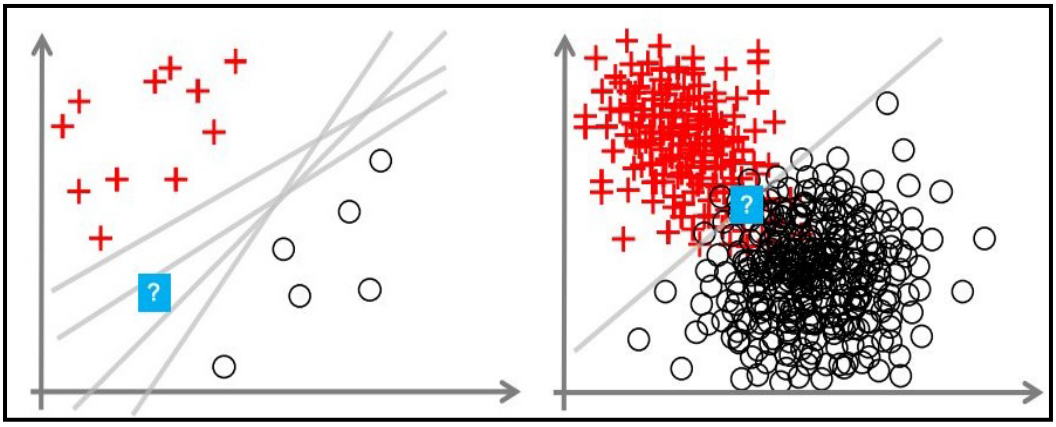
Conformal prediction methods for complex data : application to real estate management

Soundouss Messoudi

Par **Soundouss MESSOUDI**

Conformal prediction methods for complex data: application to real estate management

Thèse présentée
pour l'obtention du grade
de Docteur de l'UTC



Soutenue le 9 décembre 2022
Spécialité : Informatique : Unité de recherche Heudyasic (UMR-7253)
D2716

University of Technology of Compiègne

Doctoral Thesis

Conformal prediction methods for complex data: application to real estate management

Spécialité : Informatique

Soundouss MESSOUDI

Supervisors:
Dr. Sébastien DESTERCKE & Assoc. Prof. Sylvain ROUSSEAU

December 09, 2022

Jury:

Prof.	Henrik BOSTRÖM	KTH Royal Institute of Technology	Reviewer
Prof.	Ines COUSO	University of Oviedo	Reviewer
Dr.	Yves GRANDVALET	University of Technology of Compiègne	Examiner
Assoc. Prof.	Marie-Jeanne LESOT	Sorbonne University	Examiner
Assoc. Prof.	Mohamed HEBIRI	Gustave Eiffel University	Examiner

A thesis submitted in fulfillment of the requirements for the degree of Doctor in the

CID Team (Knowledge, Uncertainty, Data)
Heudiasyc Laboratory

UNIVERSITY OF TECHNOLOGY OF COMPIÈGNE

Abstract

CID Team (Knowledge, Uncertainty, Data)
Heudiasyc Laboratory

Doctor of Philosophy

Conformal prediction methods for complex data: Application to real estate management

by Soundouss Messoudi

Uncertainty quantification is not an easy task. Its difficulty depends on various factors related to the available data, the application domain, and also the learned task. Having multiple outputs to predict simultaneously can be even more demanding, principally when these outputs are correlated.

This research work focuses on producing confidence regions for such complex problems, by using conformal prediction: a theoretically proven method that can be added to any Machine Learning model to generate set predictions whose size and statistical guarantee depend on a user-defined error rate.

Our first and main problem of interest consists of multi-target regression, where the objective is to predict many real-valued outputs at once. First, a simple extension of single-target regression conformal methods is proposed by following a naive approach that treats these targets as independent. Second, copulas are exploited to take into account the existing correlations between outputs when giving conformal regions. Third, ellipsoids are considered in order to produce more flexible conformal regions according to the possible relationships between targets while maintaining the desired error rate.

Our second problem of interest is an applied research work that deals with debt prediction for tenants of rented social housing to control the errors of a particular class in a imbalanced binary classification context. In this case, mondrian conformal prediction, which is a variant of conformal inference, is used to treat this problem with the guidance of real estate experts.

Keywords: conformal prediction, multi-target regression, uncertainty, debt prediction.

Contents

List of Notations

Symbols	Description
Data related notations	
$\mathbf{X}$	Object space
x	Object belonging to $\mathbf{X}$
$\mathbf{Y}$	Target space
y	Target belonging to $\mathbf{Y}$
$\hat{y}$	Precise output prediction
$\mathbf{Z}$	Example space with $\mathbf{Z} := \mathbf{X} \times \mathbf{Y}$
z	Example belonging to $\mathbf{Z}$
H	Simple predictor
C_k	Class of labels
Probability and statistics	
$P(A)$	Unconditional probability of event A
$P(A \mid B)$	Conditional probability of event A knowing event B
$\mathbb{P}$	Probability measure
$\mathscr{P}$	Set of probabilities
θ	Unknown parameters
$\mathbf{D}$	Distribution over $\mathbf{X} \times \mathbf{Z}$
$\hat{p}(A)$	Probabilistic prediction of event A
$p(A)$	True probability of event A
U_i	Uniform distributed function
F_i	Cumulative distribution function
$F^{\leftarrow}$	Generalized function of c.d.f. F_i
C	Copula
ϕ	Copula generator
$\hat{\Sigma}$	Sample covariance matrix
$\hat{Cov}_i$	Local covariance matrix of x_i
Multi-target regression	
y	Multivariate target
y^j	Individual target
m	Number of dimensions

Conformal Prediction

$\{z_1, \ldots, \ldots, z_n\}$	Bag of examples
$\mathbf{Z}^{tr}$	Proper training set
$\mathbf{Z}^{cal}$	Calibration set
$\mathbf{Z}^{ts}$	Test set
h	Underlying algorithm
ϵ	Significance level
Γ	Confidence predictor
A_n	Non-conformity measure
π_i	P-value of z_i
α_i	Non-conformity score of z_i
σ_i	Difficulty estimator of z_i
$\Gamma^\epsilon(x_i)$	Conformal prediction set of x_i with a chosen ϵ for single-output regression
$[\Gamma^\epsilon(x_i)]$	Conformal prediction set of x_i with a chosen ϵ for multi-target regression

Mathematic and other notations

$(\cdot)^T$	Transpose operator
$(\cdot)^{-1}$	Inverse operator
$\|\cdot\|$	Euclidian norm
E	Ellipsoid
B_m	Unit ball

To my parents
To my husband

Chapter 1

Introduction

> *Sometimes it is the people no one can imagine anything of who do the things no one can imagine.*
>
> **— Alan Turing**

Contents

Scientific advances in Artificial Intelligence have been highly useful in many domains during the last few decades, with new emerging methods being employed more and more in everyday life applications [Sarker, 2021]. However, exclusively relying on Machine Learning and Deep Learning model predictions for decision-making can sometimes come with irreparable consequences, especially when the domain application implicates human safety, such as health-care or autonomous driving. In these cases, it is of the utmost importance to introduce uncertainty estimation to the model predictions so that experts can decide to which degree they can trust these outcomes.

To solve these issues, conformal prediction was proposed by Vladimir Vovk, Alexander Gammerman and Vladimir Vapnik [Gammerman et al., 1998] in the late nineties as a simple, robust and flexible framework that can provide confidence regions while respecting the desired error rate.

Although being relatively recent, this theoretically-proven framework has gained popularity over the last few years. Indeed, it has its own Symposium "COPA" (Conformal and Probabilistic Prediction and its Applications), which is held each year since 2012, and a dedicated GitHub repository [Manokhin, 2022] created in late 2019, and with over 700 stars at the time this thesis is written. Also, it has more and more papers being published each year, as it can be seen in Figure 1.1.

