M$ *for all* $j \in [d]$. *Then, for all* $t > 0$,

$$\mathbb{P}\left(|D - \mathbb{E}\left[D\right]| > t\right) \leq 2\exp\left(\frac{-2t^2}{dv_{max}^4 M^4}\right).$$

*Proof.*

This is a straightforward application of Hoeffding's inequality once we notice that the random variables $v_j^2 M_j^2$ are bounded and independent, and that $\sum_j m_j^4 v_j^4 \leq dv_{\max}^4 M^4$ under our assumptions.

□

Note that Lemma A.1.6 is tight, since Hoeffding's inequality is tight for Bernoulli random variables, a case which is possible under our assumption. Lemma A.1.6 allows controlling the small deviations of $D$, a fact that we will maybe not use in the following, but can nonetheless be useful to split a complicated event. Next, we control the size of $D$.

**Lemma A.1.7** ($D$ **is small with high probability**). *Assume that* $0 < v_{min} \leq v_j \leq v_{max}$ *for all* $j \in [d]$. *Then*

$$\mathbb{P}\left(D < \frac{1}{4}dv_{min}^2\right) \leq 2\exp\left(\frac{-dv_{min}^4}{8v_{max}^4 M^4}\right).$$

*Proof.*

We write

$$\begin{aligned}\mathbb{P}\left(D < \frac{1}{4}dv_{\min}^2\right) &= \mathbb{P}\left(D - \mathbb{E}\left[D\right] < \frac{1}{4}dv_{\min}^2 - \mathbb{E}\left[D\right]\right) \\ &\leq \mathbb{P}\left(|D - \mathbb{E}\left[D\right]| < \frac{1}{2}dv_{\min}^2 - \frac{1}{4}dv_{\min}^2\right),\end{aligned}$$

since $\frac{1}{4}\{(m_j + a_j)^2 + m_j - a_j\} \geq \frac{1}{4} \cdot (1^2 + 1)$ for all $j \in [d]$. We conclude by applying Lemma A.1.6 with $t = \frac{1}{4}dv_{\min}^2$.

□

Now we can control the key quantity:

**Lemma A.1.8** (**Control of the key quantity**). *Assume that* $0 < v_{min} \leq v_j \leq v_{max}$ *and* $m \leq m_j \leq M$ *for all* $j \in [d]$. *Assume further that* $A$ *is not the empty anchor. Then, for any* $t > 0$,

$$\mathbb{P}\left(\left|1 - \frac{\sqrt{D}}{\sqrt{\mathbb{E}\left[D\right]}}\right| > t\right) \leq 2\exp\left(\frac{-d^2 t^2 v_{min}^6}{4v_{max}^4 M^4}\right).$$

In particular, by taking $t$ of the order $\frac{1}{d^{1/2-\varepsilon}}$ for some $\varepsilon > 0$, we see that $\mathbb{P}\left(\left|1 - \sqrt{D/\mathbb{E}\left[D\right]}\right| > t\right) = o\left(1\right)$.

*Proof.*

Multiplying by $\mathbb{E}[D]$, we see that we want to control

$$\mathbb{P}\left(\left|\sqrt{D}-\sqrt{\mathbb{E}[D]}\right|>t\sqrt{\mathbb{E}[D]}\right).$$

Since $\mathbb{E}[D]\leq\frac{1}{2}dv_{\text{min}}^2$, we can simply control

$$\mathbb{P}\left(\left|\sqrt{D}-\sqrt{\mathbb{E}[D]}\right|>\frac{1}{2}dtv_{\text{min}}^2\right).$$

Additionally, since $A$ is not the empty anchor, $D\geq v_{\text{min}}^2$ almost surely, which is positive. Since the mapping $x\mapsto\sqrt{x}$ is $1/2\sqrt{C}$-Lipschitz on $[C,+\infty)$, we see that

$$\left|\sqrt{D}-\sqrt{\mathbb{E}[D]}\right|\leq\frac{1}{2v_{\text{min}}}\left|D-\mathbb{E}[D]\right|,$$

which allows us to focus on

$$\mathbb{P}\left(\left|D-\mathbb{E}[D]\right|>dtv_{\text{min}}^3\right).$$

We control this last display using Lemma A.1.6, and we obtain

$$\mathbb{P}\left(\left|1-\frac{\sqrt{D}}{\sqrt{\mathbb{E}[D]}}\right|>t\right)\leq 2\exp\left(\frac{-d^2t^2v_{\text{min}}^6}{4v_{\text{max}}^4M^4}\right),$$

as promised.

□

Finally, coming back to the original problem, we write

$$\mathbb{P}\left(\left|Z_d-\tilde{Z}_d\right|>s\right)=\mathbb{P}\left(|Z_d|\cdot\left|1-\frac{\sqrt{D}}{\sqrt{\mathbb{E}[D]}}\right|>s\right)\leq\mathbb{P}\left(\left|1-\frac{\sqrt{D}}{\sqrt{\mathbb{E}[D]}}\right|>s\right). \tag{A.10}$$

By Cauchy-Schwarz inequality, we have

$$\left|\sum_j\lambda_jM_jv_j\right|\leq\|\lambda\|\cdot\sqrt{D},$$

and we deduce that $|Z_d|\leq\|\lambda\|=1$ under our assumptions. Coming back to Eq. (A.10), we can therefore take $s=\frac{1}{d^{1/2-\varepsilon}}$ and use Lemma A.1.8 to conclude.

□

We can now conclude this section with the proof of our main result.

**Proof of Proposition A.1.4.** Let us set $N:=\sum_j\lambda_jM_jv_j$. With this notation, $\mathbb{P}\left(\lambda^\top\phi(x)\leq t\right)=\mathbb{P}(Z_d\leq t)$, and

$$\lambda^\top\phi(x)=\frac{N}{\sqrt{D}}=Z_d\quad\text{and}\qquad\tilde{Z}_d=\frac{N}{\sqrt{\mathbb{E}[D]}}.$$

Let $s>0$. Using Lemma A.2.4, we have

$$\mathbb{P}\left(\tilde{Z}_d\leq t-s\right)-\mathbb{P}\left(\left|Z_d-\tilde{Z}_d\right|>s\right)\leq\mathbb{P}(Z_d\leq t)\leq\mathbb{P}\left(\tilde{Z}_d\leq t+s\right)+\mathbb{P}\left(\left|Z_d-\tilde{Z}_d\right|>s\right).$$

Let us set $s = \frac{1}{d^{1/2-\varepsilon}}$ for some small $\varepsilon > 0$. By Proposition A.1.5, we know that $\mathbb{P}\left(\left|Z_d - \tilde{Z}_d\right| > s\right) \leq 2\exp\left(-\frac{d^{1+2\varepsilon}v_{\min}^6}{4v_{\max}^4 M^4}\right)$. Let us now turn towards the remaining terms, depending on $\tilde{Z}$. We write, for any $u \in \mathbb{R}$,

$$\begin{aligned}
\mathbb{P}\left(\tilde{Z}_d \leq u\right) &= \mathbb{P}\left(\frac{N}{\sqrt{\mathbb{E}\left[D\right]}} \leq u\right) \\
&= \mathbb{P}\left(\frac{N - \mathbb{E}\left[N\right]}{\sqrt{\mathrm{Var}\left(N\right)}} \leq \frac{u\sqrt{\mathbb{E}\left[D\right]} - \mathbb{E}\left[N\right]}{\sqrt{\mathrm{Var}\left(N\right)}}\right) \\
\mathbb{P}\left(\tilde{Z}_d \leq u\right) &= \Phi\left(\frac{u\sqrt{\mathbb{E}\left[D\right]} - \mathbb{E}\left[N\right]}{\sqrt{\mathrm{Var}\left(N\right)}}\right) + o\left(1\right) ,
\end{aligned}$$

uniformly in $u$, where we used Proposition 4.4.3 in the last derivation.

Since $\Phi$ is 1-Lipschitz, we see that

$$\left|\mathbb{P}\left(\tilde{Z}_d \leq t + s\right) - \Phi\left(\frac{t\sqrt{\mathbb{E}\left[D\right]} - \mathbb{E}\left[N\right]}{\sqrt{\mathrm{Var}\left(N\right)}}\right)\right| \leq s\frac{\sqrt{\mathbb{E}\left[D\right]}}{\sqrt{\mathrm{Var}\left(N\right)}} + 2\exp\left(-\frac{d^{1+2\varepsilon}v_{\min}^6}{4v_{\max}^4 M^4}\right) .$$

Moreover, under our assumptions, $\sqrt{\mathbb{E}\left[D\right]/\mathrm{Var}\left(N\right)} = \mathcal{O}\left(1\right)$ and $s = \frac{1}{d^{1/2-\varepsilon}}$. Therefore,

$$\mathbb{P}\left(\tilde{Z}_d \leq t + s\right) = \Phi\left(\frac{t\sqrt{\mathbb{E}\left[D\right]} - \mathbb{E}\left[N\right]}{\sqrt{\mathrm{Var}\left(N\right)}}\right) + \frac{1}{d^{1/2-\varepsilon}}\frac{\sqrt{\mathbb{E}\left[D\right]}}{\sqrt{\mathrm{Var}\left(N\right)}} + 2\exp\left(-\frac{d^{1+2\varepsilon}v_{\min}^6}{4v_{\max}^4 M^4}\right) .$$

Additionally, one can show that $\mathrm{Var}\left(M_j^2\right) = \nu(m_j - a_j)$, with

$$\nu(x) := \frac{x}{8}(2(2a + x)^2 + x - 1) .$$

It is straightforward to show that $\nu$ is non-decreasing. When $m_j > a_j$, we see that

$$\nu(m_j - a_j) \geq \nu(1) = a_j^2 + a_j + \frac{1}{4} \geq \frac{1}{4} .$$

Therefore $\sum_j v_j^4 \mathrm{Var}\left(M_j^2\right) \geq \frac{d}{4}v_{\min}^4$. Under our assumptions, $\|\lambda\| = 1$, and by applying Cauchy-Schwarz inequality $\mathbb{E}\left[D\right] \leq \sqrt{d}v_{\max}M$ , we deduce

$$\frac{\mathbb{E}\left[D\right]}{\mathrm{Var}\left(N\right)} \leq \frac{4v_{\max}M}{\sqrt{d}v_{\min}^4} .$$

Thus, we find that

$$\begin{aligned}
\mathbb{P}\left(\tilde{Z}_d \leq t + s\right) &\leq \Phi\left(\frac{t\sqrt{\mathbb{E}\left[D\right]} - \mathbb{E}\left[N\right]}{\sqrt{\mathrm{Var}\left(N\right)}}\right) + \frac{1}{d^{1/2-\varepsilon}}\frac{2\sqrt{M}}{d^{1/4}v_{\min}^{3/2}} + 2\exp\left(-\frac{d^{1+2\varepsilon}v_{\min}^6}{4v_{\max}^4 M^4}\right) \\
&= \Phi\left(\frac{t\sqrt{\mathbb{E}\left[D\right]} - \mathbb{E}\left[N\right]}{\sqrt{\mathrm{Var}\left(N\right)}}\right) + \frac{2\sqrt{M}}{d^{3/4-\varepsilon}v_{\min}^{3/2}} + 2\exp\left(-\frac{d^{1+2\varepsilon}v_{\min}^6}{4v_{\max}^4 M^4}\right) .
\end{aligned}$$

The same reasoning applies to $t - s$, and we can conclude by recognizing $\Phi\left(\frac{t\sqrt{\mathbb{E}[D]} - \mathbb{E}[N]}{\sqrt{\mathrm{Var}(N)}}\right)$ as $P(t)$.

□

## A.2 Technical results

We present here some technical results that were used in our analysis regarding binomial random variables (Section A.2.1) and two additional lemmas (Section A.2.2).

### A.2.1 Binomial wonderland

In this section, we collect some facts about binomial random variables. We focus on the case $p = 1/2$ because of the sampling scheme of Anchors, with a few exceptions. We start with straightforward moment computations, which are stated here for completeness' sake.

**Lemma A.2.1 (Moments of the binomial distribution).** *Let $m \geq 1$ be an integer and $B \sim \mathrm{Bin}(m, 1/2)$. Then*

$$\mathbb{E}[B] = \frac{m}{2}, \quad \mathbb{E}\left[B^2\right] = \frac{m^2}{4} + \frac{m}{4}, \quad \mathbb{E}\left[B^3\right] = \frac{m^3}{8} + \frac{3m^2}{8}, \quad \textit{and} \quad \mathbb{E}\left[B^4\right] = \frac{m^4}{16} + \frac{3m^3}{8} + \frac{3m^2}{16} - \frac{m}{8}.$$

In particular, $\mathrm{Var}(B) = m/4$.

*Proof.*

We use the formula

$$\mathbb{E}[B^p] = \sum_{k=1}^{p} S_{k,p} m^{\underline{k}} \frac{1}{2^k},$$

where $m^{\underline{k}} = m(m-1)\cdots(m-k+1)$ and $S_{k,p}$ are the Stirling numbers of the second kind (see Knoblauch [2008] for instance).

□

Next, we turn to the computation of the third absolute moment of the binomial, which intervenes in the proof of Proposition 4.4.3.

**Lemma A.2.2 (Third absolute moment of the binomial).** *Let $m \geq 1$ be an even integer. Then*

$$\mathbb{E}\left[|B - m/2|^3\right] = \frac{m^2}{2^{m+2}} \binom{m}{m/2}.$$

From Lemma A.2.2, we deduce that

$$\forall m \geq 1, \qquad \mathbb{E}\left[|B - m/2|^3\right] \leq \frac{1}{\sqrt{8\pi}} m^{3/2}, \tag{A.11}$$

where we used the well-known bound $\binom{m}{m/2} \leq \frac{\sqrt{2}2^m}{\sqrt{\pi m}}$. Eq. (A.11) is better than a Jensen-type bound, which can be obtained by noticing that

$$\left(\mathbb{E}\left[|B - m/2|^3\right]\right)^{4/3} \leq \mathbb{E}\left[(B - \mathbb{E}[B])^4\right] = \frac{m}{4}\left(1 + \frac{3m-6}{4}\right),$$

where we used Lemma A.2.1 in the last step. This last expression is less than $3m^2/16$, and this approach yields $\mathbb{E}\left[|B - m/2|^3\right] \leq (3/16)^{3/4} m^{3/2}$. Since $1/\sqrt{8\pi} \approx 0.2$ whereas $(3/16)^{3/4} \approx 0.28$, we prefer the use of Eq. (A.11) when bounding the third absolute moment of the binomial.

*Proof.*

We follow Diaconis and Zabell [1991]. First, we notice that the polynomial $(X - m/2)^3$ can be written

$$(X - m/2)^3 = \frac{-3}{4} P_3^m(X) + \frac{-3m+2}{8} P_1^m(X)\,, \tag{A.12}$$

where $P_k^m$ denotes the Kravchuk polynomial of order $k$ [MacWilliams and Sloane, 1977]. Using Lemma 1 of Diaconis and Zabell [1991], we see that

$$\begin{aligned}
\frac{1}{2^m}\sum_{k=0}^{m/2}\binom{m}{k}(k-m/2)^3 &= \frac{-3}{4}\frac{1}{2^m}\sum_{k=0}^{m/2}\binom{m}{k}P_3^m(k) + \frac{-3m+2}{8}\frac{1}{2^m}\sum_{k=0}^{m/2}\binom{m}{k}P_3^m(k) \\
&= \frac{-3}{4}\frac{m}{6}\frac{1}{2^m}\binom{m}{m/2}P_2^{m-1}(m/2) \\
&\quad + \frac{-3n+2}{8}\frac{m}{2}\frac{1}{2^m}\binom{m}{m/2}P_0^{m-1}(m/2) \\
\frac{1}{2^m}\sum_{k=0}^{m/2}\binom{m}{k}(k-m/2)^3 &= \frac{-m^2}{2^{m+3}}\binom{m}{m/2}\,.
\end{aligned}$$

Observing that the third absolute moment is twice the absolute value of the last display yields the desired result.

□

*Remark A.2.1 –* It is unfortunately not possible to obtain a simple closed-form for a parameter of the binomial $p$ not equal to $1/2$ using this method. Indeed, using the more general expression of the Kravchuk polynomials (sometimes called the Meixner polynomials [Meixner, 1934]) the decomposition obtained in Eq. (A.12) becomes $(X - mp)^3 = \sum_{q=0}^{3} \lambda_q P_q^m(X)$, with

$$\begin{cases}
\lambda_0 &= mp(1-p)(1-2p) \\
\lambda_1 &= \frac{-3m+2}{4}(1-p) + \frac{3m-6}{4}(1-p)(1-2p)^2 \\
\lambda_2 &= 6(1-2p)(1-p)^2 \\
\lambda_3 &= -6(1-p)^3\,.
\end{cases}$$

In particular, $\lambda_0$ is nonzero whenever $p \neq 1/2$. Therefore, the partial sums of the binomial coefficients make their appearance, for which there is no simple closed-form.