This thesis contributes to the research field around this method by applying conformal prediction to complex data: a first methodological work focused on multi-target regression, and a second applied work on a

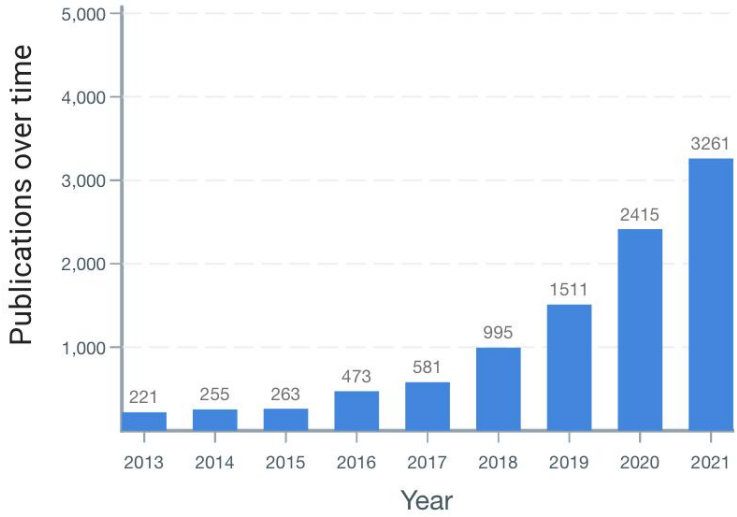


Figure 1.1: Conformal prediction paper publication growth.

real-estate problem encountered by domain experts from Sopra/Steria, a major French company that is interested in our research studies.

1.1 THESIS OUTLINE

This thesis manuscript is organized as follows:

- CHAPTER 2 - LITERATURE REVIEW

  This chapter presents a small reminder on Machine Learning, in particular supervised learning, and a quick overview on uncertainty and its sources, before introducing some methods to estimate it in predictions.

- CHAPTER 3 - CONFORMAL PREDICTION

  This chapter details conformal inference methods in both classification and regression cases. An application example is also given as a walk-through to understand better conformal prediction and its principles.

- CHAPTER 4 - CONFORMAL PREDICTION FOR MULTI-TARGET REGRESSION

  This chapter presents our contributions to the conformal inference research field by extending it to the multi-target regression problem, with three main methods (naive, copula, and ellipsoid).

- CHAPTER 5 - CONFORMAL PREDICTION APPLIED TO REAL ESTATE MANAGEMENT

This chapter presents a real-world application of conformal prediction to solve a tenant's debt classification problem, by using Mondrian conformal prediction and following the requirements of real estate experts.

1.2 MAIN CONTRIBUTIONS

During the last three years, we published a collection of scientific papers in peer-reviewed conferences and one journal. These contributions are presented throughout this thesis manuscript in the following chapters:

- CHAPTER 3 - CONFORMAL PREDICTION
  - Messoudi et al. (2020). "Deep Conformal Prediction for Robust Models". In: International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems.

  This paper consists of a first exploratory work on conformal prediction based on a density estimation method, and applied to three kinds of data. It is used in Chapter 3 as an illustrative example of conformal prediction and how it can make DL models more robust to noise and outliers.

- CHAPTER 4 - CONFORMAL PREDICTION FOR MULTI-TARGET REGRESSION
  - Messoudi et al. (2020). "Conformal multi-target regression using neural networks". In: Conformal and Probabilistic Prediction and Applications.
  - Messoudi et al. (2021). "Copula-based conformal prediction for multi-target regression". In: Pattern Recognition.
  - Messoudi et al. (2022). "Ellipsoidal conformal inference for Multi-Target Regression". In: Conformal and Probabilistic Prediction with Applications.

  In the context of multi-target regression, the first paper proposes a naive approach of conformal prediction applied to multi-target regression by considering that the targets are independent. The second one extends this work by using copulas to exploit dependencies between targets when constructing hyper-rectangle conformal regions. The third one suggests another approach that produces ellipsoidal conformal regions, a more flexible form that adds in efficiency while maintaining validity. These papers constitute the bulk of our contributions in Chapter 4.

- CHAPTER 5 - CONFORMAL PREDICTION APPLIED TO REAL ESTATE MANAGEMENT

  - Messoudi et al. (2021). "Class-wise confidence for debt prediction in real estate management: discussion and lessons learned from an application". In: Conformal and Probabilistic Prediction and Applications.

  This applied research paper presented in Chapter 5 uses a variant of conformal prediction to control the errors of a minority class in a real-world problem, which is the binary classification of tenant's debt. This work was conducted by following suggestions of real estate experts.

In addition to these publications, a conference paper was published as a side collaboration on the label ranking problem. This work uses imprecise probabilities to propose an efficient way to make partial predictions with constraint satisfaction.

- Alarcón et al. (2020). "Cautious label-wise ranking with constraint satisfaction". In: International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems.

Finally, and for a scientific diffusion in France and other French-speaking countries, some of the published papers were presented in French national conferences:

- Alarcón et al. (2020). "Apprentissage de rangements prudent avec satisfaction de contraintes". In: 29èmes Rencontres francophones sur la Logique Floue et ses Applications.

- Messoudi et al. (2020). "Prédiction conformelle profonde pour des modèles robustes". In: Extraction et Gestion des Connaissances (EGC 2020).

- Messoudi et al. (2021). "Confiance de classe pour la prédiction de dette en gestion immobilière". In: 30èmes Rencontres francophones sur la Logique Floue et ses Applications.

- Messoudi et al. (2022). "Prédiction conformelle basée sur les copules pour la régression multi-cibles". In: Extraction et Gestion des Connaissances (EGC 2022).

Chapter 2

Literature review

> *As far as the laws of mathematics refer to reality, they are not certain; and as far as they are certain, they do not refer to reality.*
>
> **— Albert Einstein**

Contents

Machine Learning (ML) is a branch of Artificial Intelligence (AI) that focuses on building mathematical models based on data to make decisions on their own, without being explicitly programmed to predict a specific outcome. One of these models' main objective is to be able to generalize from data samples presented to them so as to give accurate and precise predictions on never-before-seen examples.

There exists many types of ML algorithms depending on the tasks that need to be learned. In this thesis, we will focus on *predictive uncertainty*, i.e. estimating the uncertainty of each prediction given its corresponding input example. Hence, in this chapter, we will only fixate on supervised learning problems and predictive uncertainty, by presenting its types, sources, and methods to treat it.

2.1 REMINDER ON SUPERVISED LEARNING

The supervised learning setting consists of training a model on input data that has been labeled by human experts to generate a corresponding output data. Thus, a supervised learning model aims at finding the hidden patterns between the inputs and the outputs.

To give a clearer mathematical explanation to supervised learning, let us consider an *object* x belonging to an *object space* $\mathbf{X}$ and its *target* y belonging to a *target space* $\mathbf{Y}$. A *predictor* H can be defined as a map:

$$H : \mathbf{X} \to \mathbf{Y}.$$

In classification, the target y is a label from $\mathbf{Y} = \{C_1, \ldots, C_k\}, k \in \mathbb{N}$. In regression, y is a real value in $\mathbf{Y} \subset \mathbb{R}$. Note that, in some cases, we will use the notation $z = (x, y)$ to talk about *examples*, *samples* or *instances* [1], with $\mathbf{Z}$ being the *example space* defined as $\mathbf{Z} := \mathbf{X} \times \mathbf{Y}$.

Obtaining a prediction with the predictor H can be done in two ways as illustrated in Figure 2.1 [Vovk et al., 2005]: using *transduction*, which points out at the inference setting where the logic behind predictions on test objects comes from observing available training examples, or *induction*, i.e. the predictor H derives a general rule from observing training examples in the inductive step, that is then applied to the test objects in the deductive step.

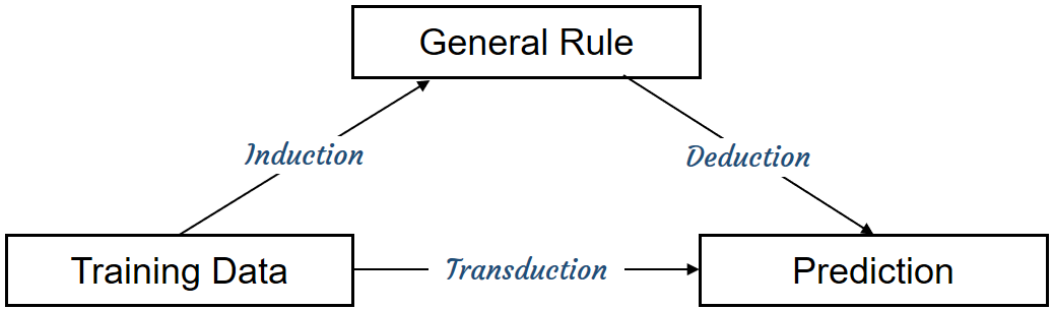


Figure 2.1: Inductive and transductive prediction.

The predictor H is often parametrized with $\theta \in \Theta$. Thus, a predictor H models $P_\theta(y \mid x)$ and estimates θ given samples from the distribution $\mathbf{D}$ over $\mathbf{X} \times \mathbf{Z}$.

2.2 OVERVIEW ON UNCERTAINTY

The provided data samples used for training an ML algorithm are often finite, limited and sometimes biased. Hence, it is logical to believe that the predictions of the algorithm hold no guarantees over their validity and

[1] The three terms example, sample and instance will be used interchangeably.

so are uncertain. Thus, it is important to quantify this uncertainty and take it into consideration in the decision-making process, especially when it comes to sensitive real-world applications.

An example can be seen in Computer Vision where the Deep Learning (DL) model makes mistakes that a human eye will not do. For instance, [Hendrycks et al., 2021] builds two natural adversarial data sets: ImageNet-A, which contains harder examples that the model should be able to classify, and ImageNet-O, which contains anomalies of unforeseen classes. Figure 2.2 shows that models can be easily fooled even by natural examples and synthetically generating them. [Messoudi et al., 2020b] presents in Figure 2.3 for a binary classification setting (with a score close to 0% being a female, and a score close to 100% being a male) an adversarial attack in which an image of a female is tweaked in different ways to make the Convolutional Neural Network (CNN) mistakenly predict it as a male.

These examples emphasize the importance of uncertainty quantification and how it can make ML and DL models more robust in order to avoid such wrong predictions, especially when it comes to high-risk real-world applications such as with self-driving cars or in the medical field.

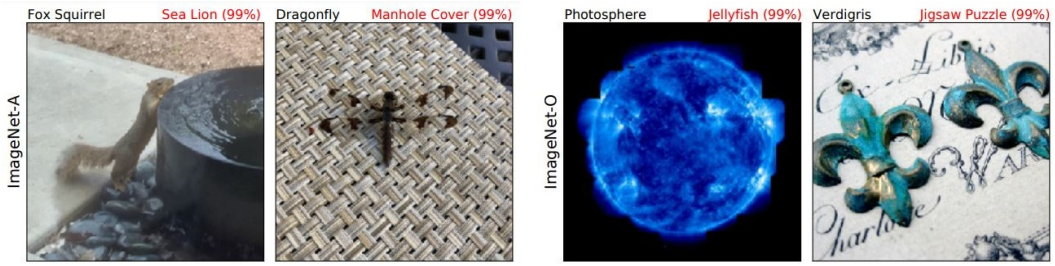


Figure 2.2: Natural adversarial examples. The black text is the actual class, and the red text is a ResNet-50 prediction and its confidence.

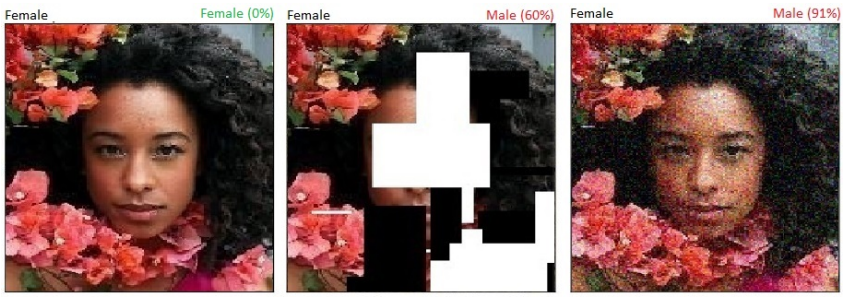


Figure 2.3: Adversarial attack examples. The black text is the actual class, and the green (when correct) and red (when incorrect) texts are a CNN prediction.

2.2.1 *Types of uncertainty*

It is quite common to divide uncertainty into two different aspects [Der Kiureghian et al., 2009]:

- **Epistemic uncertainty:** arises due to the lack of knowledge, meaning that the learner does not have enough information to make a decision. For instance, in a computer vision setting, predicting the type of round fruit (apple, plum, tangerine or orange) from sample pictures may be a difficult and uncertain task if few fruit pictures are available. This kind of uncertainty is important to account for in high-risk applications and when dealing with small, sparse data.

- **Aleatoric uncertainty:** refers to the variability in the predictions that comes from randomness. A simple example would be the outcome of a dice roll, which has 6 different results with the same probability of happening without any way of knowing on which face it is going to land.

Another way of differentiating between the types of uncertainty is to check whether we can get rid of the uncertainty or not by adding new information. If we take the same examples as above, adding new pictures to the training set will enable the learner to differentiate between round fruits. In this case, epistemic uncertainty is often referred to as "reducible" uncertainty. However, rolling the dice 1000, 10000 or 100000 times will not affect the randomness of its outcome. Aleatoric uncertainty is thus often associated with "irreducible" uncertainty.

Figure 2.4 taken from [Hüllermeier et al., 2021] shows the difference between these two types of uncertainty in a binary classification setting that tries to tell crosses and circles apart, with the prediction at the query point (indicated by a question mark). It is reducible in the left figure by adding more data, thus more knowledge is needed to eliminate other hypotheses and keep only the correct one, and non-reducible in the right figure because the two classes overlap in the region where the query point is, even if we only have one hypothesis.

2.2.2 *Sources of uncertainty*

Another way to differentiate uncertainty is by specifying on what it bears:

- **Model uncertainty:** comes from the selection of an appropriate model and an estimation of its parameters, and can happen if training and testing data distributions are generated differently, often known as data set shift. It can also come from a reduced hypothesis space and a finite sample. This form of uncertainty can

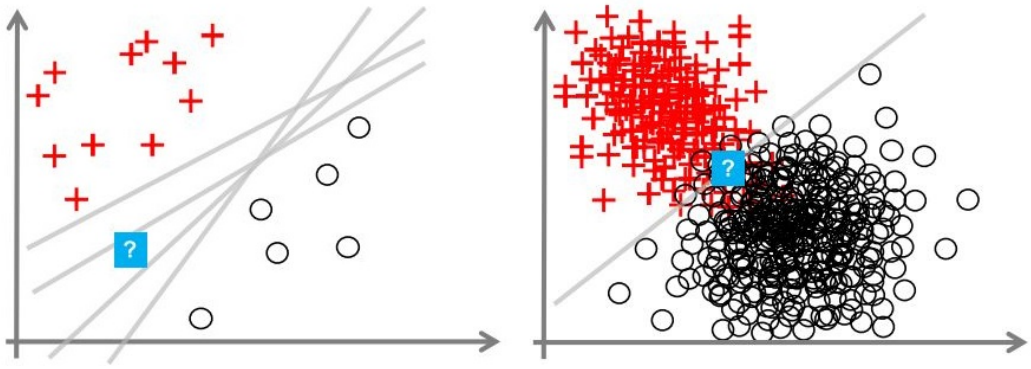


Figure 2.4: Illustrative example of reducible (left) vs. non reducible (right) uncertainty.

mostly be reduced by providing more training data, which is why it is linked to epistemic and reducible uncertainty.

- **Data uncertainty:** is a characteristic of the underlying distribution that generated the data, such as having a defective sensor that introduces a noise when measuring some input feature, or having a non-expert annotator that hardly differentiates between the labels, particularly if data is missing or if the input features are ambiguous. Since data uncertainty is intrinsically related to the data collecting process, getting more examples will not help resolve this issue. Thus, it is often connected with aleatoric and irreducible uncertainty.

Knowing which type and source of uncertainty we are dealing with is a first important step towards building robust ML and DL models, as each one of them has different techniques to quantify them and thus to treat them in order to help the model generalize better. For example, the next data points to inject into an active learning model can be chosen based on uncertainty quantification [Monarch, 2021], by determining whether the model is confused because of a lack of knowledge (as not having enough pedestrian images when it comes to a self-driving car object detection), or because of a problem in the data itself (as training on images taken during a sunny day and testing on images taken on a snow storm). The approach to solve those distinct issues will differ. For instance, the first uncertainty type can be reduced by injecting more samples (of pedestrians) and the second by adding a new feature (that describes the weather).