### A.2.2 Other probability results

**Lemma A.2.3 (Probability splitting).** *Let $X$ and $Y$ be two random variables, $t \in \mathbb{R}$ and $\varepsilon > 0$. Then*

$$\mathbb{P}(Y \leq t) \leq \mathbb{P}(X \leq t + \varepsilon) + \mathbb{P}(|X - Y| > \varepsilon)\,.$$

*Proof.*

This result is classical, we report the proof for completeness' sake.

$$\begin{aligned}
\mathbb{P}(Y \leq t) &= \mathbb{P}(Y \leq t, X \leq t+\varepsilon) + \mathbb{P}(Y \leq t, X > t+\varepsilon) \\
&\leq \mathbb{P}(X \leq t+\varepsilon) + \mathbb{P}(Y - X \leq t - X, t - X < -\varepsilon) \\
&\leq \mathbb{P}(X \leq t+\varepsilon) + \mathbb{P}(Y - X < -\varepsilon) \\
&\leq \mathbb{P}(X \leq t+\varepsilon) + \mathbb{P}(Y - X < -\varepsilon) + \mathbb{P}(Y - X > \varepsilon) \\
\mathbb{P}(Y \leq t) &\leq \mathbb{P}(X \leq t+\varepsilon) + \mathbb{P}(|X - Y| > \varepsilon) \ .
\end{aligned}$$

□

As a direct consequence, we have the following:

**Lemma A.2.4 (Convergence in probability implies convergence in distribution).** *Let $X$ and $Y$ be two random variables, $t \in \mathbb{R}$ and $s > 0$. Then*

$$\mathbb{P}(X \leq t - s) - \mathbb{P}(|X - Y| > s) \leq \mathbb{P}(Y \leq t) \leq \mathbb{P}(X \leq t + s) + \mathbb{P}(|X - Y| > s) \ .$$

*Proof.*

Applying Lemma A.2.3 to $Y$ and $X$ instead of $X$ and $Y$, and $t - s$ instead of $t$ yields

$$\mathbb{P}(X \leq t - s) \leq \mathbb{P}(Y \leq t) + \mathbb{P}(|X - Y| > s) \ . \tag{A.13}$$

Combined with the original statement, we obtain the result.

□

## A.3 Additional experimental results

In this section, additional experimental results are collected. Specifically, in Section A.3.1 we report statistics about the TF-IDF vectorization, in Section A.3.2 we empirically show that Anchors and exhaustive Anchors produce similar explanations, In Section A.3.3 we provide a counterexample proving that the default implementation of Anchors does not satisfy Property 4.4.1 (Dummy Property). Sections A.3.4 and A.3.6 provide empirical validation of Propositions 4.4.3 and A.1.4, respectively. Finally, additional experimental results for Sections 4.4 and 4.5 are in Sections A.3.5 and A.3.7. The code used for the experiments is available at `https://github.com/gianluigilopardo/anchors_text_theory`.

**Setting.** All the experiments reported in this Section and in Chapter 4 are implemented in `Python` and executed on CPUs. Three dataset are used: Restaurant Reviews (available at `https://www.kaggle.com/hj5992/restaurantreviews`), Yelp Reviews (available at `https://www.kaggle.com/omkarsabnis/yelp-reviews-dataset`), and IMDB Reviews (available at `https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews`). Unless otherwise specified, all the experiments work with the official implementation of Anchors (available and licensed at `https://github.com/marcotcr/anchor`) and default parameters. The vectorizer is always TF-IDF from `https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html` with the option

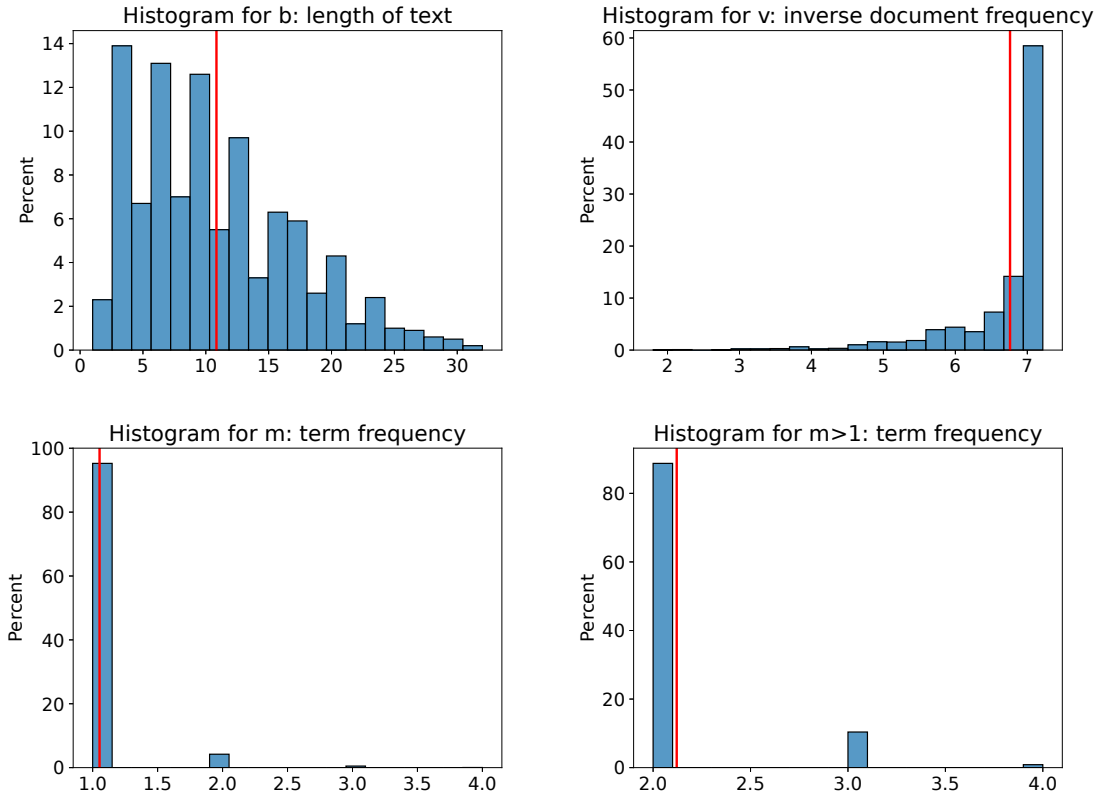


Figure A.2 – **Histograms for the Restaurant Reviews dataset.** Figures report the length of the document $b$ (upper left), the inverse document frequency $v_j$ (upper right), the term frequency $m$ (lower left), the term frequency when $m > 1$ (lower right). Average value is shown in red.

`norm=None`. When experiments require it (Sections A.3.2, A.3.3, A.3.5, A.3.7), we use 75% of the dataset for training and 25% for testing. All machine learning models used in the experiments were trained with the default parameters of `https://scikit-learn.org/`. Finally, we remark that we always consider documents with positive predictions, *i.e.*, such that $f(z) = 1$.

### A.3.1 Typical values of $m_j$ and $v_j$

Figure A.2 and Figure A.3 show statistics about the TF-IDF transforms of the two considered datasets. In Figure A.2 the average document length $b$ is 11: each document is a short review, generally containing one or two short sentences, while in Figure A.3 the average length $b$ is 133: documents are quite longer. This significant difference in documents size is also visible in the multiplicities. In Figure A.2, the typical value for the term frequency $m_j$ is 1 and it is rarely higher than 3, while in Figure A.3 the average is closer to 2 and multiplicities greater than 10 are present. In contrast, the average, median, and maximum value for the inverse document frequency $v_j$ are around 7 for both datasets: indeed, considering their size is around $N = 1000$ and that the typical value for $N_j$ is 1, we get $v_j = \log \frac{N+1}{N_j+1} + 1 \approx 7$.

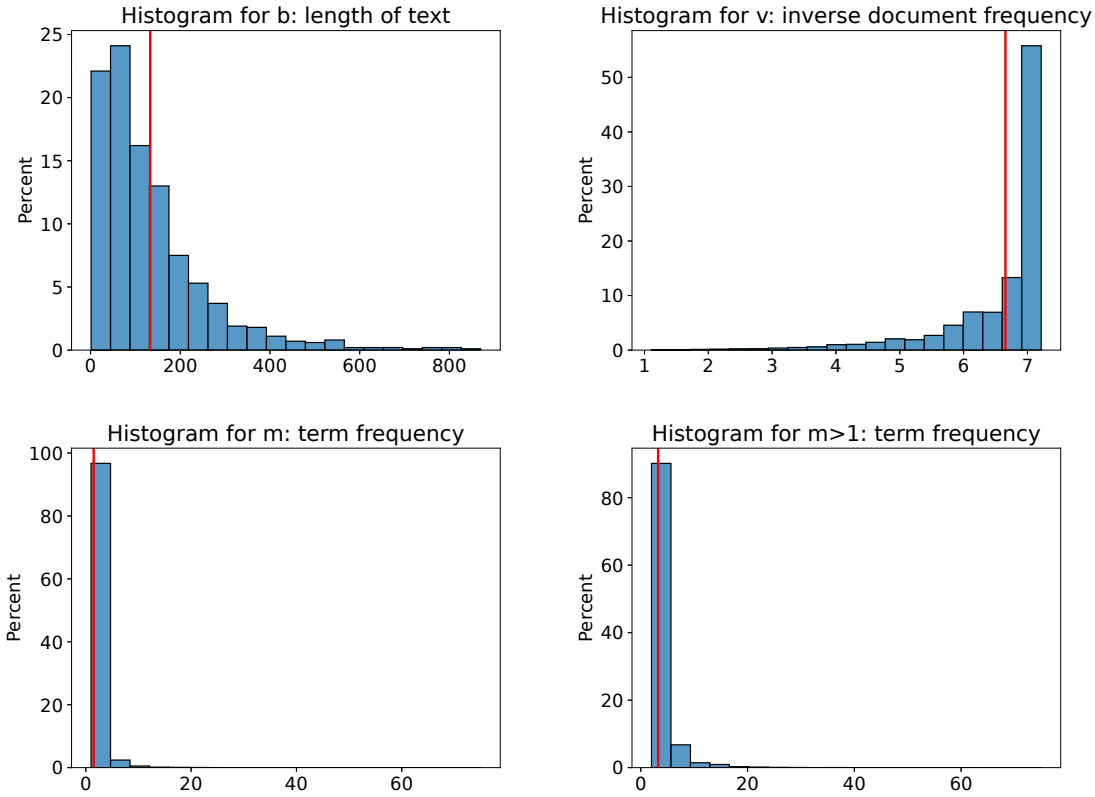


Figure A.3 – **Histograms for the Yelp Reviews dataset.** Figures report the length of the document $b$ (upper left), the inverse document frequency $v_j$ (upper right), the term frequency $m$ (lower left), the term frequency when $m > 1$ (lower right) for a subset of the dataset. Note that the maximum value of multiplicity $m$ is 75 in this case, while the average (in red) is 1.5.

### A.3.2 Comparison between Anchors and exhaustive Anchors

We compute the similarity through the Jaccard index, defined as

$$J(A^{\mathrm{p}}, A^{\mathrm{d}}) := \frac{\left|A^{\mathrm{p}} \cap A^{\mathrm{d}}\right|}{\left|A^{\mathrm{p}} \cup A^{\mathrm{d}}\right|} = \frac{\left|A^{\mathrm{p}} \cap A^{\mathrm{d}}\right|}{|A^{\mathrm{p}}| + |A^{\mathrm{d}}| - |A^{\mathrm{p}} \cap A^{\mathrm{d}}|},$$

where $A^{\mathrm{p}}$ is the anchor obtained by running empirical Anchors (exhaustive version with empirical precision as an evaluation function) and $A^{\mathrm{d}}$ with default implementation. Table A.1 shows the average Jaccard index for the two datasets considered and five different models. Overall, the output of the two methods is quite similar.

As shown in Figure A.3, the Yelp dataset has longer documents, making Anchors more unstable (namely outputting quite different anchors for the same model / document configuration). This explains why the similarity is lower in that case. In addition, Anchors requires a computational capacity that grows exponentially with the length of the document (and the length of the

TABLE A.1 – Jaccard similarity between exhaustive Anchors and default implementation.

| | Indicator | DTree | Logistic | Perceptron | RandomForest |
|---|---|---|---|---|---|
| Restaurants | 1.00 | 1.00 | 0.90 | 0.87 | 0.93 |
| Yelp | 1.00 | 1.00 | 0.71 | 0.68 | 0.75 |

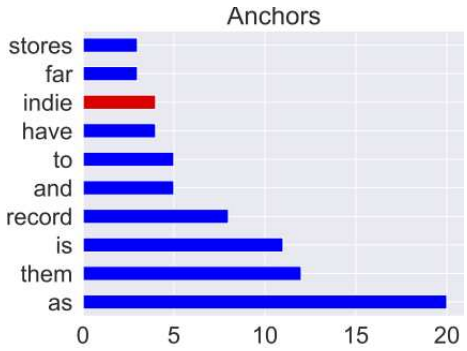


Figure A.4 – **Anchors includes dummy features.** 10 most frequent anchors on 100 runs of default Anchors on a logistic model with zero coefficient for *indie* and arbitrary coefficients for the other words. *indie* is a dummy feature, but still appears in 4 anchors.

optimal anchor). This makes it particularly onerous to apply empirical Anchors to large documents. Indeed, the experiment of Table A.1 requires about half an hour on Restaurants reviews, while more than 24 hours are needed on Yelp reviews.

### A.3.3 Dummy property

We report a counterexample showing that the default implementation of Anchors does not satisfy Proposition 4.4.1. In Figure A.4, the word *indie* appears in 4 anchors, even though the model does not depend on it. While the frequency of occurrence is not high, it is still non-zero. This is slightly problematic in our opinion: since the model does not depend on the word *indie*, its appearance in the explanation is misleading for the user. We conjecture that this behavior is entirely due to the optimization procedure used in the default implementation of Anchors, since the exhaustive version is guaranteed not to have this behavior by Proposition 4.4.1. We want to emphasize that there is nothing special with the example presented here and other counterexamples can be readily created.

### A.3.4 Empirical validation of Proposition 4.4.3: Precision of a linear classifier

Figure A.5 shows an empirical validation for Proposition 4.4.3 for different document size and for anchors of different sizes. The fit between the empirical distribution and $\overline{\Phi} \circ L$ is much better as predicted by Proposition 4.4.3, even for small values of $d$. This motivates our further study of the approximate precision instead of the precision. From the results in Figure A.5, we can see why the anchors need to be small with respect to the document size: if they are two large, the approximation of the precision is not justified. We remark, again, that this assumption is entirely reasonable, since an anchor using more than half the document to explain a prediction is not interpretable. In addition, Anchors rarely returns such anchors.

### A.3.5 Additional experiments for Section 4.4: Analysis on explainable classifiers

In this Section we report additional experiments for our Analysis on explainable classifiers. First, we validate our results on simple if-then-rules: Figure A.7 and Figure A.8 illustrate Proposition A.1.3 and Proposition 4.4.2, respectively.

Second, we validate Proposition 4.4.4 as in Figure A.5, *i.e.*, after training a logistic model (Figure A.9) and a perceptron model (Figure A.10), we apply a shift $S$ to the intercept $\lambda_0$, as follows

$$f(z) = \mathbb{1}_{\lambda^\top \varphi(z) + (\lambda_0 - S) > 0} \,. \tag{A.14}$$

As $S$ increases, the prediction becomes harder, and longer anchors are needed to reach the precision threshold. When a new word is included, we show that, as predicted by Proposition 4.4.4, the first word with higher $\lambda_j v_j$ is picked.

**Error bars** In our experiments there are two sources of variability, coming from different runs and documents, as we ran 10 times Anchors on each positively classified document. Figure A.6 shows the standard deviation for 10 runs on Restaurant reviews (model is a 10-layers neural network): for half the documents, it is actually zero.

To further demonstrate this phenomenon, we also conducted the following experiment. We first trained a logistic model on three review datasets, achieving accuracies between $80\%$ and $90\%$. We then ran Anchors with the default setting 10 times on positively classified documents. For each document, we measure the Jaccard similarity between the anchor $A$ and the first $|A|$ words ranked by $\lambda_j v_j$. In Table 4.1 we report the average Jaccard index: results validate Proposition 4.4.4.

### A.3.6 Empirical validation of Proposition A.1.4: Normalized-TF-IDF, Berry-Esseen

Figure A.11 shows an empirical validation for Proposition A.1.4 for different document size and for anchors of different sizes.

### A.3.7 Additional experiments for Section 4.5: Anchors on Neural Networks

We show in Table 4.2 additional experiments that validate our conjecture expressed in Section 4.5. To this end, we trained, for each dataset (Restaurants, Yelp, and IMDB), three feed-forward neural networks, with 3, 10, and 20 layers, achieving accuracies around $90\%$. The code used for model training is available at `https://github.com/gianluigilopardo/anchors_text_theory`. We then ran Anchors with default settings 10 times on positively classified documents. For each document, we get the gradient of the model with respect to the input: for all $j \in [d]$, $\lambda_j := \frac{\partial g(\varphi(x))}{\partial \varphi(x)_j}$. We then measure the average Jaccard similarity between the anchor $A$ and the first $|A|$ word ranked by $\lambda_j v_j$.

### A.3.8 BERT replacement

As discussed in Section 4.4.1, we study the `UNK`-replacement option even if when replacing words with a fixed token can produce unrealistic samples and lead to out-of-distribution issue. Nevertheless, we performed the same experiments of Section 4.5 using the BERT-replacement option when a 3-layers neural network is applied on a sample of 50 Restaurants reviews. Somewhat

surprisingly, our message still stands: we reach a Jaccard Similarity of $0.83 \pm 0.2$, similarly to the `UNK` setting. What is more, we notice that such option is $10$ times slower and produces longer anchors.

TABLE A.2 – **Anchors on a neural network.** Average Jaccard similarity between the anchor $A$ and the first $|A|$ words ranked by $\lambda_j v_j$ for a 3-layers feed forward neural network on $50$ Restaurant reviews, using the `BERT`-replacement option.

| | full | $\texttt{pr} < 0.85$ | $\texttt{pr} < 0.75$ |
|---|---|---|---|
| Restaurants | $0.83 \pm 0.2$ | $0.83 \pm 0.2$ | $0.81 \pm 0.2$ |

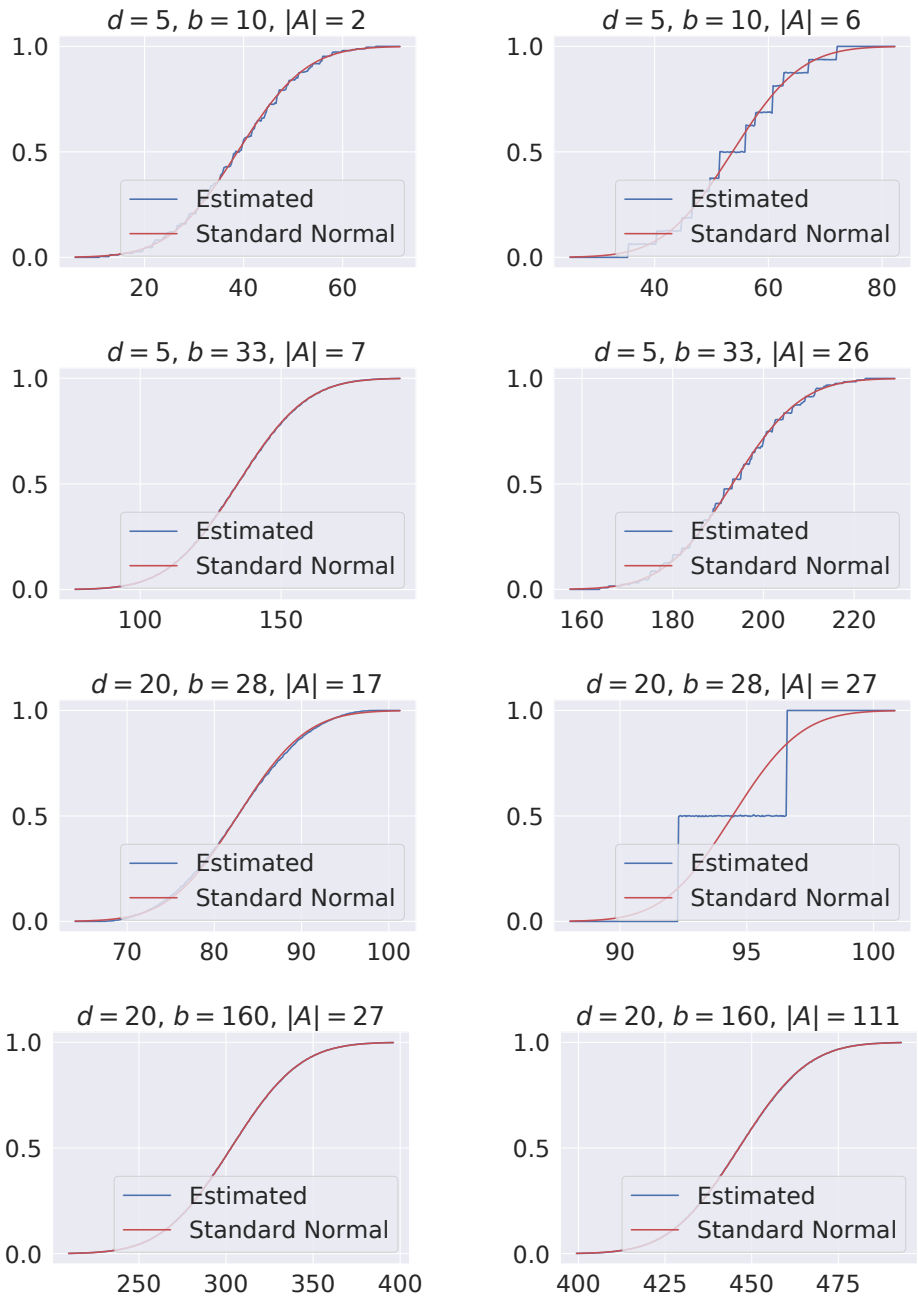


Figure A.5 – **Illustration of Proposition 4.4.3.** The multiplicities are arbitrary numbers between 1 and 10. The $\lambda_j v_j$ where drawn according to a Gaussian. Monte-Carlo simulation of the probability in blue $10^5$ simulations). In red, the cumulative distribution function of the $\mathcal{N}(0, 1)$. Note that Proposition 4.4.3 assumes $|A| \leq b/2$: the approximation may be inaccurate when this assumption is not satisfied (right column).

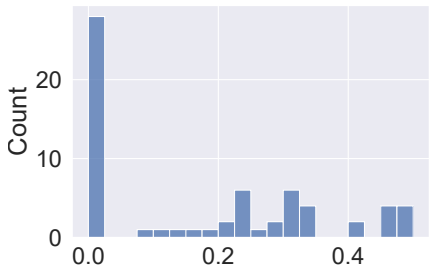


Figure A.6 – **Jaccard index standard deviations** for 10 runs on Restaurant reviews (model is a 10-layers neural network), same experiment as Table 2 in Chapter 4. Overall standard deviation is 0.39, while the average one is 0.17.


Figure A.7 – **Illustration of Proposition A.1.3.** Count on 100 runs of Anchors on four different reviews. Classification rules consist of two conditions, shown above each figure. Note that, for each case, **both conditions are satisfied** by the example. The shorter is always selected, as predicted by Proposition A.1.3.

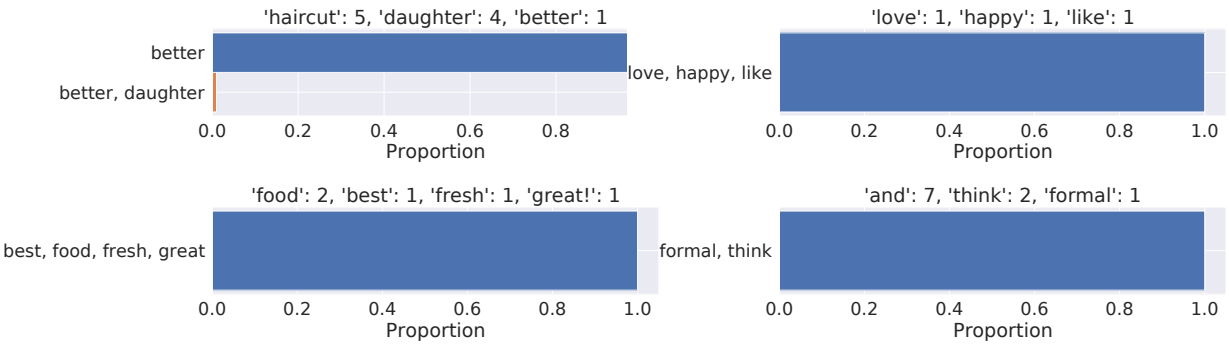


Figure A.8 – **Illustration of Proposition 4.4.2.** Count on 100 runs of Anchors on four different reviews. The classifier looks for the joint presence of the words reported above each figure (for instance, the model in the upper left panel predicts 1 if $z$ contains "haircut," "daughter," and "better"). The multiplicities of each word in the document is reported. As predicted by Proposition 4.4.2, words with multiplicity higher than a given threshold disappear from the explanation.

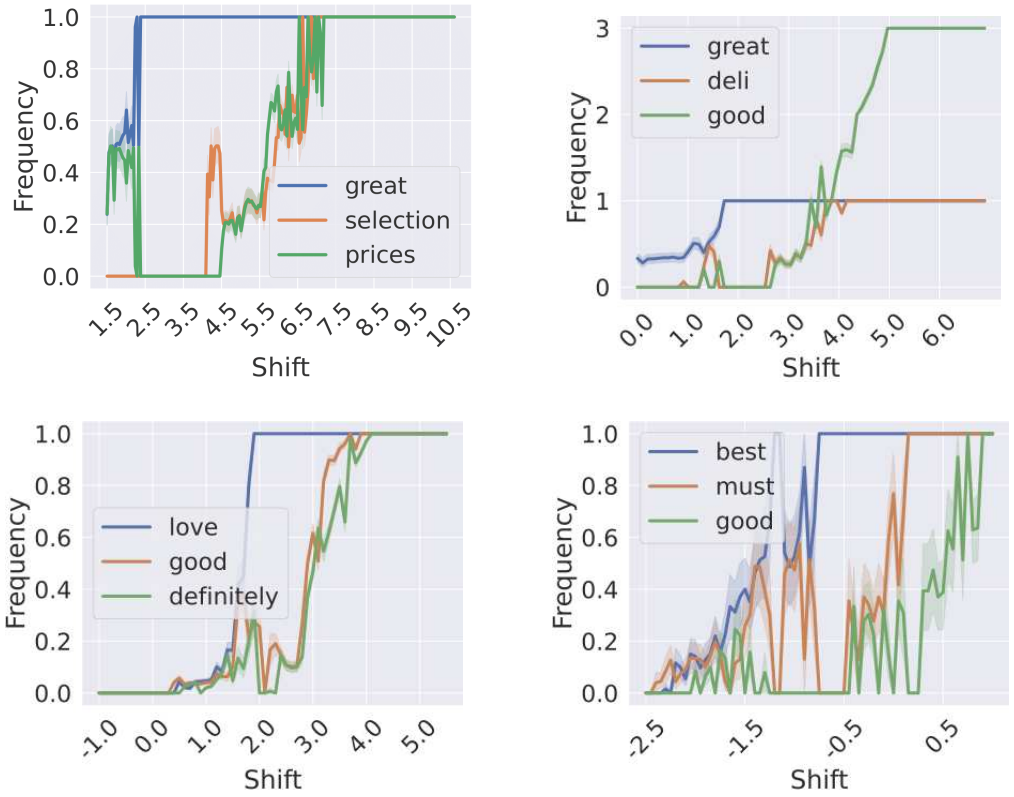


Figure A.9 – **Illustration of Proposition 4.4.4.** Frequency on 100 runs of Anchors on four different documents with four different logistic models when a shift $S$ is applied. Legend shows the first three words ordered by $\lambda_j v_j$. For example, in the lower-right figure, the shift $S$ increases from $-2.5$ to 1. As predicted by Proposition 4.4.4, first the words with higher $\lambda_j v_j$ are selected. Note that the overlap of some curves is due to similar coefficients for the corresponding words.

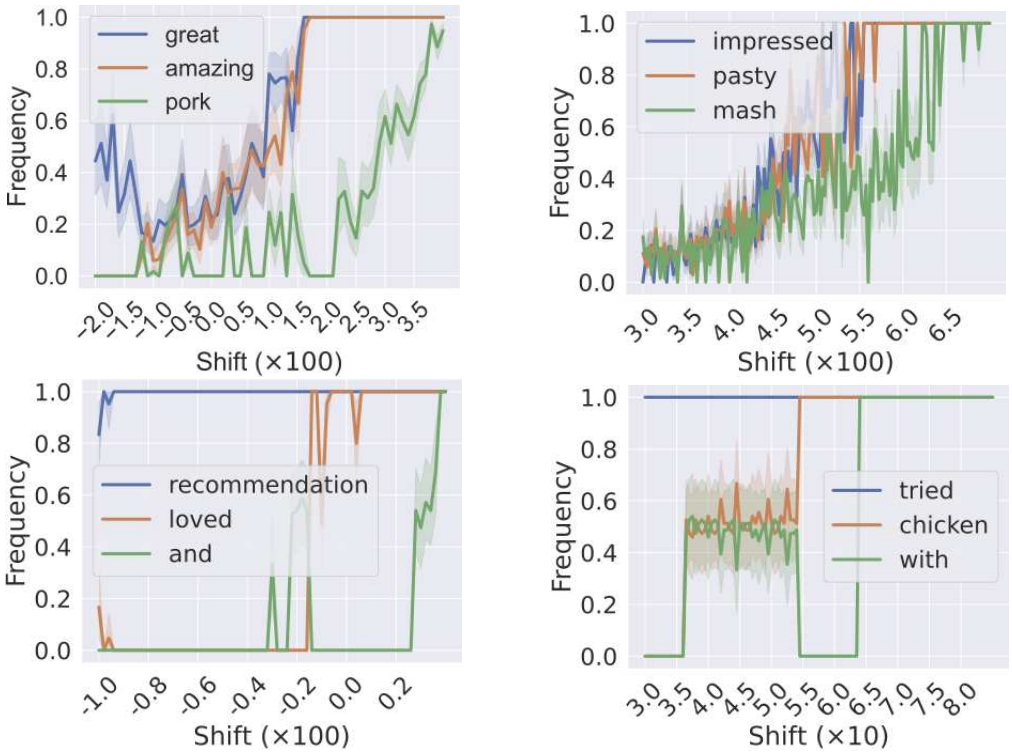


Figure A.10 – **Illustration of Proposition 4.4.4.** Frequency on 100 runs of Anchors on four different documents with four different perceptron models when a shift $S$ is applied. Legend shows the first three words ordered by $\lambda_j v_j$. Anchors includes new words in order of $\lambda_j v_j$, as predicted by Proposition 4.4.4. The overlap of some curves is due to similar coefficients for the corresponding words. For example, in the top-left figure, "great" has a coefficient $\lambda_j v_j$ equal to 207, close to the coefficient for "amazing", 192.