2.3 UNCERTAINTY ESTIMATION METHODS

In probability theory and statistics, there exists two main approaches to estimate the unknown parameters θ:

- **Bayesian statistics:** use priors and Bayes rule to estimate uncertainty from available information, by considering that the unknown parameters θ are uncertain quantities modelled by probabilities. This approach can be applied both to repeated and non-repeated events.

- **Frequentist statistics:** use sampling distribution to represent uncertainty, meaning that the unknown parameters θ are calculated by repeating the event many times.

In this section, we will present some of the methods used in both approaches to estimate predictive uncertainty.

2.3.1 *Bayesian Inference*

Bayesian inference relies on Bayes' Theorem on conditional probabilities [Bayes, 1763] to quantify the ignorance about the unknown parameters θ after knowing the distribution $\mathbf{D}$. This quantity is referred to as the *posterior distribution* $P(\theta \mid \mathbf{D})$ and is calculated as follows:

$$P(\theta \mid \mathbf{D}) = \frac{P(\theta) \cdot P(\mathbf{D} \mid \theta)}{P(\mathbf{D})} = \frac{P(\theta) \cdot P(\mathbf{D} \mid \theta)}{\int P(\theta') \cdot P(\mathbf{D} \mid \theta') \cdot d\theta'}, \tag{2.1}$$

where $P(\mathbf{D})$ is the *marginal likelihood* computed by integrating out θ, $P(\theta)$ is the *prior distribution* that defines the available information or beliefs before knowing the distribution $\mathbf{D}$, and $P(\mathbf{D} \mid \theta)$ is the *likelihood function* that captures assumptions about the data dependence on the parameters. Using Bayes' Theorem also enables us to update the calculations whenever additional data is available, which is highly useful in an online learning setting.

Once we have the posterior distribution, we can calculate the *posterior predictive distribution* of a new instance x as:

$$P(y \mid x, \mathbf{D}) = \int P(y \mid x, \theta) \cdot P(\theta \mid \mathbf{D}) \cdot d\theta \tag{2.2}$$

As mentioned earlier, Bayesian inference is effective for events that can not be repeated, which makes it more general than frequentist approaches. It is also widely adopted in many domains, such as in glaciology to estimate the sea-level rise due to the ice sheets melting [Gopalan et al., 2021], in hydrology to model water quality [Freni et al., 2010], in medical sciences to robustly diagnose tuberculosis [Abideen et al., 2020], or in autonomous driving control [Michelmore et al., 2020]. However, it has significant drawbacks, mainly related to the sensitive task of choosing a prior and the computational cost of the integral calculations of the marginal likelihood.

2.3.2 *Imprecise Probabilities*

Imprecise probability theory [Walley, 1991] is a generalization of traditional probability theory, in which a set of probability distributions $\mathscr{P}$ (i.e. a credal set) is used instead of only one precise probability measure $\mathbb{P}$. In case of high uncertainty, they can provide set-valued predictions based on lower and upper probabilities defined as:

$$\underline{P}(\theta \mid \mathbf{D}) = \inf_{\mathbb{P} \in \mathscr{P}} P(\theta \mid \mathbf{D}) \qquad \text{and} \qquad \overline{P}(\theta \mid \mathbf{D}) = \sup_{\mathbb{P} \in \mathscr{P}} P(\theta \mid \mathbf{D}) \tag{2.3}$$

When there is not enough information, [Zaffalon, 2002] considers that credal sets are better at depicting true uncertainty in noisy data. In particular, they seem well-fitted to distinguish between aleatoric and epistemic uncertainty. However, using imprecise probability approaches comes at a high computational cost, since they require handling a set of probabilities instead of one single probability.

2.3.3 *Calibration methods*

Calibration can be expressed as the requirement that a probabilistic prediction $\hat{p}(y)$ should converge to the true probability $p(y)$. This is often done through the use of proper scoring rules (for more details, please read [Gneiting et al., 2007]). Another popular means of quantifying uncertainty in a probabilistic setting is to perform a Bayesian analysis [Kendall et al., 2017], yet such analysis methods do not come with statistical or calibration guarantees.

There are different methods of calibration, but we will only present the most famous ones:

- **Platt scaling** [Platt, 1999]: This parametric calibration method turns outputs into probability distributions by using logistic regression. To do so, Platt scaling transforms a real-valued function f whose sign gives a binary classification into the function:

$$P(y = 1 \mid f) = \frac{1}{1 + \exp(af + b)},$$

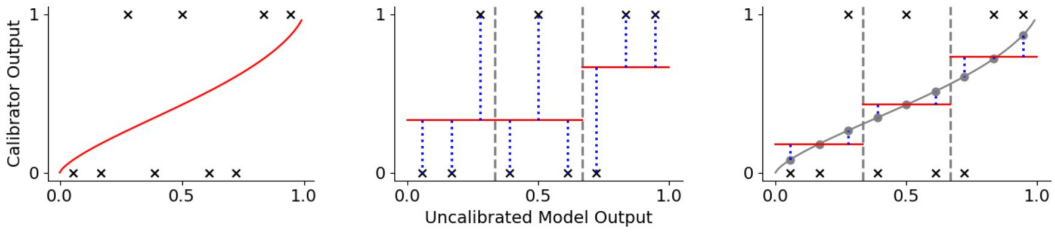
  where a and b are scalar parameters estimated using maximum likelihood on a calibration set.

- **Histogram binning** [Zadrozny et al., 2001]: This method transforms the predicted probabilities by partitioning them into m bins, and then estimating the respective posterior probability $P(y = 1 \mid B)$ for each bin B by the fraction of positive training examples. Thus, histogram

binning is a non-parametric and non-monotonic method that does not need any model to get calibrated results.

- **Isotonic regression** [Zadrozny et al., 2002]: This calibration method relies on fitting a non-parametric regression line such that it is chosen from the class of all isotonic (i.e. non-decreasing) functions. It is seen as an intermediary approach between Platt scaling and histogram binning, since it is a non-parametric monotonic function that does not rely on any model to get calibrated results. Isotonic regression is considered as the most common method of calibration used in Machine Learning.

Figure 2.5 compares between three calibration approaches: Platt scaling, histogram binning and scaling-binning (a hybrid approach that applies a scaling function and then takes the average of the function values [Kumar et al., 2019]).



(a) Platt scaling (b) Histogram binning (c) Scaling-binning calibrator

Figure 2.5: Visualization of calibration approaches. The black crosses are the ground truth labels, and the red lines are the output of the calibration methods.

To quantify calibration, *Expected Calibration Error (ECE)* [Naeini et al., 2015] is often used to compute the gap between accuracy and confidence of equally-sized bins. Plotting accuracy vs. confidence results into a *reliability diagram* as seen in Figure 2.6, with red bars reflecting this gap for each bin, which clearly state the importance of using a calibration method. For more details about calculating accuracy, confidence and ECE values as well as plotting the reliability diagram in Figure 2.6, please refer to the paper [Guo et al., 2017].

Despite these great results, calibration methods have two limitations. The first one is that these post-processing methods only improve calibrated probabilities, but do not guarantee perfectly calibrated predictions. Moreover, they give a point prediction, which does not distinguish between different sources of uncertainty.

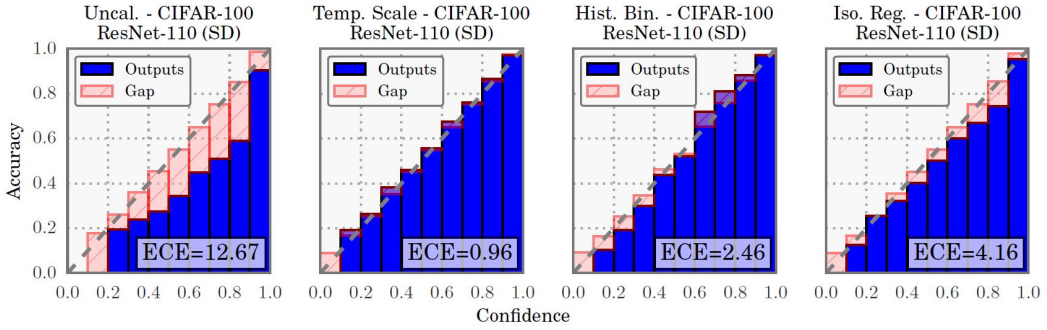


Figure 2.6: Reliability diagrams for CIFAR-100 before (far left) and after calibration (middle left, middle right, far right).

2.3.4 *Conformal Prediction*

Conformal prediction [Vovk et al., 2005] is a model-agnostic method that provides strong frequentist statistical guarantees on the error rate ϵ. It can be considered as a form of distribution-free uncertainty quantification, since it makes very little assumption about the underlying data distribution (except exchangeability), and thus is valid for many practical cases.

This method can be applied on any ML algorithm, by employing its predictions to compute a non-conformity score that is used to quantify how strange an example is compared to training examples. This score then enables us to obtain a prediction set $\hat{Y}$ such that:

$$P(y \in \hat{Y}) \geqslant 1 - \epsilon,$$

where size reflects the difficulty of the example. Figure 2.7 taken from [Angelopoulos et al., 2021] illustrates this property by showing the ability of conformal prediction to give larger prediction sets (always containing the true label) when the given image becomes more and more ambiguous.

Figure 2.7: Prediction set generated by conformal prediction for three progressively more difficult examples of the class fox squirrel.

2.4 CONCLUSION

In this chapter, a reminder on Machine Learning, more specifically supervised learning problems, was given along with a small overview on uncertainty and famous methods widely used for treating predictive uncertainty. For more details about each type of uncertainty and different ways to quantify them, the reader can refer to [Hüllermeier et al., 2021].

Since conformal prediction is the main method used in our thesis, it will be thoroughly presented in the next chapter.

Chapter 3

Conformal Prediction

> *It is also a good rule not to put overmuch confidence in the observational results that are put forward until they are confirmed by theory.*
>
> **— Sir Arthur Stanley Eddington**

Contents

Conformal prediction was initially introduced by Vovk, Gammerman and Shafer [Vovk et al., 2005] based on transduction, randomness and hypothesis testing in order to obtain prediction regions having a desired confidence. This chapter presents the main ideas behind conformal prediction in the transductive setting, and its extension to the inductive setting for both classification and regression. It also demonstrates the effectiveness of this framework through an application example for classification that we presented in our paper [Messoudi et al., 2020b]. For more details about conformal prediction, one can read the books "Algorithmic Learning in a Random World" [Vovk et al., 2005] and "Conformal Prediction for

Reliable Machine Learning: Theories and Applications" [Balasubramanian et al., 2014], and follow the tutorial [Shafer et al., 2008].

3.1 TRANSDUCTIVE CONFORMAL PREDICTION (TCP)

To explain how conformal inference works in the transductive setting, we will consider a classification problem, with the successive examples:

$$z_1 = (x_1, y_1), z_2 = (x_2, y_2), \ldots, z_n = (x_n, y_n),$$

with $x_i \in \mathbf{X}$ an object and $y_i \in \mathbf{Y}$ its label. For any sequence $z_1, z_2, \ldots, z_n \in \mathbf{Z}^*$ and any new object $x_{n+1} \in \mathbf{X}$, we can redefine the *simple predictor* $\mathbf{H}$ seen before as:

$$\mathbf{H} : \mathbf{Z}^* \times \mathbf{X} \to \mathbf{Y}. \tag{3.1}$$

This simple predictor $\mathbf{H}$ gives a point prediction $\mathbf{H}(z_1, \ldots, z_n, x_{n+1}) \in \mathbf{Y}$ for y_{n+1}, the true label of x_{n+1}.

Let $\epsilon \in (0, 1)$ be the probability of error called the *significance level* that enables this simple predictor to become a *confidence predictor* Γ such that:

$$\Gamma : \mathbf{Z}^* \times \mathbf{X} \times (0, 1) \to 2^{\mathbf{Y}}, \tag{3.2}$$

where $2^{\mathbf{Y}}$ denotes the power set of $\mathbf{Y}$. This confidence predictor can predict a subset of $\mathbf{Y}$ with a *confidence level* $1 - \epsilon$, corresponding to a statistical guarantee of covering y_{n+1}, the true label of x_{n+1}.

To build such a predictor, conformal prediction relies on a *non-conformity measure* (NCM) A_n that estimates how compliant an example z_i is to other examples in a bag $\rbag z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_n \lbag$. This A_n produces a *non-conformity score* α_i such as:

$$\alpha_i := A_n(\rbag z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_n \lbag, z_i). \tag{3.3}$$

Comparing α_i with other non-conformity scores α_j with $j \neq i$, a *p-value* π_i of z_i is used to calculate the proportion of less conforming examples than z_i, with:

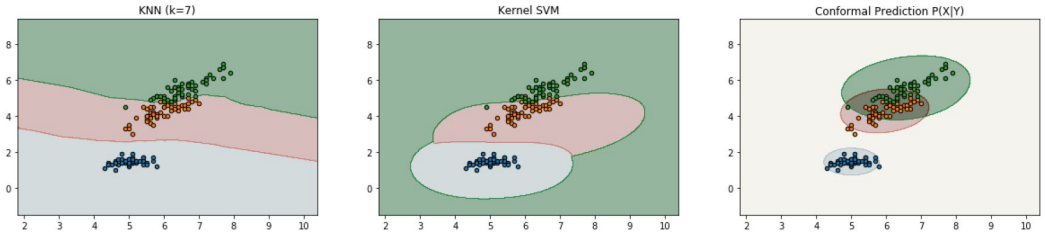
$$\pi_i = \frac{|\{j = 1, \ldots, n : \alpha_i \leqslant \alpha_j\}|}{n}. \tag{3.4}$$

If the p-value approaches the upper bound 1, then z_i is consistent compared to the other examples. On the contrary, if it approaches the lower bound $1/n$, then z_i is seen as an outlier.

For a classification problem, and to compute the p-value for the new object x_{n+1}, each possible label $y \in \mathbf{Y}$ is considered in (3.4). Hence, the conformal predictor is obtained by predicting the set:

$$\Gamma^{\epsilon}(x_{n+1}) = \left\{ y \in \mathbf{Y} : \frac{|\{i = 1, \ldots, n, n+1 : \alpha_{n+1}^{y} \leqslant \alpha_{i}^{y}\}|}{n+1} > \epsilon \right\}. \tag{3.5}$$

[Hechtlinger et al., 2018] shows in Figure 3.1 the decision boundaries on a classification problem using three methods. While the algorithms kNN and kernel SVM provide 3 regions corresponding to each class in the well-known Iris data set problem, the conformal prediction method is able to produce 2 more additional regions: the overlapping area is classified as a $\{0, 1\}$ set and the white area is classified as the empty set .



(a) (b) (c)

Figure 3.1: Decision boundaries on Iris data set with a (a) kNN, (b) kernel SVM, and (c) conformal prediction methods.

Constructing a conformal predictor therefore amounts to defining a non-conformity measure that can be built based on an ML algorithm called the *underlying algorithm* of the conformal prediction.

3.2 ASSUMPTIONS, PROPERTIES AND LIMITATIONS

One of the most interesting characteristics of conformal prediction is that it needs little assumptions to provide set predictions based on order statistics. These assumptions are either the i.i.d. condition or exchangeability, which is a weaker condition. These minimal assumptions enable the conformal inference framework to offer many interesting properties and advantages with a few drawbacks that can be overcome.

3.2.1 *i.i.d. and exchangeability assumptions*

A sequence of random variables is independent and identically distributed (i.i.d.) when the individual random variables are all drawn from the same probability distribution and are mutually independent. In other words, let $X_1, \ldots, X_n$ be n random variables in $I \subseteq \mathbb{R}$ and F_{X_i} be the distribution of X_i. $X_1, \ldots, X_n$ are i.i.d. if and only if:

$$F_{X_1}(x) = F_{X_k}(x) \qquad \forall k \in \{1, \ldots, n\} \quad \text{and} \quad \forall x \in I$$
$$F_{X_1,\ldots,X_n}(x_1, \ldots, x_n) = F_{X_1}(x_1) \ldots F_{X_n}(x_n) \qquad \forall x_1, \ldots, x_n \in I$$

If the sequence of random variables $X_1, \ldots, X_n$ is i.i.d. then the rank of X_n will be uniform over $\{1, 2, \ldots, n\}$. By choosing a probability of error (or significance level) $\epsilon \in (0, 1)$, conformal inference relies on this result to define a quantile $q_{1-\epsilon}$ based on the order statistics $X_{(1)} \leqslant X_{(2)} \leqslant \ldots \leqslant X_{(n)}$ such that $q_{1-\epsilon} = X_{(\lceil (1-\epsilon)\cdot n \rceil)}$.