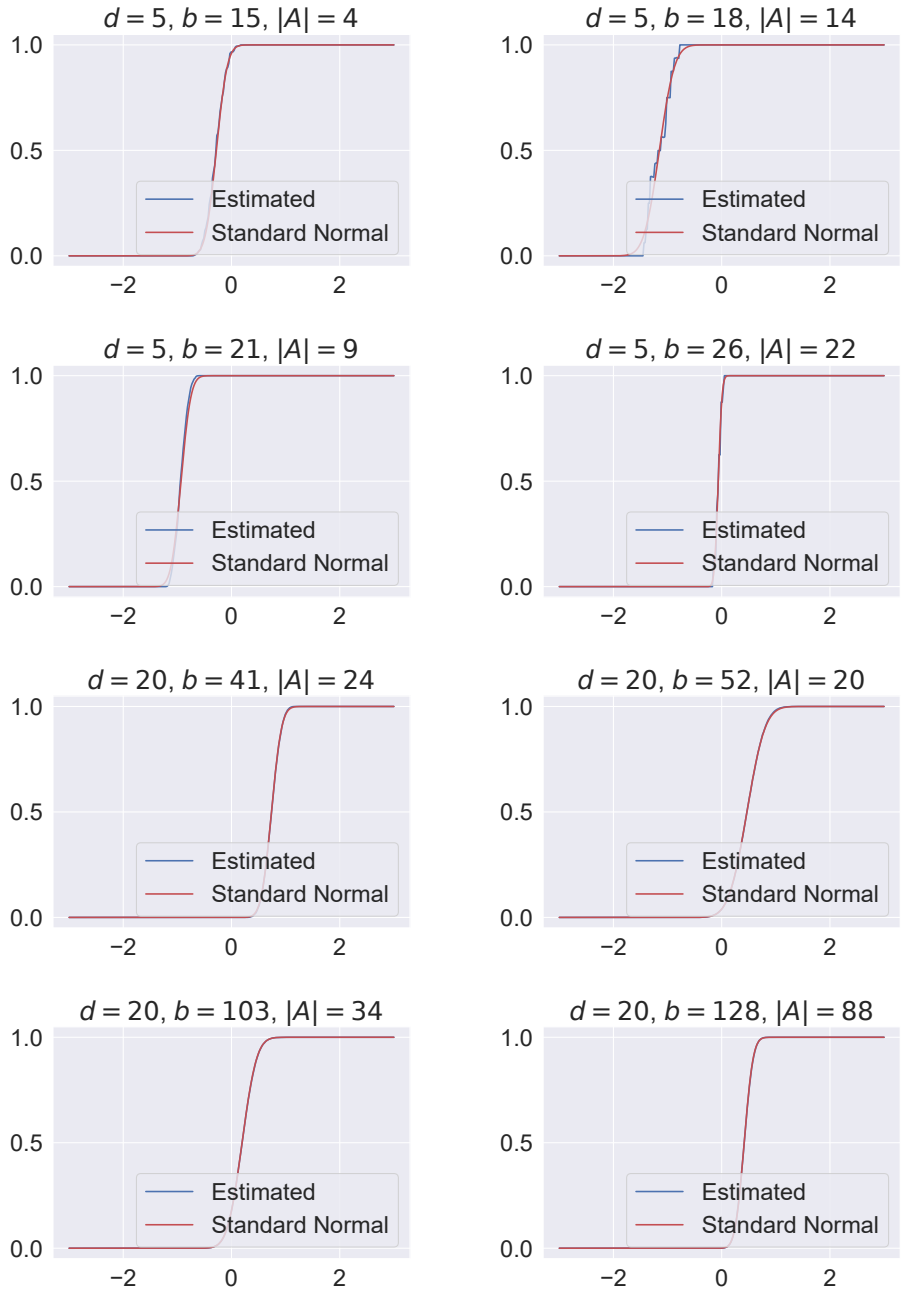


Figure A.11 – **Illustration of Proposition A.1.4.** The multiplicities are arbitrary numbers between 1 and 10. The $\lambda_j v_j$ were drawn according to a Gaussian. Monte-Carlo simulation of the probability in blue $10^5$ simulations). In red, the cumulative distribution function of the $\mathcal{N}(0, 1)$. Note that Proposition A.1.4 assumes $|A| \leq b/2$: the approximation may be inaccurate when this assumption is not satisfied (right column).

# APPENDIX B

# Appendix for Chapter 5: *Faithful and Robust Local Interpretability for Textual Predictions*

**Organization of the Appendix.** Section B.1 provides the theoretical proofs for the results presented in Chapter 5. Section B.2 details the implementation and presents additional results. Further experimental details, along with the code for FRED and the experiments, are available at https://github.com/gianluigilopardo/fred.

## B.1 Proofs

In this section, we collect the proofs of the theoretical results presented in the Chapter 5. First, we prove Lemma 5.2.1 and Lemma 5.2.2 from Section 5.2. Then, we move to the mathematical validation of our analysis presented in Section 5.3 by proving Proposition 5.3.1 and Proposition 5.3.2.

### B.1.1 Proof of Lemma 5.2.1: Convergence of Empirical Drop $\widehat{\Delta}_c$

Let $n$ denote the sample size. We are interested in the empirical average of predictions on instances that do not contain the candidate $c$, *i.e.*, $\frac{1}{n_c}\sum_{c\notin x^{(i)}} f(x^{(i)})$. This quantity can be rewritten using indicator functions, as

$$\frac{1}{n_c}\sum_{c\notin x^{(i)}} f(x^{(i)}) = \frac{n}{n_c}\frac{1}{n}\sum_{i=1}^{n} f(x^{(i)})\mathbb{1}_{c\notin x^{(i)}} \,.$$

We can therefore apply the weak law of large numbers. As $n$ grows infinitely, this empirical average converges in probability to the expected value of the predictions for instances without $c$:

$$\frac{1}{n}\sum_{i=1}^{n} f(x^{(i)})\mathbb{1}_{c\notin x^{(i)}} \xrightarrow{\mathbb{P}} \mathbb{E}\left[f(x)\mathbb{1}_{c\notin x}\right] \,.$$

Likewise, as $n \to +\infty$, $\frac{n}{n_c}$ converges in probability to $\frac{1}{\mathbb{P}(c \notin x)}$. By multiplying the two limits, we get:

$$\frac{1}{n_c} \sum_{c \notin x^{(i)}} f(x^{(i)}) \xrightarrow{\mathbb{P}} \frac{\mathbb{E}\left[f(x)\mathbb{1}_{c \notin x}\right]}{\mathbb{P}\left(c \notin x\right)}.$$

This approximation holds true when both $n$ and $n_c$ are sufficiently large, facilitating the estimation of the drop in prediction associated to a candidate. □

### B.1.2 Proof of Lemma 5.2.2: Choosing $n$

For any candidate $c$ with size $l = |c|$, and any sample $x^{(i)}$, $i \in [n]$, $\mathbb{P}\left(c \notin x^{(i)}\right) = \frac{1}{2^l}$, since word removals are *i.i.d.* with probability $1/2$. Then,

$$\begin{aligned}
\mathbb{P}\left(\exists i \in [n] : c \notin x^{(i)}\right) &= \sum_{k=1}^{n} \binom{n}{k} \left(\frac{1}{2^l}\right)^k \left(1 - \frac{1}{2^l}\right)^{n-k} \\
&= 1 - \left(1 - 1/2^l\right)^n \\
\mathbb{P}\left(\exists i \in [n] : c \notin x^{(i)}\right) \geq \alpha &\iff n \geq \frac{\log\left(1-\alpha\right)}{\log\left(1 - 1/2^l\right)}.
\end{aligned}$$

Since a candidate has maximal length $l_{max}$, we can choose the sample size by setting $n = \left\lceil \frac{\log\left(1-\alpha\right)}{\log\left(1-1/2^{l_{max}}\right)} \right\rceil$. □

### B.1.3 Proof of Proposition 5.3.1: Linear models

Let $\Delta_c$ be the drop in prediction associated to candidate $c$. Let $\lambda, \lambda_0$ be the coefficients associated to the linear classifier $f$ defined by Eq. (4.7). Assume $\lambda_1 v_1 > \lambda_2 v_2 > \cdots > \lambda_d v_d$. Then,

$$\begin{aligned}
\Delta_c &= \mathbb{E}_c\left[d(x)\right] = \mathbb{E}\left[f(x)\right] - \mathbb{E}_c\left[f(x)\right] = \mathbb{E}\left[\lambda^\top \varphi(\xi)\right] - \mathbb{E}_c\left[\lambda^\top \varphi(x))\right] \\
&= \mathbb{E}\left[\sum_{j=1}^{d} \lambda_j v_j M_j\right] - \mathbb{E}_c\left[\sum_{j=1}^{d} \lambda_j v_j M_j\right] \\
&= \sum_{j=1}^{d} \lambda_j v_j \mathbb{E}\left[M_j\right] - \sum_{j=1}^{d} \lambda_j v_j \mathbb{E}_{c_j}\left[M_j\right] && \text{(since removals are } \textit{i.i.d.}\text{)} \\
&= \sum_{j=1}^{d} \lambda_j v_j p m_j - \left(\sum_{c_j=0} \lambda_j v_j \mathbb{E}\left[M_j\right] + \sum_{c_i \neq 0} \lambda_i v_i \mathbb{E}\left[M_i\right]\right) \\
&= \sum_{j=1}^{d} \lambda_j v_j p m_j - \left(\sum_{c_j=0} \lambda_j v_j p m_j + \sum_{c_i \neq 0} \lambda_i v_i p (m_i - 1)\right) && \text{(since } M_i \sim B(m_i - c_i, p)\text{)} \\
&= p \sum_{c_i \neq 0} \lambda_i v_i c_i, \,,
\end{aligned}$$

and, for any fixed length $l$, this quantity is maximized by the subset $c$ having $|c| = l$ and containing the firsts $l$ words ranked by $\lambda_j v_j$. □

### B.1.4 Proof of Proposition 5.3.2: Presence of shortcuts

Let us assume $f(\xi) = 1$ (the case $f(\xi) = 0$ is specular). We can rewrite the drop in prediction associated to the candidate $c$ as follows.

$$
\begin{aligned}
\Delta_c &= \mathbb{E}_c\left[d(x)\right] = \mathbb{E}\left[f(x)\right] - \mathbb{E}_c\left[f(x)\right] = \mathbb{E}\left[\prod_{j\in J} \mathbb{1}_{w_j \in x}\right] - \mathbb{E}_c\left[\prod_{j\in J} \mathbb{1}_{w_j \in x}\right] \\
&= \mathbb{E}\left[\prod_{j\in J} \mathbb{1}_{\varphi(x)_j > 0}\right] - \mathbb{E}_c\left[\prod_{j\in J} \mathbb{1}_{\varphi(x)_j > 0}\right] \\
&= \prod_{j\in J} \mathbb{E}\left[\mathbb{1}_{\varphi(x)_j > 0}\right] - \prod_{j\in J} \mathbb{E}_c\left[\mathbb{1}_{\varphi(x)_j > 0}\right] && \text{(since removals are } \textit{i.i.d.}\text{)} \\
&= \prod_{j\in J} \mathbb{P}\left(\varphi(x)_j > 0\right) - \prod_{j\in J} \mathbb{P}_c\left(\varphi(x)_j > 0\right) \\
&= \prod_{j\in J} \mathbb{P}\left(M_j v_j > 0\right) - \prod_{j\in J} \mathbb{P}_c\left(M_j v_j > 0\right) && \text{(since } \varphi(x)_j = M_j v_j\text{)} \\
&= \prod_{j\in J} \mathbb{P}\left(M_j > 0\right) - \prod_{j\in J} \mathbb{P}_c\left(M_j > 0\right) && \text{(since } v_j > 0 \text{ for all } j \in [d]\text{)} \\
&= \text{const} - \prod_{j\in J} \left(1 - \mathbb{P}_c\left(M_j = 0\right)\right) \\
&= \text{const} - \prod_{j\in J} \left(1 - (1-p)^{m_j - c_j}\right) \\
&= \text{const} - \prod_{j\in J} \left(1 - (1-p)^{m_j - c_j}\right).
\end{aligned}
$$

Thus, the following conditions are equivalent.

$$
\begin{aligned}
\text{Maximize} \quad \Delta_c &\iff \text{Maximize} \quad \text{const} - \prod_{j\in J} \left(1 - (1-p)^{m_j - c_j}\right) \\
&\iff \text{Minimize} \quad \prod_{j\in J} \left(1 - p^{m_j - c_j}\right) && \text{(we use } p = 1/2\text{)} \\
&\iff \text{Minimize} \quad \prod_{j\in J} \left(1 - p^{m_j - c_j}\right) && \text{(B.1)} \\
&\iff \text{Minimize} \quad \log\left(\prod_{j\in J} \left(1 - p^{m_j - c_j}\right)\right) \\
&\iff \text{Minimize} \quad \sum_{j\in J} \log\left(1 - p^{m_j - c_j}\right). && \text{(B.2)}
\end{aligned}
$$

Let us first consider Eq. (B.1). We study the problem for a length $l \leq l_{max}$ of the candidates, *i.e.*, the problem is

$$\begin{aligned} \text{Minimize} \quad & G(c) := \prod_{j \in J} (1 - p^{m_j - c_j}) \\ \text{subject to} \quad & \sum_{j=1}^{d} c_j = l \\ \text{and} \quad & c_j \in [m_j]\,, \quad j \in [d]\,. \end{aligned} \tag{B.3}$$

$G(c) = 0$ is the global minimum. We split the proof in three cases: (1) $l = m_1$, (2) $l > m_1$, and (3) $l < m_1$.

**(1)** $l = m_1$**.** When $l$ is equal to the smallest multiplicity, the optimal candidate $c^\star$ is such that $c^\star_1 = m_1$ and $c^\star_j = 0$ for $j \neq 1$. Indeed, $G(c^\star) = 0 < G(c)$ for any $c \neq c^\star$ such that $|c| = |c^\star|$.

**(2)** $l > m_1$**.** The previous paragraph implies that the optimal candidate $c^\star$ is always such that $|c^\star| \leq m_1$. Indeed, any candidate $c$ of size $l$ such that $G(c) = G(c^\star) = 0$ has size $|c| > m_1 \geq |c^\star|$. Hence, we can disregard this case.

**(3)** $l < m_1$**.** The optimal candidate is $c^\star$ with $c^\star_1 = l$ and $c^\star_j = 0$ for $j \neq 1$. To prove this, let us study the continuous problem from Eq. (B.2):

$$\begin{aligned} \text{Minimize} \quad & G(c) := \sum_{j \in J} \log\left(1 - p^{m_j - x_j}\right) \\ \text{subject to} \quad & \sum_{j=1}^{d} x_j = l \\ \text{and} \quad & 0 \leq x_j \leq m_j\,, \quad j \in [d]\,. \end{aligned} \tag{B.4}$$

First, we notice that Problem (B.4) consists of minimizing a concave function over a convex set, indeed, since the problem is separable the Hessian matrix $H$ of $F$ is diagonal and defined, for $i, j \in [d]$, as

$$(H)_{i,j} = \begin{cases} -\frac{(\log(p))^2 p^{m_j - x}}{(p^x - p^{m_j})^2} < 0\,, & \text{if } i = j\,, \\ 0\,, & \text{if } i \neq j\,. \end{cases}$$

As a consequence, the solutions are found on the corners (see Corollary 32.3.1, Rockafellar [1997]), defined as $(l, 0, \ldots, 0)$, $(0, l, \ldots, 0)$, …, $(0, 0, \ldots, l)$. Finally, we notice that the candidate $c$ minimizing Eq. (B.4) is such that $m_j - c_j = m_j - l$ is minima, *i.e.*, $c_1 = l$ and $c_i = 0$ for $i \in [d]$. □

## B.2 Experiments

In this section, we report details of experiments and additional results omitted in Chapter 5 due to space constraints.

### B.2.1 Setting

All the experiments reported in this Section and in Chapter 5 are implemented in `Python` and ran on GPU `Nvidia A100`. The code for FRED and the experiments is available at https://github.com/gianluigilopardo/fred.

**Datasets.** We employ three sentiment analysis datasets: Restaurants [*], Yelp reviews [†], and IMDb [‡], and the Tweets hate speech detection [§] dataset. Note that the datasets have a substantial difference in document length: while the average length of tokens in Restaurants and Tweets is about 10, for Yelp it is about 150, in IMDb it is 230. As shown below, this difference has a relevant impact on explainers' behavior.

**Models.** We trained a logistic classifier, a decision tree, and a random forest classifier on each dataset (with the default parameters of https://scikit-learn.org/). Additionally, we employed a pretrained version of RoBERTa [¶] [Hartmann et al., 2023] and DistilBERT [‖] for the three sentiment analysis datasets. These are two cutting-edge transformer models for sentiment analysis sourced from the Hugging Face repository. Finally, we remark that we always consider documents with positive predictions, and we explain the positive class.

Table B.1 reports the accuracies for each model and dataset. Remark that logistic classifiers, decision trees, random forest classifiers have been trained on each dataset according to their task. Contrarily, we use pre-trained version of Roberta and DistilBERT on the datasets.

**Explainers.** FRED is compared to three well-known explainers: Anchors, SHAP, and LIME. We use the official implementation (respectively available and licensed at https://github.com/marcotcr/anchor, https://github.com/shap/shap, https://github.com/marcotcr/lime) of these methods with all default parameters.

Remark that while LIME and SHAP share the principle of attributing an importance score for each token, Anchor aim at outputting the most significant subset of tokens (while FRED has both the options). We apply the following scheme to perform a fair evaluation. We use as explanation set $E$ for LIME and SHAP the first $k$ tokens ranked by importance, where $k$ is the size of FRED's explanation (`fredpos`) for the same example. This is used for computing comprehensiveness, sufficiency, robustness, and proportion. Conversely, for computing the AUC-MoRF of Anchors we select the tokens ordered as in the anchor.

Finally, the metrics are computed over the first 100 instances ranked by length of positively classified documents from each test set.