Conformal prediction also holds under the exchangeability assumption, another condition that is weaker than the i.i.d. assumption. In this case, the random variables $X_1, \ldots, X_n$ are exchangeable if for every permutation τ of the indices $1, \ldots, n$, we have:

$$P(X_1, \ldots, X_n) = P(X_{\tau(1)}, \ldots, U_{\tau(n)})$$

In other words, the joint probability distribution of the original sequence is exactly the same as the joint probability distribution of the permuted sequence, meaning that the order of the random variables is not important. This is considered when handling data examples in the conformal prediction framework.

3.2.2 *Properties and advantages*

Validity

This is one of the two fundamental characteristics desired in conformal inference. It indicates that the error rate does not exceed the chosen significance level ϵ for a confidence level equal to $1 - \epsilon$, meaning that

$$P(y_{n+1} \in \Gamma^{\epsilon}(x_{n+1})) \geqslant 1 - \epsilon. \tag{3.6}$$

Efficiency

The prediction set $\Gamma^{\epsilon}(x_{n+1}) = \mathbf{Y}$ is always valid, but it does not provide any information. That is why efficiency is another desirable property of conformal prediction. It aims at obtaining prediction sets that are as small as possible, which is more informative. Thus, in a classification setting, the prediction set of object x_{n+1} must have as few classes as possible (i.e., $|\Gamma^{\epsilon}(x_{n+1})|$ should be small), and in regression, the prediction interval must be as tight as possible.

Flexibility

Conformal inference is a framework that can be applied on top of most Machine Learning algorithms, whether in classification or in regression, as we will see later. Hence, the quality of the resulting prediction sets depends on the defined non-conformity measure.

Nested prediction sets

This property refers to the fact that the confidence predictor Γ^ϵ must be decreasing for the inclusion with respect to ϵ, such that:

$$\forall n > 0, \quad \forall \epsilon_1 \geqslant \epsilon_2, \quad \Gamma^{\epsilon_1}(z_1, \ldots, z_n, x_{n+1}) \subseteq \Gamma^{\epsilon_2}(z_1, \ldots, z_n, x_{n+1}). \quad (3.7)$$

That is, the more confident we want to be, the larger Γ^ϵ should be. Figure 3.2 shows this concept of nested conformal regions with different confidence levels.

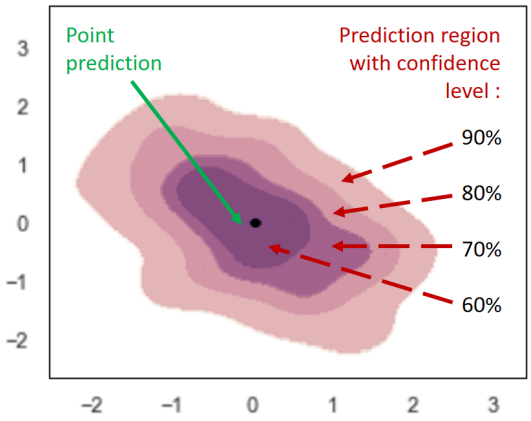


Figure 3.2: Illustration of conformal nested predictions.

3.2.3 *Limitations*

One important drawback of conformal prediction in the transductive setting is that it is not computationally efficient, since the non-conformity measure needs to be derived from all previous examples for each new example. Thus, this approach is defective when dealing with a large amount of data, especially for any time consuming training tasks such as Deep Learning models.

Another limitation that is worth mentioning is that conformal predictors are not conditionally valid, meaning that the confidence level is only guaranteed on all data generally instead of subsets of data. That is, we do have $P(y_{n+1} \in \Gamma^\epsilon(x_{n+1})) \geqslant 1 - \epsilon$ but not $P(y_{n+1} \in \Gamma^\epsilon(x_{n+1}) \mid x_{n+1}) \geqslant 1 - \epsilon$. An example where this could be problematic would be the case of highly imbalanced data, where the minority class's participation in the validity is largely insignificant, since it is dominated by the majority class. Thus, the probability of error for examples in the minority class will be much higher.

Both of these limitations can be overcome with other variants of conformal prediction, respectively using Inductive and Mondrian conformal prediction. These will be detailed in the following sections.

3.3 INDUCTIVE CONFORMAL PREDICTION (ICP)

Inductive Conformal Prediction (ICP) is a method presented in [Papadopoulos, 2008] that replaces the transductive setting with an inductive one, in order to solve the computational inefficiency problem. In this case, the predictor derives a general rule from observing training examples, that is then applied to the test examples.

In the inductive approach, the original training data set $\{z_1, \ldots, z_n\}$ is split into two parts. The first part $\mathbf{Z}^{tr} = \{z_1, \ldots, z_l\}$ is called the *proper training set*, and is used to train the underlying algorithm on which the non-conformity measure A_l is based. The second smaller part $\mathbf{Z}^{cal} = \{z_{l+1}, \ldots, z_n\}$ is called the *calibration set*. For each example of the calibration set $i = l+1, \ldots, n$, a non-conformity score α_i is calculated once by applying (3.3) to get the sequence $\alpha_{l+1}, \ldots, \alpha_n$, which is used when computing the p-value for the new object x_{n+1} and obtaining the set $\Gamma^{\epsilon}(x_{n+1})$.

3.3.1 *ICP for classification*

In a classification problem, the objective is to associate to an object $x_i \in \mathbf{X}$ a certain class among a set of labels with $y_i \in \mathbf{Y} = \{C_1, \ldots, C_p\}$. Given any new object $x_{n+1} \in \mathbf{X}$, it is possible to predict $y_{n+1} \in \mathbf{Y}$ by following the inductive conformal framework steps:

1. Split the original data set $\mathbf{Z}$ into a *proper training set* $\mathbf{Z}^{tr}$ with $|\mathbf{Z}^{tr}| = l$ and a *calibration set* $\mathbf{Z}^{cal}$ with $|\mathbf{Z}^{cal}| = n - l = q$.

2. Train a classification *underlying algorithm* $H : \mathbf{X} \to \mathbf{Y}$ on $\mathbf{Z}^{tr}$ to obtain the *non-conformity measure* A_l. A usual non-conformity measure in a classification setting when h is a probabilistic classifier is defined as follows:
$$A_l = 1 - \hat{P}_H[y \mid x]. \tag{3.8}$$

3. Apply the non-conformity measure A_l to each example z_i of $\mathbf{Z}^{cal}$ to get the non-conformity scores $\alpha_1, \ldots, \alpha_q$.

4. Choose a *significance level* $\epsilon \in (0, 1)$ to get a prediction set with a *confidence level* of $1 - \epsilon$.

5. For a new object x_{n+1}, compute a non-conformity score for each class $C_k \in \mathbf{Y}$:
$$\alpha_{n+1}^{C_k} = A_l((x_{n+1}, y = C_k)). \tag{3.9}$$

6. For each class $C_k \in \mathbf{Y}$, compute the p-value:
$$\pi_{n+1}^{C_k} = \frac{|\{i \in 1, \ldots, q, n+1 : \alpha_{n+1}^{C_k} \leqslant \alpha_i\}|}{q+1}. \tag{3.10}$$

7. Build the prediction set:

$$\Gamma^{\epsilon}(x_{n+1}) = \{C_k \in \mathbf{Y} : \pi^{C_k}_{n+1} > \epsilon\}. \tag{3.11}$$

Figure 3.3 shows the different cases that can occur in a conformal prediction set depending on calibration data, the chosen ϵ, and Equation (3.11). The prediction set can be a singleton when the predictor is sure (in blue), a set with more than one class (in green) in case of ambiguity and an empty set $\emptyset$ (in red) when the model does not know or did not see a similar example during training.

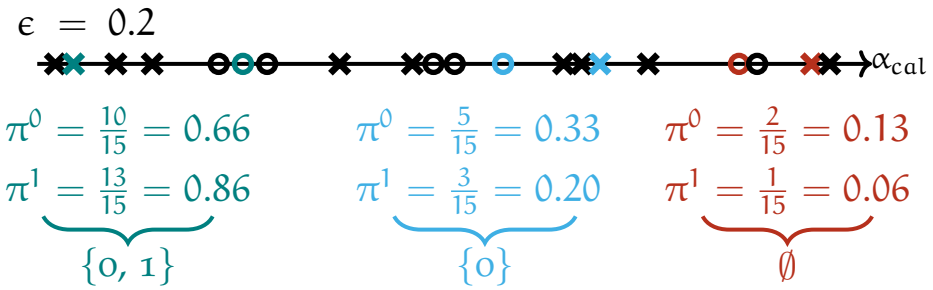


Figure 3.3: Illustrative examples of inductive conformal prediction sets for a binary classification problem.

3.3.2 *ICP for regression*

In a regression problem, the objective is to associate to an object $x_i \in \mathbf{X}$ a real value with $y_i \in \mathbb{R}$ its label. In conformal prediction, it is not possible in regression to replace $\hat{y}$ with all possible values in $\mathbb{R}$ when calculating the non-conformity score for each new object x_{n+1}. Thus, the result of a conformal regressor is to produce a prediction interval depending on the chosen significance level. In this case, the inductive conformal setting follows these steps:

1. Split the original training data set $\mathbf{Z} = \langle z_1, \ldots, z_n \rangle$ into a *proper training set* $\mathbf{Z}^{tr} = \langle z_1, \ldots, z_l \rangle$, with $|\mathbf{Z}^{tr}| = l$ and a *calibration set* $\mathbf{Z}^{cal} = \langle z_{l+1}, \ldots, z_n \rangle$, with $|\mathbf{Z}^{cal}| = n - l = q$.

2. Train a regression underlying algorithm on $\mathbf{Z}^{tr}$, and get the *non-conformity measure* A_l. The standard non-conformity measure is the absolute difference between the actual value y_i and the predicted value $\hat{y}_i$ by the underlying algorithm:

$$A_l = \alpha_i = |y_i - \hat{y}_i|. \tag{3.12}$$

3. For each example z_i of $\mathbf{Z}^{cal}$, calculate the non-conformity score α_i to get the sequence $\alpha_1, \ldots, \alpha_q$.

4. Set a significance level $\epsilon \in (0, 1)$ depending on the desired confidence level.

5. Sort the calibration non-conformity scores $\alpha_1, \ldots, \alpha_q$ in a descending order and get the index of the $(1-\epsilon)$-percentile of the non-conformity score α_s, such as $\alpha_s := \alpha_{\lceil (1-\epsilon)\cdot q \rceil}$. Thus, we obtain:

$$P(\alpha_i \leqslant \alpha_s) \geqslant 1 - \epsilon. \tag{3.13}$$

6. For a new object x_{n+1}, get the underlying algorithm's prediction $\hat{y}_{n+1}$ and compute its conformal prediction interval:

$$\Gamma^{\epsilon}(x_{n+1}) = [\hat{y}_{n+1} - \alpha_s, \hat{y}_{n+1} + \alpha_s]. \tag{3.14}$$

Using the standard non-conformity measure in (3.12) means that all prediction intervals have the same size $2\alpha_s$. Hence, the standard prediction interval does not reflect the difficulty of predicting y_i for each individual example x_i. Using a *normalized* non-conformity measure provides individual bounds for each example by scaling the standard non-conformity measure with a difficulty estimator σ_i, so that the prediction interval is smaller for "easy" examples, and bigger for "hard" examples. In this case, the non-conformity measure becomes:

$$\alpha_i = \frac{|y_i - \hat{y}_i|}{\sigma_i}. \tag{3.15}$$

Thus, we have:

$$P\left(\frac{|y_i - \hat{y}_i|}{\sigma_i} \leqslant \alpha_s\right) \geqslant 1 - \epsilon, \tag{3.16}$$

which becomes an equality if the method is perfectly calibrated. For a new object x_{n+1}, the prediction interval is expressed as:

$$\Gamma^{\epsilon}(x_{n+1}) = [\hat{y}_{n+1} - \alpha_s \sigma_{n+1}, \hat{y}_{n+1} + \alpha_s \sigma_{n+1}]. \tag{3.17}$$

There are a lot of ways to calculate σ_i. Table 3.1 presents some of the well-known methods to do so.

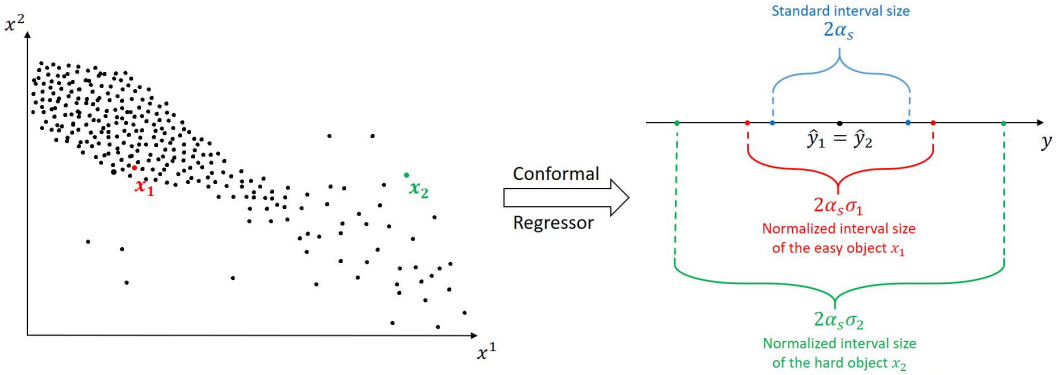
Figure 3.4 illustrates the difference in interval sizes between two objects that have the same prediction from the underlying regressor, but have different difficulty estimator values, mainly since object x_1 is located in a high density region, making it easier to predict compared to x_2, which is located in a low density region.

3.4 MONDRIAN CONFORMAL PREDICTION (MCP)

Mondrian Conformal Prediction (MCP) is a variant of conformal prediction that provides a guarantee on a subset of the data set, or on specific

Normalizing Algorithm	Difficulty Estimator	Description
Artificial Neural Network [Papadopoulos et al., 2011b]	$\sigma_i = \exp(\mu_i) + \beta$, with $\mu_i = \ln(\lvert y_i - \hat{y}_i \rvert)$, $\beta \geqslant 0$ is a sensitivity parameter.	Estimates the error of the underlying algorithm by predicting the value μ_i.
k-nearest neighbors [Papadopoulos et al., 2011a]	$\sigma_i = \exp(\gamma\lambda_i^k)$, with $\lambda_i^k = \frac{d_i^k}{\text{median}(d_j^k, z_j \in \mathbf{Z}^{tr})}$, where $d_i^k = \sum_{j=1}^{k} \delta(x_i, x_{i_j})$, δ is a distance and $\gamma \geqslant 0$ is a sensitivity parameter.	Measures d_i^k, the sum of the distance of z_i from its k-nearest neighbors $x_{i_1}, \ldots, x_{i_k}$, and normalizes it with the median of all d_j^k over all training examples.
k-nearest neighbors [Papadopoulos et al., 2011a]	$\sigma_i = \exp(\gamma\lambda_i^k) + \exp(\rho\xi_i^k)$, with $\xi_i^k = \frac{s_i^k}{\text{median}(s_j^k, z_j \in \mathbf{Z}^{tr})}$, where $s_i^k = \sqrt{\frac{1}{k}\sum_{j=1}^{k}(y_{i_j} - \overline{y_{i_{1\ldots k}}})^2}$ and $\overline{y_{i_{1\ldots k}}} = \frac{1}{k}\sum_{j=1}^{k} y_{i_j}$, $\rho \geqslant 0$ is a sensitivity parameter.	Adds the parameter ξ_i^k to the previous difficulty estimator that calculates the standard deviation s_i^k of the outputs and scales it with the median of all standard deviations s_j^k of all training examples.

Table 3.1: Difficulty estimator examples for normalized non-conformity measures in regression.



Visualisation of a 2D projection of the object space

Conformal interval sizes of two objects with the same regressor prediction

Figure 3.4: Conformal interval predictions of two examples with the same regressor prediction value.

categories of the data set. This variant is originally established for a classification problem by creating class-conditional or attribute-conditional categories [Vovk et al., 2003]. However, a mondrian version exists for regressors [Boström et al., 2020]. We will only focus on the class-conditional conformal classifier that guarantees an error rate for each class based on the chosen confidence level.