### B.2.2 Additional experimental results

We report below additional experimental results omitted in Chapter 5 due to space constraints. Again, both FRED-mask and FRED-pos produce more faithful explanations than the others. In

∗. https://www.kaggle.com/hj5992/restaurantreviews
†. https://www.kaggle.com/omkarsabnis/yelp-reviews-dataset
‡. https://huggingface.co/datasets/imdb
§. https://huggingface.co/datasets/tweets_hate_speech_detection
¶. https://huggingface.co/siebert/sentiment-roberta-large-english
‖. https://huggingface.co/distilbert/distilbert-base-uncased

TABLE B.1 – Accuracy of machine learning models evaluated on datasets used in the experiments. An asterisk (*) denotes models that were trained specifically on the corresponding dataset. RoBERTa and DistilBERT are leveraged here as pre-trained models, meaning their weights were not further fine-tuned on the individual datasets.

| Dataset | Model | Accuracy |
|---|---|---|
| Restaurants | Logistic Regression* | 0.808 |
| | Decision Tree* | 0.676 |
| | Random Forest* | 0.748 |
| | RoBERTa | 0.972 |
| | DistilBERT | 0.500 |
| Yelp | Logistic Regression* | 0.892 |
| | Decision Tree* | 0.748 |
| | Random Forest* | 0.868 |
| | RoBERTa | 0.988 |
| | DistilBERT | 0.472 |
| IMDB | Logistic Regression* | 0.710 |
| | Decision Tree* | 0.700 |
| | Random Forest* | 0.834 |
| | RoBERTa | 0.950 |
| | DistilBERT | 0.500 |
| Tweets | Logistic Regression* | 0.930 |
| | Decision Tree* | 0.941 |
| | Random Forest* | 0.959 |
| | RoBERTa | 0.273 |
| | DistilBERT | 0.929 |

general, Anchor performs well on small documents (Restaurant dataset), while its behavior on larger documents is highly nonlinear, as it tends to be conservative on the size of the anchor. LIME and SHAP have a close behavior, as somehow expected, but the latter is significantly more efficient. On small documents, SHAP performs an exhaustive search over all possible token subsets, motivating its high robustness.

TABLE B.2 – Comparison on a logistic classifier for Restaurant reviews ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | −0.118(0.09) | 0.130(0.05) | 0.866(0.20) | 0.709(0.09) | 0.055(0.02) | 0.227(0.13) |
| fredpos | −0.094(0.11) | 0.097(0.06) | 0.876(0.21) | 0.721(0.09) | 0.059(0.02) | 0.159(0.09) |
| lime | −0.111(0.09) | 0.102(0.06) | 0.968(0.10) | 0.726(0.09) | 0.130(0.02) | 0.159(0.09) |
| shap | −0.120(0.09) | 0.100(0.06) | 1.000(0.00) | 0.712(0.09) | 0.065(0.37) | 0.159(0.09) |
| anchor | −0.058(0.15) | 0.072(0.06) | 0.772(0.30) | 0.874(0.10) | 0.121(0.19) | 0.136(0.07) |

TABLE B.3 – Comparison on a logistic classifier for Restaurant reviews ($p = 0.5$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | −0.119(0.09) | 0.120(0.05) | 0.926(0.15) | 0.707(0.09) | 0.075(0.03) | 0.216(0.14) |
| fredpos | −0.089(0.11) | 0.086(0.07) | 0.924(0.20) | 0.718(0.09) | 0.075(0.02) | 0.128(0.07) |
| lime | −0.105(0.10) | 0.093(0.06) | 0.982(0.09) | 0.726(0.09) | 0.128(0.02) | 0.128(0.07) |
| shap | −0.114(0.09) | 0.092(0.06) | 1.000(0.00) | 0.712(0.09) | 0.065(0.36) | 0.128(0.07) |
| anchor | −0.054(0.15) | 0.069(0.07) | 0.784(0.30) | 0.874(0.10) | 0.122(0.19) | 0.127(0.07) |

TABLE B.4 – Comparison on a decision tree for Restaurant reviews ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.030(0.17) | 0.940(0.24) | 0.960(0.14) | 0.138(0.14) | 0.057(0.02) | 0.158(0.09) |
| fredpos | 0.040(0.20) | 0.750(0.43) | 0.947(0.15) | 0.183(0.16) | 0.062(0.02) | 0.137(0.08) |
| lime | 0.020(0.14) | 0.750(0.43) | 0.894(0.21) | 0.110(0.09) | 0.132(0.01) | 0.137(0.08) |
| shap | 0.010(0.10) | 0.690(0.46) | 1.000(0.00) | 0.132(0.11) | 0.054(0.20) | 0.137(0.08) |
| anchor | 0.020(0.14) | 0.630(0.48) | 0.968(0.15) | 0.381(0.45) | 0.190(0.76) | 0.111(0.05) |

TABLE B.5 – Comparison on a decision tree for Restaurant reviews ($p = 0.5$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.020(0.14) | 0.620(0.49) | 0.807(0.28) | 0.116(0.09) | 0.067(0.02) | 0.116(0.05) |
| fredpos | 0.030(0.17) | 0.650(0.48) | 0.938(0.17) | 0.167(0.17) | 0.080(0.03) | 0.113(0.05) |
| lime | 0.020(0.14) | 0.650(0.48) | 0.822(0.27) | 0.110(0.09) | 0.132(0.01) | 0.113(0.05) |
| shap | 0.010(0.10) | 0.600(0.49) | 1.000(0.00) | 0.132(0.11) | 0.051(0.15) | 0.113(0.05) |
| anchor | 0.020(0.14) | 0.630(0.48) | 0.968(0.15) | 0.381(0.45) | 0.190(0.76) | 0.111(0.05) |

TABLE B.6 – Comparison on a random forest classifier for Restaurant reviews ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.040(0.15) | 0.192(0.09) | 0.919(0.17) | 0.562(0.13) | 0.134(0.02) | 0.160(0.07) |
| fredpos | 0.084(0.17) | 0.156(0.11) | 0.942(0.18) | 0.544(0.11) | 0.140(0.02) | 0.126(0.06) |
| lime | 0.061(0.14) | 0.157(0.11) | 0.974(0.12) | 0.516(0.10) | 0.218(0.01) | 0.126(0.06) |
| shap | 0.054(0.13) | 0.150(0.11) | 1.000(0.00) | 0.495(0.09) | 0.172(0.21) | 0.126(0.06) |
| anchor | 0.069(0.15) | 0.129(0.13) | 0.877(0.24) | 0.778(0.18) | 0.733(0.88) | 0.118(0.05) |

TABLE B.7 – Comparison on a random forest classifier for Restaurant reviews ($p = 0.5$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.047(0.15) | 0.162(0.12) | 0.955(0.15) | 0.512(0.11) | 0.219(0.03) | 0.146(0.08) |
| fredpos | 0.072(0.15) | 0.156(0.12) | 0.942(0.18) | 0.523(0.10) | 0.233(0.03) | 0.138(0.08) |
| lime | 0.063(0.15) | 0.162(0.12) | 0.980(0.09) | 0.516(0.09) | 0.329(0.02) | 0.138(0.08) |
| shap | 0.055(0.14) | 0.157(0.12) | 1.000(0.00) | 0.500(0.09) | 0.187(0.27) | 0.138(0.08) |
| anchor | 0.075(0.15) | 0.139(0.13) | 0.862(0.27) | 0.770(0.17) | 0.837(0.98) | 0.135(0.08) |

TABLE B.8 – Comparison on DistilBERT for Restaurant reviews ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | −0.017(0.01) | 0.003(0.00) | 0.910(0.17) | 0.997(0.01) | 7.273(0.40) | 0.145(0.17) |
| fredpos | −0.014(0.01) | −0.004(0.01) | 0.571(0.26) | 1.002(0.01) | 7.271(0.32) | 0.363(0.23) |
| lime | −0.018(0.01) | 0.001(0.00) | 0.943(0.17) | 0.998(0.01) | 7.806(0.34) | 0.164(0.16) |
| shap | −0.014(0.01) | −0.004(0.01) | 1.000(0.00) | 1.015(0.01) | 0.183(0.20) | 0.363(0.23) |
| anchor | −0.016(0.01) | −0.003(0.00) | 1.000(0.00) | 1.008(0.01) | 0.500(0.23) | 0.121(0.07) |

TABLE B.9 – Comparison on DistilBERT for Restaurant reviews ($p = 0.5$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.003(0.01) | 0.023(0.01) | 0.961(0.08) | 0.970(0.01) | 7.331(0.40) | 0.700(0.18) |
| fredpos | 0.007(0.01) | 0.015(0.01) | 0.882(0.15) | 0.978(0.02) | 7.372(0.30) | 0.461(0.32) |
| lime | 0.006(0.01) | 0.017(0.01) | 0.982(0.08) | 0.971(0.01) | 7.833(0.53) | 0.455(0.31) |
| shap | 0.007(0.01) | 0.015(0.01) | 1.000(0.00) | 0.974(0.01) | 0.180(0.20) | 0.461(0.32) |
| anchor | 0.014(0.01) | 0.006(0.01) | 0.912(0.25) | 0.988(0.01) | 0.981(1.31) | 0.130(0.09) |

TABLE B.10 – Comparison on Roberta for Restaurant reviews ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.487(0.49) | 0.766(0.42) | 0.916(0.20) | 0.151(0.12) | 35.276(1.48) | 0.158(0.08) |
| fredpos | 0.577(0.49) | 0.528(0.50) | 0.953(0.18) | 0.230(0.18) | 34.953(1.49) | 0.130(0.06) |
| lime | 0.507(0.49) | 0.538(0.50) | 0.930(0.21) | 0.162(0.13) | 38.554(2.36) | 0.130(0.06) |
| shap | 0.756(0.42) | 0.323(0.46) | 1.000(0.00) | 0.306(0.21) | 1.814(1.58) | 0.130(0.06) |
| anchor | 0.509(0.49) | 0.538(0.50) | 0.850(0.32) | 0.454(0.45) | 15.556(20.26) | 0.126(0.06) |

TABLE B.11 – Comparison on logistic classifier for Yelp reviews ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | −0.200(0.09) | 0.101(0.03) | 0.551(0.27) | 0.722(0.07) | 0.251(0.15) | 0.088(0.07) |
| fredpos | −0.193(0.09) | 0.088(0.03) | 0.571(0.29) | 0.721(0.07) | 0.266(0.16) | 0.075(0.06) |
| lime | −0.207(0.09) | 0.094(0.03) | 0.897(0.18) | 0.790(0.07) | 0.280(0.12) | 0.075(0.06) |
| shap | −0.209(0.09) | 0.086(0.03) | 1.000(0.00) | 0.748(0.07) | 0.188(0.49) | 0.075(0.06) |
| anchor | −0.012(0.23) | 0.034(0.05) | 0.658(0.37) | 0.948(0.07) | 0.983(2.53) | 0.041(0.05) |

TABLE B.12 – Comparison on logistic classifier for Yelp reviews ($p = 0.5$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | −0.205(0.09) | 0.103(0.02) | 0.749(0.23) | 0.724(0.07) | 0.355(0.21) | 0.090(0.08) |
| fredpos | −0.188(0.10) | 0.068(0.04) | 0.754(0.29) | 0.718(0.07) | 0.398(0.24) | 0.069(0.09) |
| lime | −0.196(0.10) | 0.067(0.04) | 0.866(0.23) | 0.790(0.07) | 0.287(0.12) | 0.060(0.07) |
| shap | −0.188(0.10) | 0.066(0.04) | 1.000(0.00) | 0.747(0.08) | 0.171(0.30) | 0.069(0.09) |
| anchor | −0.031(0.23) | 0.037(0.05) | 0.629(0.39) | 0.945(0.07) | 1.123(2.85) | 0.043(0.06) |

TABLE B.13 – Comparison on decision tree for Yelp reviews ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.240(0.43) | 0.840(0.37) | 0.899(0.23) | 0.144(0.27) | 0.292(0.17) | 0.042(0.05) |
| fredpos | 0.250(0.43) | 0.810(0.39) | 0.842(0.31) | 0.194(0.33) | 0.320(0.18) | 0.041(0.05) |
| lime | 0.200(0.40) | 0.790(0.41) | 0.936(0.18) | 0.105(0.19) | 0.324(0.14) | 0.041(0.05) |
| shap | 0.180(0.38) | 0.730(0.44) | 0.966(0.12) | 0.143(0.23) | 0.177(0.15) | 0.041(0.05) |
| anchor | 0.240(0.43) | 0.560(0.50) | 0.837(0.31) | 0.477(0.48) | 9.167(47.26) | 0.028(0.03) |

TABLE B.14 – Comparison on decision tree for Yelp reviews ($p = 0.5$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.200(0.40) | 0.560(0.50) | 0.811(0.28) | 0.102(0.16) | 0.357(0.21) | 0.030(0.03) |
| fredpos | 0.190(0.39) | 0.570(0.50) | 0.818(0.29) | 0.122(0.19) | 0.445(0.27) | 0.030(0.03) |
| lime | 0.200(0.40) | 0.580(0.49) | 0.830(0.26) | 0.105(0.19) | 0.328(0.14) | 0.030(0.03) |
| shap | 0.180(0.38) | 0.550(0.50) | 0.972(0.11) | 0.143(0.23) | 0.175(0.15) | 0.030(0.03) |
| anchor | 0.250(0.43) | 0.560(0.50) | 0.840(0.31) | 0.477(0.48) | 9.164(47.25) | 0.028(0.03) |

TABLE B.15 – Comparison on random forest classifier for Yelp reviews ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | −0.126(0.11) | 0.107(0.04) | 0.653(0.28) | 0.749(0.10) | 0.306(0.13) | 0.081(0.08) |
| fredpos | −0.107(0.12) | 0.100(0.04) | 0.634(0.30) | 0.753(0.09) | 0.328(0.14) | 0.076(0.09) |
| lime | −0.138(0.10) | 0.102(0.04) | 0.909(0.17) | 0.782(0.08) | 0.345(0.13) | 0.074(0.08) |
| shap | −0.159(0.09) | 0.086(0.05) | 0.968(0.09) | 0.776(0.08) | 0.640(0.25) | 0.076(0.09) |
| anchor | −0.050(0.16) | 0.032(0.05) | 0.757(0.33) | 0.942(0.09) | 2.522(6.23) | 0.040(0.04) |

TABLE B.16 – Comparison on DistilBERT for Yelp reviews ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | −0.011(0.01) | 0.012(0.01) | 0.477(0.19) | 0.983(0.02) | 11.324(3.49) | 0.141(0.12) |
| fredpos | −0.011(0.01) | 0.007(0.01) | 0.359(0.16) | 0.986(0.01) | 11.816(3.69) | 0.146(0.13) |
| lime | −0.012(0.01) | 0.008(0.01) | 0.837(0.21) | 0.986(0.01) | 16.691(6.84) | 0.125(0.14) |
| shap | −0.011(0.01) | 0.005(0.01) | 1.000(0.00) | 0.994(0.01) | 2.827(1.16) | 0.146(0.13) |
| anchor | −0.007(0.01) | 0.000(0.00) | 0.865(0.32) | 0.999(0.01) | 1.452(1.11) | 0.038(0.04) |

TABLE B.17 – Comparison on DistilBERT for Yelp reviews ($p = 0.5$, $n = 70$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.003(0.01) | 0.019(0.01) | 0.517(0.13) | 0.966(0.01) | 16.013(4.38) | 0.222(0.14) |
| fredpos | 0.007(0.01) | 0.017(0.01) | 0.425(0.11) | 0.975(0.02) | 16.320(4.36) | 0.232(0.17) |
| lime | 0.004(0.01) | 0.017(0.01) | 0.799(0.15) | 0.973(0.01) | 25.053(9.54) | 0.197(0.14) |
| shap | 0.006(0.01) | 0.016(0.01) | 1.000(0.00) | 0.973(0.01) | 4.612(1.58) | 0.232(0.17) |
| anchor | 0.003(0.01) | 0.011(0.01) | 0.520(0.29) | 0.981(0.01) | 184.020(228.83) | 0.134(0.13) |

TABLE B.18 – Comparison on Roberta for Yelp reviews ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.145(0.33) | 0.180(0.38) | 0.419(0.29) | 0.656(0.36) | 44.358(7.37) | 0.096(0.09) |
| fredpos | 0.170(0.34) | 0.100(0.30) | 0.393(0.38) | 0.716(0.33) | 45.003(8.14) | 0.065(0.07) |
| lime | 0.103(0.28) | 0.100(0.30) | 0.601(0.34) | 0.676(0.39) | 69.072(21.34) | 0.065(0.07) |
| shap | 0.244(0.40) | 0.031(0.17) | 1.000(0.00) | 0.770(0.31) | 8.455(3.40) | 0.065(0.07) |
| anchor | 0.419(0.43) | 0.050(0.22) | 0.626(0.43) | 0.926(0.25) | 22.924(47.58) | 0.033(0.04) |

TABLE B.19 – Comparison on logistic classifier for IMDb ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | −0.122(0.05) | 0.045(0.02) | 0.465(0.11) | 0.873(0.05) | 0.194(0.04) | 0.151(0.08) |
| fredpos | −0.117(0.05) | 0.043(0.02) | 0.416(0.09) | 0.872(0.05) | 0.211(0.04) | 0.146(0.09) |
| lime | −0.141(0.05) | 0.048(0.02) | 0.947(0.10) | 0.908(0.03) | 0.204(0.03) | 0.142(0.09) |
| shap | −0.141(0.05) | 0.047(0.02) | 1.000(0.00) | 0.876(0.05) | 0.202(0.30) | 0.146(0.09) |
| anchor | −0.090(0.09) | 0.020(0.02) | 0.591(0.35) | 0.964(0.03) | 0.671(0.74) | 0.056(0.04) |

TABLE B.20 – Comparison on logistic classifier for IMDb ($p = 0.5$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | −0.103(0.04) | 0.066(0.02) | 0.595(0.13) | 0.876(0.05) | 0.603(0.12) | 0.294(0.13) |
| fredpos | −0.097(0.04) | 0.063(0.02) | 0.539(0.10) | 0.874(0.05) | 0.687(0.14) | 0.296(0.12) |
| lime | −0.138(0.05) | 0.052(0.02) | 0.950(0.10) | 0.911(0.03) | 0.375(0.05) | 0.175(0.10) |
| shap | −0.112(0.04) | 0.066(0.02) | 1.000(0.00) | 0.880(0.04) | 0.336(0.44) | 0.296(0.12) |
| anchor | −0.099(0.08) | 0.020(0.02) | 0.622(0.31) | 0.964(0.03) | 1.213(1.53) | 0.057(0.04) |

TABLE B.21 – Comparison on decision tree for IMDb ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.250(0.43) | 0.690(0.46) | 0.792(0.29) | 0.174(0.24) | 0.145(0.03) | 0.059(0.04) |
| fredpos | 0.270(0.44) | 0.530(0.50) | 0.749(0.35) | 0.341(0.36) | 0.160(0.03) | 0.048(0.03) |
| lime | 0.200(0.40) | 0.550(0.50) | 0.820(0.27) | 0.091(0.08) | 0.199(0.02) | 0.048(0.03) |
| shap | 0.160(0.37) | 0.480(0.50) | 0.933(0.19) | 0.146(0.19) | 0.185(0.13) | 0.048(0.03) |
| anchor | 0.150(0.36) | 0.460(0.50) | 0.777(0.30) | 0.542(0.48) | 1.946(8.13) | 0.035(0.01) |

TABLE B.22 – Comparison on decision tree for IMDb ($p = 0.5$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.160(0.37) | 0.440(0.50) | 0.736(0.30) | 0.098(0.09) | 0.172(0.03) | 0.039(0.02) |
| fredpos | 0.220(0.41) | 0.420(0.49) | 0.855(0.26) | 0.304(0.34) | 0.213(0.04) | 0.035(0.02) |
| lime | 0.190(0.39) | 0.470(0.50) | 0.726(0.31) | 0.091(0.08) | 0.200(0.02) | 0.035(0.02) |
| shap | 0.180(0.38) | 0.440(0.50) | 0.921(0.22) | 0.146(0.19) | 0.182(0.14) | 0.035(0.02) |
| anchor | 0.180(0.38) | 0.460(0.50) | 0.775(0.31) | 0.542(0.48) | 1.972(8.24) | 0.033(0.01) |

TABLE B.23 – Comparison on random forest classifier for IMDb ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | −0.141(0.09) | 0.091(0.03) | 0.489(0.18) | 0.840(0.09) | 0.319(0.03) | 0.139(0.10) |
| fredpos | −0.136(0.09) | 0.069(0.03) | 0.401(0.23) | 0.847(0.08) | 0.324(0.03) | 0.122(0.10) |
| lime | −0.163(0.09) | 0.082(0.03) | 0.904(0.16) | 0.861(0.06) | 0.363(0.02) | 0.111(0.07) |
| shap | −0.173(0.10) | 0.072(0.03) | 0.961(0.11) | 0.821(0.08) | 0.822(0.22) | 0.122(0.10) |
| anchor | −0.095(0.11) | 0.007(0.03) | 0.813(0.32) | 0.987(0.06) | 0.995(2.98) | 0.043(0.03) |

TABLE B.24 – Comparison on random forest classifier for IMDb ($p = 0.5$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | −0.175(0.09) | 0.124(0.04) | 0.582(0.17) | 0.904(0.11) | 0.780(0.14) | 0.299(0.14) |
| fredpos | −0.141(0.10) | 0.077(0.05) | 0.639(0.25) | 0.870(0.10) | 0.736(0.16) | 0.215(0.19) |
| lime | −0.158(0.11) | 0.076(0.04) | 0.916(0.15) | 0.861(0.06) | 0.589(0.04) | 0.113(0.09) |
| shap | −0.167(0.11) | 0.086(0.05) | 0.972(0.06) | 0.821(0.08) | 1.390(0.35) | 0.215(0.19) |
| anchor | −0.093(0.11) | 0.008(0.03) | 0.807(0.32) | 0.987(0.06) | 1.719(5.10) | 0.046(0.04) |

TABLE B.25 – Comparison on DistilBERT for IMDb ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | −0.011(0.01) | 0.007(0.00) | 0.460(0.18) | 0.993(0.01) | 9.092(0.73) | 0.155(0.09) |
| fredpos | −0.011(0.01) | 0.004(0.00) | 0.344(0.15) | 0.990(0.01) | 9.346(0.92) | 0.167(0.08) |
| lime | −0.014(0.01) | 0.005(0.00) | 0.872(0.18) | 0.991(0.01) | 11.029(1.29) | 0.119(0.07) |
| shap | −0.012(0.01) | 0.002(0.01) | 1.000(0.00) | 0.995(0.01) | 1.827(0.56) | 0.167(0.08) |
| anchor | −0.008(0.01) | −0.001(0.00) | 0.990(0.10) | 1.001(0.01) | 0.645(0.09) | 0.044(0.03) |

TABLE B.26 – Comparison on Roberta for IMDb ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.278(0.44) | 0.488(0.49) | 0.635(0.35) | 0.353(0.36) | 45.841(3.75) | 0.090(0.06) |
| fredpos | 0.417(0.48) | 0.266(0.44) | 0.674(0.35) | 0.454(0.35) | 46.691(4.01) | 0.060(0.03) |
| lime | 0.210(0.40) | 0.278(0.44) | 0.855(0.28) | 0.295(0.33) | 65.123(7.64) | 0.060(0.03) |
| shap | 0.436(0.48) | 0.159(0.36) | 1.000(0.00) | 0.475(0.32) | 15.901(5.05) | 0.060(0.03) |
| anchor | 0.453(0.48) | 0.228(0.42) | 0.642(0.41) | 0.777(0.40) | 54.867(112.66) | 0.038(0.02) |

TABLE B.27 – Comparison on decision tree for tweets hate speech detection ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.870(0.34) | 0.950(0.22) | 0.982(0.10) | 0.078(0.04) | 0.046(0.01) | 0.143(0.05) |
| fredpos | 0.880(0.32) | 1.000(0.00) | 0.952(0.15) | 0.078(0.04) | 0.052(0.01) | 0.148(0.05) |
| lime | 0.870(0.34) | 0.990(0.10) | 0.907(0.21) | 0.082(0.06) | 0.125(0.01) | 0.148(0.05) |
| shap | 0.880(0.32) | 0.970(0.17) | 1.000(0.00) | 0.088(0.08) | 0.035(0.18) | 0.148(0.05) |
| anchor | 0.880(0.32) | 0.850(0.36) | 0.669(0.39) | 0.109(0.14) | 0.578(0.47) | 0.144(0.05) |

TABLE B.28 – Comparison on random forest classifier for tweets hate speech detection ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | 0.782(0.21) | 0.391(0.15) | 0.976(0.09) | 0.162(0.06) | 0.273(0.06) | 0.108(0.03) |
| fredpos | 0.784(0.20) | 0.388(0.15) | 0.985(0.09) | 0.166(0.06) | 0.250(0.07) | 0.109(0.03) |
| lime | 0.784(0.20) | 0.385(0.14) | 0.974(0.14) | 0.159(0.05) | 0.453(0.05) | 0.109(0.03) |
| shap | 0.784(0.20) | 0.384(0.15) | 1.000(0.00) | 0.156(0.05) | 0.153(0.14) | 0.109(0.03) |
| anchor | 0.788(0.20) | 0.281(0.19) | 0.493(0.40) | 0.221(0.10) | 7.841(4.00) | 0.109(0.03) |

TABLE B.29 – Comparison on DistilBERT for tweets hate speech detection ($p = 0.1$, $\varepsilon = 0.15$).

| | suffic. ↓ | compreh. ↑ | robust. ↑ | aucmorf ↓ | time (s) ↓ | proport. ↓ |
|---|---|---|---|---|---|---|
| fred | −0.002(0.01) | 0.019(0.01) | 0.691(0.21) | 0.974(0.01) | 8.309(0.88) | 0.411(0.10) |
| fredpos | 0.001(0.01) | 0.012(0.01) | 0.460(0.16) | 0.978(0.01) | 8.392(0.91) | 0.450(0.13) |
| lime | −0.005(0.01) | 0.016(0.01) | 0.979(0.07) | 0.972(0.01) | 9.414(1.11) | 0.427(0.10) |
| shap | −0.003(0.01) | 0.013(0.01) | 1.000(0.00) | 0.978(0.01) | 0.497(0.33) | 0.450(0.13) |
| anchor | −0.002(0.01) | 0.012(0.01) | 0.891(0.20) | 0.978(0.01) | 11.732(11.18) | 0.307(0.17) |

# APPENDIX C

# Appendix for Chapter 6: *Attention Meets Post-hoc Interpretability*

**Organization of the Appendix.** The appendix begins with proofs for the theoretical results presented in Chapter 6. Theorems 6.5.1 and 6.6.1 are proven in Sections C.1 and C.2, respectively. Sections C.3 and C.4 provide additional technical details crucial for the proofs. Finally, Section C.6 details the model used for the experiments. Further information, including the training and experimental code, is available at https://github.com/gianluigilopardo/attention_meets_xai.

## C.1 Proof of Theorem 6.5.1

In this section, we show how to compute the gradient of $f$ with respect to the embedding $e_t$, $t \in [T]$. By linearity, we can focus on one head $f_i$, $i \in [K]$, and thus we momentarily drop the $i$ superscripts. Let us start by computing the gradients of the key, query, and value vectors $k_t$, $q_t$, and $v_t$ (Eqs. (6.5), (6.6), and (6.7)). For any $t \in [T]$, one has

$$\nabla_{e_t} k_t = \nabla_{e_t} (W_k e_t) = W_k (\nabla_{e_t} e_t) = W_k^\top \in \mathbb{R}^{d_e \times d_{\text{att}}}, \tag{C.1}$$

$$\nabla_{e_t} q_t = \nabla_{e_t} (W_q e_t) = W_q (\nabla_{e_t} e_t) = W_q^\top \in \mathbb{R}^{d_e \times d_{\text{att}}}, \tag{C.2}$$

and

$$\nabla_{e_t} v_t = \nabla_{e_t} (W_v e_t) = W_v^\top \in \mathbb{R}^{d_e \times d_{\text{out}}}. \tag{C.3}$$

Therefore, the gradient of the attention $\alpha_t$ as defined in Eq. (6.8), for any $t \in [T]$,

$$\begin{aligned} \nabla_{e_t} \alpha_t &= \nabla_{e_t} \left( \frac{\exp\left(q^\top k_t / \sqrt{d_{\text{att}}}\right)}{\sum_{u=1}^{T_{max}} \exp\left(q^\top k_u / \sqrt{d_{\text{att}}}\right)} \right) \\ &= \frac{\alpha_t}{\sqrt{d_{\text{att}}}} \left( (W_k^\top q) - \sum_{s=1}^{T_{\max}} \alpha_s (W_k^\top q) \right) \quad \in \mathbb{R}^{d_e}. \end{aligned}$$

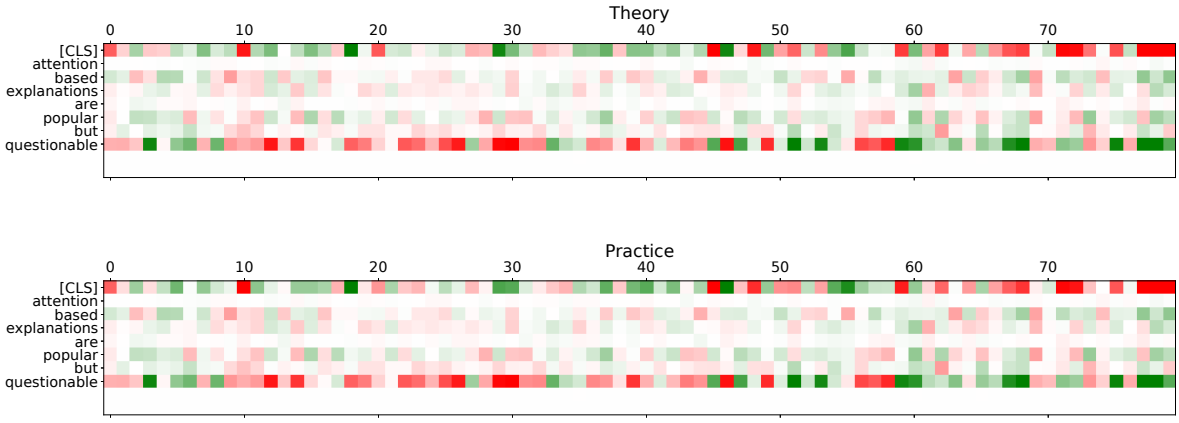


Figure C.1 – **Illustration of the accuracy of Theorem 6.5.1.** For illustrative purpose, $d_e = 80$.

The situation is similar if we look at another attention coefficient: let $s \neq t$, then

$$\begin{aligned}\nabla_{e_t}\alpha_s &= \nabla_{e_t}\left(\frac{\exp\left(q^\top k_s/\sqrt{d_{\text{att}}}\right)}{\sum_{u=1}^{T_{max}}\exp\left(q^\top k_u/\sqrt{d_{\text{att}}}\right)}\right)\\ &= \frac{-\alpha_u\alpha_t}{\sqrt{d_{\text{att}}}}(W_k^\top q)\,.\end{aligned}$$

Finally, we can compute the gradient of $\tilde{v}$ as

$$\begin{aligned}\nabla_{e_t}\tilde{v} &= \nabla_{e_t}\left(\sum_{u=1}^{T_{\max}}\alpha_u v_u\right)\\ &= \nabla_{e_t}(\alpha_t)v_t + \alpha_t(\nabla_{e_t}v_t) + \sum_{s\neq t}(\nabla_{e_t}\alpha_s)v_s\\ &= \frac{1}{\sqrt{d_{\text{att}}}}(W_k^\top q)(\alpha_t - \alpha_t^2)v_t + \alpha_t W_v^\top + \sum_{s\neq t}\frac{-\alpha_u\alpha_t}{\sqrt{d_{\text{att}}}}(W_k^\top q)\\ \nabla_{e_t} &= \frac{\alpha_t}{\sqrt{d_{\text{att}}}}\left(v_t - \sum_{s=1}^{T_{\max}}\alpha_s v_s\right)(W_k^\top q) + \alpha_t W_v^\top\,.\end{aligned}$$

Finally, since $f(x) = W_\ell\tilde{v}$, we deduce Eq. (6.14) from the last display, multiplying by $W_\ell^\top$ and averaging. □

Theorem 6.5.1 is also true in practice, as illustrated in Figure C.1.

## C.2 Proof of Theorem 6.6.1

**Preliminaries.** The key idea of this proof is to leverage Eq. (6.17) and find a good approximation for the conditional expectations involved. Looking closer at Eq. (6.17), we first notice that, by linearity, we can focus on the limit coefficients associated to a single head. Thus we drop the $i$ indexation in this proof.

Now let us recall that $S$ is the random subset of words from the dictionary being removed when generating $X$. Our first key observation is that $X$ **has random token embeddings** $E_t$. More

precisely, Eq. (6.2) becomes

$$\forall t \in [T_{\max}], \quad E_t := e_t \mathbb{1}_{\xi_t \notin S} + (h + W_p(t))\mathbb{1}_{\xi_t \in S} \,. \tag{C.4}$$

We note that $E_t$ for $t > T$ is actually not random (LIME does not perturb outside of $\xi$), but this will be of no consequence. In turn, keys and queries are modified, that is, Eqs. (6.5) and (6.6) become, respectively,

$$\forall t \in [T_{\max}], \quad K_t := k_t \mathbb{1}_{\xi_t \notin S} + W_k(h + W_p(t))\mathbb{1}_{\xi_t \in S} \,, \tag{C.5}$$

and

$$\forall t \in [T_{\max}], \quad Q_t := q_t \mathbb{1}_{\xi_t \notin S} + W_q(h + W_p(t))\mathbb{1}_{\xi_t \in S} \,. \tag{C.6}$$

The attention coefficients associated to the `[CLS]` token also become random. In analogous fashion to Eq. (6.18), let us define

$$\forall u \in [T_{\max}], \quad G_u := \exp\left(q^\top K_u / \sqrt{d_{\text{att}}}\right) \,,$$

where we recall that $q \in \mathbb{R}^{d_{\text{att}}}$ is the query vector associated to the `[CLS]` token. Taking dot product and exponential, and noting that the indicator functions concern disjoint events, we see that

$$\forall u \in [T_{\max}], \quad G_u = g_u \mathbb{1}_{\xi_u \notin S} + g_{h,u} \mathbb{1}_{\xi_u \in S} \,, \tag{C.7}$$

where we let $g_u := \exp\left(q^\top k_u / \sqrt{d_{\text{att}}}\right)$ and $g_{h,t} = \exp\left(q^\top k_{h,t} / \sqrt{d_{\text{att}}}\right)$ as in Eq. (6.18). Then, with this notation in hand, we define the random attention coefficient associated to token $t$ by

$$\forall t \in [T_{\max}], \quad A_t := \frac{G_t}{\sum_{u=1}^{T_{\max}} G_u} \,. \tag{C.8}$$

Finally, value vectors are also random in this setting. Namely,

$$\forall t \in [T_{\max}], \quad V_t := v_t \mathbb{1}_{\xi_t \notin S} + v_{h,t} \mathbb{1}_{\xi_t \in S} \,, \tag{C.9}$$

where we recall that $v_{h,t} = W_v(h + W_p(t))$.

**Reduction to key computation.** Looking at Eq. (6.17), and now with appropriate notation, we need to compute

$$\mathbb{E}\left[f(X) \mid \ell \notin S\right] = \mathbb{E}\left[\sum_{t=1}^{T_{\max}} A_t V_t \mid \ell \notin S\right]$$

for all $\ell \in [d]$. Again by linearity, one can focus on the computation of $\mathbb{E}\left[A_t V_t \mid \ell \notin S\right]$. The following result gives an approximation of this quantity when both $T_{\max}$ and $d$ are large:

**Proposition C.2.1** (Approximated conditional expectation)**.** *Assume that $d = T$. Assume further that there exist positive constants $0 < c < C$ such that, as $T \to +\infty$, for all $t \in [T_{\max}]$, $\max(|v_t|, |v_{h,t}|) \leq C$, and $c \leq \min(g_t, g_{h,t}) \leq C$. Then, for any $t \in [T_{\max}]$, if $\xi_t = \ell$,*

$$\mathbb{E}\left[A_t V_t \mid \ell \notin S\right] = \frac{1}{d} \sum_{s=1}^{d-1} \frac{g_t v_t}{\left(1 - \frac{s}{d-1}\right) \sum_u g_u + \frac{s}{d-1} \sum_u g_{h,u}} + \mathcal{O}\left(T_{\max}^{-3/2}\right) \,, \tag{C.10}$$

*and otherwise*

$$\mathbb{E}\left[A_t V_t \mid \ell \notin S\right] = \frac{1}{d} \sum_{s=1}^{d-1} \frac{\left(1 - \frac{s}{d-1}\right) g_t v_t + \frac{s}{d-1} g_{h,t} v_{h,t}}{\left(1 - \frac{s}{d-1}\right) \sum_u g_u + \frac{s}{d-1} \sum_u g_{h,u}} + \mathcal{O}\left(T_{\max}^{-3/2}\right) \,. \tag{C.11}$$

The proof of Proposition C.2.1 is deferred to Section C.3. From Eqs. (C.10) and (C.11), coming back to Eq. (6.17), we deduce that the $j$th limit coefficient associated to $A_t V_t$ is approximately equal to

$$\frac{3}{d}\sum_{s=1}^{d-1}\frac{\frac{s}{d-1}(g_t v_t - g_{h,t}v_{h,t})}{\left(1-\frac{s}{d-1}\right)\sum_u g_u + \frac{s}{d-1}\sum_u g_{h,u}} + \mathcal{O}\left(T_{\max}^{-3/2}\right). \tag{C.12}$$

The derivative of the mapping

$$x \mapsto \frac{x}{(1-x)\sum_u g_u + x\sum_u g_{h,u}}$$

is given by

$$x \mapsto \frac{\sum_u g_u}{(x(\sum_u g_{h,u} - \sum_u g_u) + \sum_u g_u)^2}.$$

On $[0,1]$, under our assumptions, the last display is uniformly bounded by $\mathcal{O}\left(T_{\max}^{-1}\right)$ in absolute value. Thus, by standard Riemann sum approximation, the last display is

$$3(g_t v_t - g_{h,t}v_{h,t})\int_0^1 \frac{x\mathrm{d}x}{(1-x)\sum_u g_u + x\sum_u g_{h,u}} + \mathcal{O}\left(T_{\max}^{-3/2}\right).$$

Let us recall that, if $T < u \leq T_{\max}$, $g_u = g_{h,u}$. Therefore, $\sum_u g_{h,u} - \sum_u g_u = \mathcal{O}(T) = \mathcal{O}(T_{\max}^{\varepsilon})$, and $(\sum_u g_{h,u} - \sum_u g_u)/\sum_u g_u = \mathcal{O}(T_{\max}^{\varepsilon-1})$. Therefore, according to Lemma C.4.2, the integral in the last display can be well approximated by

$$\frac{1}{\sum_u g_u}\cdot\left(\frac{1}{2} + \mathcal{O}\left(\frac{\sum_u g_{h,u} - \sum g_u}{\sum g_u}\right)\right) = \frac{1}{2\sum g_u} + \mathcal{O}\left(T_{\max}^{\varepsilon-2}\right).$$

The same reasoning shows that, whenever $\xi_t \neq j$, the approximation is zero (with the same precision in the error). Thus, by linearity (over the tokens and the model), we obtain the statement of Theorem 6.6.1. □

### C.2.1 Discussion on Theorem 6.6.1

A theoretical limitation of Theorem 6.6.1 it the assumption of distinct tokens. This assumption, while technically a simplification, enables a rigorous formalization of LIME's behavior using its default parameters as defined in the official implementation. We conducted quantitative experiments that disregarded this assumption, and the results still hold. We empirically validate the accuracy of Theorem 6.6.1 by computing the norm-2 error between the LIME weights from the official implementation (available on Github at `https://github.com/marcotcr/lime`) and our approximation. The average norm-2 error, computed over the full test set (see Section C.6), is $0.808$, with a standard deviation of $0.219$.

The primary challenge in proving a formal result that allows for repetitions lies in the use of Lemma C.4.1. The key intuition in the current proof mechanism is that if only one element of the denominator of $A_t$ varies randomly, this has a minimal overall effect on the entire denominator, given that it has $T \gg 1$ terms. However, if many tokens are identical, this is no longer true, and it can result in high variance (consider, for instance, the extreme case of two groups of identical tokens), which prevents us from using Lemma C.4.1. Nevertheless, we conjecture that if the tokens are not distinct, but the maximal multiplicity of tokens is small relative to $T$, our findings hold true (and this is empirically true).

On a more practical note, we highlight that by default, LIME-text perturbs input data by removing all occurrences of individual words or characters (`bow=True`, *i.e.*, bag-of-words), and this is the subject of our study. However, if the underlying model uses word location (as in our classifier), a possibility is to set `bow=False` (as recommended in their notebook), so that any occurrence of the same word is considered a distinct token. By using this option, the same applies in Theorem 6.6.1: the same words in different parts of the text are considered different tokens.

## C.3 Proof of Proposition C.2.1

**Sketch of the proof.** Essentially, assuming for a second that $V_t$ is constant, the crux of the result is to compute the (approximate) expectation of $A_t$, which is defined as the ratio of positive quantities (Eq. (C.8)). Since the denominator is quite large, one can use Lemma C.4.1 and approximate the expected ratio by the ratio of expectation. This works only if, concurrently, the variance is not too high, which is guaranteed by Lemma C.4.3. Lemmas C.4.1 and C.4.3 are stated and proved in Section C.4.

**Proof of Proposition C.2.1.** We first write

$$\begin{aligned}
\mathbb{E}\left[A_t V_t \mid \ell \notin S\right] &= \mathbb{E}\left[\frac{G_t V_t}{\sum_{u=1}^{T_{\max}} G_u} \mid \ell \notin S\right] && \text{(Eqs. (C.8) and (C.9))} \\
&= \mathbb{E}\left[\frac{g_t v_t \mathbb{1}_{\xi_t \notin S} + g_{h,t} v_{h,t} \mathbb{1}_{\xi_t \in S}}{\sum_{u=1}^{T_{\max}} \left\{g_u \mathbb{1}_{\xi_u \notin S} + g_{h,u} \mathbb{1}_{\xi_u \in S}\right\}} \mid \ell \notin S\right] && \text{(Eq. (C.7))} \\
&= \frac{1}{d}\sum_{s=0}^{d-1} \mathbb{E}_s\left[\frac{g_t v_t \mathbb{1}_{\xi_t \notin S} + g_{h,t} v_{h,t} \mathbb{1}_{\xi_t \in S}}{\sum_{u=1}^{T_{\max}} \left\{g_u \mathbb{1}_{\xi_u \notin S} + g_{h,u} \mathbb{1}_{\xi_u \in S}\right\}} \mid \ell \notin S\right]. \\
& && \text{(law of total expectation)}
\end{aligned}$$

Note that there is no $s = d$ term in the last display, since $d$ removals is incompatible with $\ell \notin S$. Let us set $s \in [d-1]$. Define $X := g_t v_t \mathbb{1}_{\xi_t \notin S} + g_{h,t} v_{h,t} \mathbb{1}_{\xi_t \in S}$ and $Y := \sum_{u=1}^{T_{\max}} \left\{g_u \mathbb{1}_{\xi_u \notin S} + g_{h,u} \mathbb{1}_{\xi_u \in S}\right\}$. Under our assumptions, $X$ is clearly bounded while $Y$ has order $T_{\max}$. Thus the hypotheses of Lemma C.4.1 are satisfied with $n = T_{\max}$. Moreover, Lemma C.4.3 guarantees that $\mathrm{Var}_s(Y \mid \ell \notin S) = \mathcal{O}(T_{\max})$. From Lemma C.4.1, we deduce that

$$\mathbb{E}_s\left[\frac{g_t v_t \mathbb{1}_{\xi_t \notin S} + g_{h,t} v_{h,t} \mathbb{1}_{\xi_t \in S}}{\sum_{u=1}^{T_{\max}} \left\{g_u \mathbb{1}_{\xi_u \notin S} + g_{h,u} \mathbb{1}_{\xi_u \in S}\right\}} \mid \ell \notin S\right] = \frac{\mathbb{E}_s\left[g_t v_t \mathbb{1}_{\xi_t \notin S} + g_{h,t} v_{h,t} \mathbb{1}_{\xi_t \in S} \mid \ell \notin S\right]}{\mathbb{E}_s\left[\sum_{u=1}^{T_{\max}} \left\{g_u \mathbb{1}_{\xi_u \notin S} + g_{h,u} \mathbb{1}_{\xi_u \in S}\right\} \mid \ell \notin S\right]} + \mathcal{O}\left(T_{\max}^{-3/2}\right). \tag{C.13}$$

Let us assume from now on that $\xi_t = \ell$ (the case $\xi_t \neq \ell$ is similar). Then

$$\mathbb{E}_s\left[g_t v_t \mathbb{1}_{\xi_t \notin S} + g_{h,t} v_{h,t} \mathbb{1}_{\xi_t \in S} \mid \ell \notin S\right] = g_t v_t\,, \tag{C.14}$$

and for all $u \neq t$,

$$\mathbb{E}_s\left[g_u \mathbb{1}_{\xi_u \notin S} + g_{h,u} \mathbb{1}_{\xi_u \in S} \mid \ell \notin S\right] = g_u\, \mathbb{P}_s\left(\xi_u \notin S \mid \ell \notin S\right) + g_{h,u}\, \mathbb{P}_s\left(\xi_u \in S \mid \ell \notin S\right).$$

Since we assumed distinct tokens ($d = T$), Lemma C.4.5 yields

$$\mathbb{E}_s\left[g_u \mathbb{1}_{\xi_u \notin S} + g_{h,u} \mathbb{1}_{\xi_u \in S} \mid \ell \notin S\right] = \left(1 - \frac{s}{d-1}\right) g_u + \frac{s}{d-1} g_{h,u}\,. \tag{C.15}$$

Injecting Eqs. (C.14) and (C.15) into Eq. (C.13), we obtain

$$\mathbb{E}_s\left[\frac{g_t v_t \mathbb{1}_{\xi_t \notin S} + g_{h,t} v_{h,t} \mathbb{1}_{\xi_t \in S}}{\sum_{u=1}^{T_{\max}} \left\{g_u \mathbb{1}_{\xi_u \notin S} + g_{h,u} \mathbb{1}_{\xi_u \in S}\right\}} \mid \ell \notin S\right] \tag{C.16}$$

$$= \frac{g_t v_t}{\left(1 - \frac{s}{d-1}\right) \sum_u g_u + \frac{s}{d-1} \sum_u g_{h,u} + \frac{s}{d-1}(g_t - g_{h,t})} \tag{C.17}$$

$$+ \mathcal{O}\left(T_{\max}^{-3/2}\right). \tag{C.18}$$

Under our assumptions, $\frac{s}{d-1}(g_t - g_{h,t})$ is $\mathcal{O}(1)$, whereas the remainder of the denominator is of order at least $T_{\max}$. We deduce that

$$\mathbb{E}_s\left[\frac{g_t v_t \mathbb{1}_{\xi_t \notin S} + g_{h,t} v_{h,t} \mathbb{1}_{\xi_t \in S}}{\sum_{u=1}^{T_{\max}} \left\{g_u \mathbb{1}_{\xi_u \notin S} + g_{h,u} \mathbb{1}_{\xi_u \in S}\right\}} \mid \ell \notin S\right] = \frac{g_t v_t}{\left(1 - \frac{s}{d-1}\right) \sum_u g_u + \frac{s}{d-1} \sum_u g_{h,u}} \tag{C.19}$$

$$+ \mathcal{O}\left(T_{\max}^{-3/2}\right). \tag{C.20}$$

We deduce the result coming back to the initial decomposition. □

## C.4 Technical results

**Lemma C.4.1 (Expected ratio).** *Let $X$ and $Y$ be two random variables with finite variance. Assume that there exist two positive constants $c$ and $C$ such that $|X| \leq C$ and $cn \leq Y \leq Cn$ a.s. Then*

$$\left|\mathbb{E}\left[\frac{X}{Y}\right] - \frac{\mathbb{E}[X]}{\mathbb{E}[Y]}\right| \leq \frac{C\mathrm{Var}(Y)}{c^3 n^3} + \frac{C^2\sqrt{\mathrm{Var}(Y)}}{c^2 n^2}.$$

*Proof.*

Set $\psi : \mathbb{R}_+^2 \to \mathbb{R}$ defined as $\psi(x, y) := x/y$. Multivariate Taylor expansion at order 1 with integral remainder for an arbitrary $(x_0, y_0) \in \mathbb{R}_+^2$ yields

$$\begin{aligned}
\psi(x,y) &= \psi(x_0,y_0) + (x-x_0)\partial_x\psi(x_0,y_0) + (y-y_0)\partial_y\psi(x_0,y_0) \\
&+ \frac{2}{2!}(x-x_0)^2 \int_0^1 (1-t)\partial_{xx}\psi((x_0,y_0) + t(x-x_0, y-y_0))\mathrm{d}t \\
&+ \frac{2}{1!1!}(x-x_0)(y-y_0) \int_0^1 (1-t)\partial_{xy}\psi((x_0,y_0) + t(x-x_0, y-y_0))\mathrm{d}t \\
&+ \frac{2}{2!}(y-y_0)^2 \int_0^1 (1-t)\partial_{yy}\psi((x_0,y_0) + t(x-x_0, y-y_0))\mathrm{d}t\,.
\end{aligned}$$

Let us focus on the remainder (the last three lines of the previous display). Since $\partial_{xx}\psi = 0$, $\partial_{xy}\psi = -1/y^2$, and $\partial_{yy}\psi = 2x/y^3$, we are left with

$$-2(x-x_0)(y-y_0)\int_0^1 \frac{(1-t)\mathrm{d}t}{(y_0 + t(y-y_0))^2} + 2(y-y_0)^2 \int_0^1 \frac{(1-t)(x_0 + t(x-x_0))\mathrm{d}t}{(y_0 + t(y-y_0))^3},$$

that is,

$$(y-y_0)\cdot \int_0^1 (1-t)\frac{-2(x-x_0)(y_0 + t(y-y_0)) + 2(y-y_0)(x_0 + t(x-x_0))}{(y_0 + t(y-y_0))^3}\mathrm{d}t\,.$$

One can actually compute this integral, which is

$$(y-y_0)\cdot\frac{x_0y-xy_0}{yy_0^2}=(y-y_0)\cdot\frac{x_0(y-y_0)-(x-x_0)y_0}{yy_0^2}\,.$$

Going back to the original expansion, we have proved that

$$\psi(x,y)=\psi(x_0,y_0)+(x-x_0)\partial_x\psi(x_0,y_0)+(y-y_0)\partial_y\psi(x_0,y_0) \tag{C.21}$$

$$+(y-y_0)\cdot\frac{x_0(y-y_0)-(x-x_0)y_0}{yy_0^2}\,. \tag{C.22}$$

Let us now use Eq. (C.21) with $x=X$, $y=Y$, $x_0=\mathbb{E}[X]$, and $y_0=\mathbb{E}[Y]$, and then take expectation on both sides. We see that the linear term vanishes, and we are left

$$\mathbb{E}\left[\frac{X}{Y}\right]-\frac{\mathbb{E}[X]}{\mathbb{E}[Y]}=\mathbb{E}\left[(Y-\mathbb{E}[Y])\cdot\frac{\mathbb{E}[X](Y-\mathbb{E}[Y])-(X-\mathbb{E}[X])\mathbb{E}[Y]}{Y\mathbb{E}[Y]^2}\right]\,.$$

In absolute value, the last display is smaller than

$$\frac{C\mathrm{Var}(Y)}{c^3n^3}+\frac{C\sqrt{\mathrm{Var}(X)\,\mathrm{Var}(Y)}}{c^2n^2}\,,$$

and we deduce the result using Popoviciu's inequality to bound the variance of $X$.

□

We conclude with a technical result used in approximating an integral appearing in the proof of Theorem 6.6.1.

**Lemma C.4.2 (Integral approximation).** *Let $a>0$. Then*

$$\int_0^1\frac{x\mathrm{d}x}{1+ax}=\frac{a-\log(a+1)}{a^2}=\frac{1}{2}-\frac{a}{3}+\mathcal{O}\left(a^2\right)\,.$$

*Proof.*

Taylor expansion.

□

### C.4.1 Conditional variance computations

To use Lemma C.4.1 in a meaningful way, the variance of the denominator needs to be controlled. We show that this is the case with this next result.

**Lemma C.4.3 (Conditional variance computation).** *Let $a_i$ and $b_i$ be two sequences of positive numbers for $i\in[n]$. We set $H_S$ as before. Let $\ell\in[n]$. Then, for all $s\in[n-1]$,*

$$\mathbb{E}_s\left[H_S\mid\ell\notin S\right]=\frac{n-1-s}{n-1}\sum_i a_i+\frac{s}{n-1}\sum_i b_i+\frac{s}{n-1}(a_\ell-b_\ell)\,,$$

*and*

$$\mathrm{Var}_s(H_S\mid\ell\notin S)=\frac{ns(n-s-1)}{(n-1)(n-2)}\left[\widehat{\mathrm{Var}}\,(a-b)-\frac{1}{n-1}\left(a_\ell-b_\ell-(\overline{a}-\overline{b})\right)^2\right]\,,$$

*where $\overline{a}$ (resp. $\overline{b}$) denote the empirical mean of $a$ (resp. $b$).*

Lemma C.4.3 is somewhat remarkable, connecting the variance of the random sum underpinning our problems to the empirical variance of the coefficients. In particular, if we assume that the variance of the summands is $\mathcal{O}(1)$, then $\mathrm{Var}_s(H_S \mid \ell \notin S) = \mathcal{O}(n)$.

*Proof.*

We first write

$$\begin{aligned}\mathbb{E}_s\left[H_S \mid \ell \notin S\right] &= \mathbb{E}_s\left[\sum_i \{a_i \mathbb{1}_{i\notin S} + b_i \mathbb{1}_{i\in S}\} \mid \ell \notin S\right] \\ &= \sum_i a_i \,\mathbb{P}_s\left(i \notin S \mid \ell \notin S\right) + \sum_i b_i \,\mathbb{P}_s\left(i \in S \mid \ell \notin S\right) .\end{aligned}$$

Taking special care of the case $i = \ell$ in the previous display and using Lemma C.4.5, we obtain

$$\mathbb{E}_s\left[H_S \mid \ell \notin S\right] = \frac{n-1-s}{n-1}\sum_{i\neq\ell} a_i + \frac{s}{n-1}\sum_{i\neq\ell} b_i + a_\ell \,.$$

Rearranging this expression yields the fist statement of the lemma. We now turn to the variance computation. Without loss of generality, we can assume that $b = 0$ since

$$H_S = \sum_i \{a_i \mathbb{1}_{i\notin S} + b_i \mathbb{1}_{i\in S}\} = \sum_i (a_i - b_i)\mathbb{1}_{i\notin S} + \sum_i b_i \,.$$

Moreover, we notice that

$$\sum_i (a_i + \lambda)\mathbb{1}_{i\notin S} = \sum_i a_i \mathbb{1}_{i\notin S} + \lambda \sum_i \mathbb{1}_{i\notin S} = \sum_i a_i \mathbb{1}_{i\notin S} + (n-s)\lambda \,.$$

Thus, without loss of generality, we can assume that $\sum_i a_i = 0$. Under this assumption, the expectation is simply

$$\mathbb{E}_s\left[H_S \mid \ell \notin S\right] = \frac{s}{n-1} a_\ell \,.$$

We then compute the second raw moment:

$$\begin{aligned}\mathbb{E}_s\left[H_S^2 \mid \ell \notin S\right] &= \mathbb{E}_s\left[\left(\sum_i a_i \mathbb{1}_{i\notin S}\right)^2 \mid \ell \notin S\right] \\ &= \sum_i a_i^2 \,\mathbb{P}_s\left(i \notin S \mid \ell \notin S\right) + 2\sum_{i<j} a_i a_j \,\mathbb{P}_s\left(i \notin S, j \notin S \mid \ell \notin S\right) .\end{aligned}$$

As before, we have to take care of equality cases. Using Lemma C.4.5 and our assumption that $\sum_i a_i = 0$, we find

$$\begin{aligned}\mathbb{E}_s\left[H_S^2 \mid \ell \notin S\right] &= \left(\sum_{i\neq\ell} a_i^2\right)\frac{n-s-1}{n-1} + a_\ell^2 + \left(2\sum_{\substack{i<j \\ i,j\neq\ell}} a_i a_j\right)\frac{(n-s-1)(n-s-2)}{(n-1)(n-2)} \\ &\quad + \left(2a_\ell \sum_{i\neq\ell} a_i\right)\frac{n-1-s}{n-1} \\ &= \left(\sum_i a_i^2\right)\frac{n-s-1}{n-1} + \left(2\sum_{i<j} a_i a_j\right)\frac{(n-s-1)(n-s-2)}{(n-1)(n-2)} - a_\ell^2 \frac{s(n-2s)}{(n-1)(n-2)} \\ &= \frac{s(n-s-1)}{n-1)(n-2)}\sum_i a_i^2 - \frac{s(n-2s)}{(n-1)(n-2)} a_\ell^2 \,,\end{aligned}$$

since $2\sum_{i<j} a_i a_j = -\sum_i a_i^2$. Putting everything together, we obtain

$$\mathrm{Var}_s(H_S \mid \ell \notin S) = \frac{ns(n-s-1)}{(n-1)(n-2)}\left[\widehat{\mathrm{Var}}\,(a) - \frac{1}{n-1}a_\ell^2\right],$$

from which we deduce the result.

□

### C.4.2 Probability computations

**Lemma C.4.4 (Exact expressions).** *Let $a, b, c$ be distinct elements of $[n]$. Then*

$$\begin{cases} \mathbb{P}_s\left(a \notin S\right) = \frac{n-s}{n} \\ \mathbb{P}_s\left(a \notin S, b \notin S\right) = \frac{(n-s)(n-1-s)}{n(n-1)} \\ \mathbb{P}_s\left(a \notin S, b \in S\right) = \frac{s(n-s)}{n(n-1)} \\ \mathbb{P}_s\left(a \notin S, b \notin S, c \notin S\right) = \frac{(n-s)(n-s-1)(n-s-2)}{n(n-1)(n-2)} \\ \mathbb{P}_s\left(a \notin S, b \in S, c \notin S\right) = \frac{s(n-s-1)(n-s)}{n(n-1)(n-2)} \end{cases} \tag{C.23}$$

*Proof.*

Similar to Lemma 4 in Mardaoui and Garreau [2021].

□

**Lemma C.4.5 (Exact expressions, conditional).** *Let $a, b, \ell$ be distinct elements of $[n]$. Then*

$$\begin{cases} \mathbb{P}_s\left(a \notin S \mid \ell \notin S\right) = \frac{n-1-s}{n-1} \\ \mathbb{P}_s\left(a \in S \mid \ell \notin S\right) = \frac{s}{n-1} \\ \mathbb{P}_s\left(a \notin S, b \notin S \mid \ell \notin S\right) = \frac{(n-s-1)(n-s-2)}{(n-1)(n-2)} \end{cases} \tag{C.24}$$

*Proof.*

Let us prove the first statement, the other ones are similar. By Bayes formula,

$$\mathbb{P}_s\left(a \notin S \mid \ell \notin S\right) = \frac{\mathbb{P}_s\left(a \notin S, \ell \notin S\right)}{\mathbb{P}_s\left(\ell \notin S\right)}.$$

We now simply use Lemma C.4.4.

□

## C.5 Experiments on multi-layer architecture

We have conducted numerical experiments on a multi-head, multi-layer architecture. We trained a classifier with 6 layers and 6 heads on the IMDb dataset (refer to Appendix C.6), achieving an accuracy of $82.22\%$. Our interest lies in exploring the relationship between LIME and the attention weights. We measured the correlation between LIME coefficients and the $\alpha$-avg (refer to

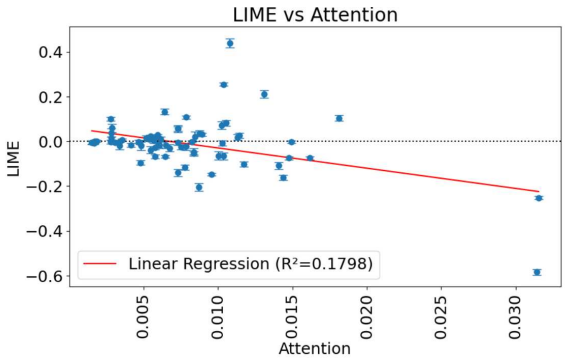


Figure C.2 – **Relation between LIME explanations and attention weights** for a 6-layer 6-head attention-based classifier. Attention weights correspond to the average attention over the 6 heads of the first layer. Error bars represent the standard deviation of LIME weights on 5 repetition.

Eq. (6.19)) for the first layer. An illustration is available at Figure C.2, where the document corresponds to Figure 6.5 of Chapter 6. The Pearson's correlation in this case is $\rho = -0.424$. We attribute the negative sign to the document being classified as negative (as in Figure 6.5). Attention weights consistently fall within the range of $[0, 1]$. Considering the absolute values, the rankings of the two explanations are closely aligned, and the correlation is $\rho = 0.672$. Although we cannot explicitly state the dependency for multi-layers as in Eq. (6.19), our experiments suggest a significant relationship. We conclude that the attention weights are interconnected with LIME coefficients, which adapt more effectively to the model. We are currently conducting broader experiments and will incorporate their results and subsequent discussions into the manuscript.

## C.6 Experiments

In this section, we report technical details for the model and the experiments. Any of the experiments presented in Chapter 6 have been performed on a `PyTorch` implementation of the model presented in Section 6.3 and ran on one GPU Nvidia A100.

**Code.** The full code is available at `https://github.com/gianluigilopardo/attention_meets_xai`.

**Model.** The model parameters were set as follows: $T_{\max} = 256$, $d_e = 128$, $d_{\text{att}} = 64$, $d_{\text{out}} = 64$.

**Dataset.** We trained the model on the `IMDB` dataset [Maas et al., 2011], which was preprocessed using standard tokenization and padding techniques. The dataset was split into training, validation, and test sets with sizes of $20,000$, $5,000$, and $25,000$ samples, respectively.

**Training.** The model was trained for 10 epochs using a batch size of 16. We employed the `AdamW` optimizer with a learning rate of $0.0001$ and used cross-entropy loss as the optimization objective.

# Fondements de l'Interprétabilité de l'Apprentissage Automatique

Gianluigi LOPARDO

**Résumé**

L'utilisation croissante de modèles complexes d'apprentissage automatique (ML), en particulier dans des applications critiques, a souligné le besoin urgent de méthodes d'interprétabilité. Malgré la variété de solutions proposées pour expliquer les décisions algorithmiques automatisées, comprendre leur processus de prise de décision reste un défi. Ce manuscrit examine l'interprétabilité des modèles ML, utilisant une analyse mathématique et une évaluation empirique pour comparer les méthodes existantes et proposer de nouvelles solutions. Notre principal objectif est sur les méthodes d'interprétabilité post-hoc, qui fournissent des informations sur le processus de prise de décision des modèles de ML après l'entraînement, indépendamment des architectures de modèles spécifiques. Nous nous interessons plus particuliermenet du langage naturel, explorant des techniques pour expliquer les modèles de texte. Nous abordons un défi clé : les méthodes d'interprétabilité peuvent produire des explications variées même pour des modèles apparemment simples. Cela met en évidence un problème critique : l'absence d'une base théorique solide pour ces méthodes. Pour tenter de resoudre ce problème , nous utilisons un cadre théorique rigoureux pour analyser formellement les techniques d'interprétabilité existantes, évaluant leur comportement et leurs limites. Sur cette base, nous proposons un nouvel explicateur pour fournir une approche plus fidèle et robuste pour interpréter les modèles de données textuelles. Nous nous engageons également dans le débat sur l'efficacité des poids d'attention comme outils explicatifs au sein des architectures de transformateurs puissants. Grâce à cette analyse, nous éclairons les forces et les limites des méthodes d'interprétabilité existantes et ouvrons la voie à des approches plus fiables et théoriquement fondées. Cela conduira à une compréhension plus profonde de la façon dont les modèles prennent des décisions, favorisant la confiance et le déploiement responsable dans les applications ML critiques.

**Mots-clés :** Interprétabilité de l'apprentissage automatique, IA Explicable, Traitement du langage

**Abstract**

The rising use of complex Machine Learning (ML) models, especially in critical applications, has highlighted the urgent need for interpretability methods. Despite the variety of solutions proposed to explain automated algorithmic decisions, understanding their decision-making process remains a challenge. This manuscript investigates the interpretability of ML models, using mathematical analysis and empirical evaluation to compare existing methods and propose novel solutions. Our main focus is on post-hoc interpretability methods, which provide insights into the decision-making process of ML models post-training, independent of specific model architectures. We delve into Natural Language Processing (NLP), exploring techniques for explaining text models. We address a key challenge: interpretability methods can yield varied explanations even for simple models. This highlights a critical issue: the absence of a robust theoretical foundation for these methods. To address this issue, we use a rigorous theoretical framework to formally analyze existing interpretability techniques, assessing their behavior and limitations. Building on this, we propose a novel explainer to provide a more faithful and robust approach to interpreting text data models. We also engage with the debate on the effectiveness of attention weights as explanatory tools within powerful transformer architectures. Through this analysis, we expose the strengths and limitations of existing interpretability methods and pave the way for more reliable, theoretically grounded approaches. This will lead to a deeper understanding of how complex models make decisions, fostering trust and responsible deployment in critical ML applications.

**Keywords:** Machine Learning Interpretability, Explainable AI, Natural Language Processing.