The main difference between an inductive conformal classifier and a class-conditional conformal classifier is in the computation of the p-value (3.10), in which, instead of taking all non-conformity scores α_i in the calibration set, we only consider those related to the examples belonging to the same class we are hypothetically testing for the object x_{n+1}. The p-value becomes:

$$\pi_{n+1}^{C_k} = \frac{\lvert\{i \in 1, \ldots, q : y_i = C_k, \alpha_{n+1}^{C_k} \leqslant \alpha_i\}\rvert}{\lvert\{i \in 1, \ldots, q : y_i = C_k\}\rvert}. \tag{3.18}$$

Class-conditional MCP is mostly used when data is imbalanced, in order to maintain the same error rate even for the minority class. Figure 3.5 illustrates the prediction sets that we can obtain using the same examples, calibration data and chosen ϵ as in Figure 3.3 for ICP, but with Equation (3.18) instead. This shows the changes in predictions for the blue example.

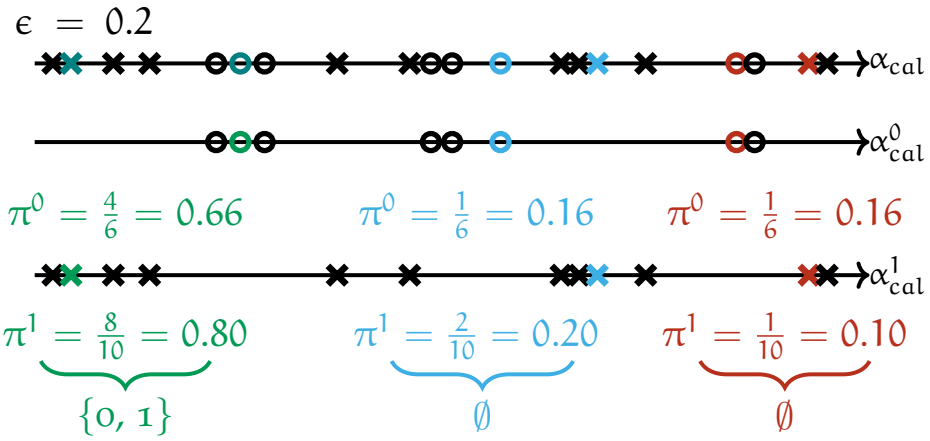


Figure 3.5: Illustrative examples of class-conditional MCP sets for a binary classification problem.

3.5 EXAMPLE: DENSITY-BASED CONFORMAL PREDICTION FOR CLASSIFICATION

This section will showcase the relevance and flexibility of conformal prediction, by applying it to different data types and ML problems based on [Hechtlinger et al., 2018] and presented in our paper [Messoudi et al., 2020b].

3.5.1 *Method*

The main idea of this approach relies on a density estimate $\hat{p}(x \mid y)$ of $p(x \mid y)$ for the label $y \in Y$. These are then used to build a class-conditional threshold $\hat{t}_y$ to which will be compared future observations such as:

$$\hat{t}_y = \sup\left\{ t : \frac{1}{n_y} \sum_{\{z_i \in \mathbf{Z}^{cal}_y\}} I(\hat{p}(x_i \mid y) \geqslant t) \geqslant 1 - \epsilon \right\}, \tag{3.19}$$

where n_y is the number of elements belonging to the class y in $\mathbf{Z}^{cal}$, and $\mathbf{Z}^{cal}_y = \{z_i \in \mathbf{Z}^{cal} : y_i = y\}$ is the subset of calibration examples of class y. For a new object x_{n+1}, we set the conformal predictor Γ^ϵ_d such that:

$$\Gamma^\epsilon_d(x_{n+1}) = \left\{ y \in Y : \hat{p}(x_{n+1} \mid y) \geqslant \hat{t}_y \right\}. \tag{3.20}$$

We can rewrite (3.20) so that it approaches (3.11) with a few differences. The main ones are the fact that Γ^ϵ_d uses a conformity measure (calculating how much an example is compliant with the others) instead of a non-conformity measure as in Γ^ϵ, with $\alpha^y_i = -\hat{p}(x_i \mid y)$ [Vovk et al., 2005], and that the number of examples used to build the prediction set depends on y. Another difference is the use of "greater or equal" sign in equations (3.21) and (3.19) that should be replaced by a "greater" sign in order to have an equivalence, and thus rewrite Γ^ϵ_d as:

$$\Gamma^\epsilon(x_{n+1}) = \left\{ y \in Y : \frac{|\{z_i \in \mathbf{Z}^{cal}_y : \alpha^y_i \geqslant \alpha^y_{n+1}\}|}{n_y} > \epsilon \right\}. \tag{3.21}$$

Proposition 1 *The equations*

$$\Gamma^\epsilon(x_{n+1}) = \left\{ y \in Y : \frac{|\{z_i \in \mathbf{Z}^{cal}_y : \alpha^y_i \geqslant \alpha^y_{n+1}\}|}{n_y} > \epsilon \right\}, \tag{3.22}$$

$$\textit{and} \quad \Gamma^\epsilon_d(x_{n+1}) = \{y \in Y : \hat{p}(x_{n+1} \mid y) > \hat{t}_y\}, \tag{3.23}$$

$$\textit{such that} \quad \hat{t}_y = \sup \left\{ t : \frac{1}{n_y} \sum_{\{z_i \in \mathbf{Z}^{cal}_y\}} I(\hat{p}(x_i \mid y) > t) \geqslant 1 - \epsilon \right\},$$

are equivalent.

Proof 1 *Let* $f(t)$ *be the decreasing function*

$$f(t) = \frac{1}{n_y} \sum_{\{z_i \in \mathbf{Z}^{cal}_y\}} I(\hat{p}(x_i \mid y) > t).$$

Let us prove that (3.23) $\implies$ *(3.22).*

Let $y \in \Gamma^\epsilon_d(x_{n+1})$. *Since* $\hat{t}_y$ *is the upper bound such that* $f(\hat{t}_y) \geqslant 1 - \epsilon$, *then* $\hat{p}(x_{n+1} \mid y)$ *does not satisfy this inequality, thus:*

$$\begin{aligned} f(\hat{p}(x_{n+1} \mid y)) &= \frac{1}{n_y} \sum_{\{z_i \in \mathbf{Z}^{cal}_y\}} I(\hat{p}(x_i \mid y) > \hat{p}(x_{n+1} \mid y)) < 1 - \epsilon \\ &= \frac{1}{n_y} \sum_{\{z_i \in \mathbf{Z}^{cal}_y\}} 1 - I(\hat{p}(x_i \mid y) \leqslant \hat{p}(x_{n+1} \mid y)) < 1 - \epsilon \\ &= 1 - \frac{1}{n_y} \sum_{\{z_i \in \mathbf{Z}^{cal}_y\}} I(\hat{p}(x_i \mid y) \leqslant \hat{p}(x_{n+1} \mid y)) < 1 - \epsilon \end{aligned}$$

Thus we have:

$$\frac{1}{n_y} \sum_{\{z_i \in \mathbf{Z}^{cal}_y\}} I(\hat{p}(x_i \mid y) \leqslant \hat{p}(x_{n+1} \mid y)) > \epsilon \tag{3.24}$$

Since $\hat{p}(x_{n+1} \mid y)$ *is a conformity score, whereas* α_i^y *is a non-conformity score, we can write* $\hat{p}(x_{n+1} \mid y) = -\alpha_i^y$ *[Vovk et al., 2005]. So (3.24) becomes*

$$\frac{1}{n_y} \sum_{\{z_i \in \mathbf{Z}_y^{cal}\}} I(\alpha_i^y \geqslant \alpha_{n+1}^y) > \epsilon \implies \frac{|\{z_i \in \mathbf{Z}_y^{cal} : \alpha_i^y \geqslant \alpha_{n+1}^y\}|}{n_y} > \epsilon$$

This shows that (3.23) $\implies$ *(3.22).*

Let us now prove that (3.22) $\implies$ *(3.23). Using the indicator function of the complement, and changing the non-conformity score into a conformity score as shown before, we can simply find that*

$$\frac{|\{z_i \in \mathbf{Z}_y^{cal} : \alpha_i^y \geqslant \alpha_{n+1}^y\}|}{n_y} > \epsilon \implies \frac{1}{n_y} \sum_{\{z_i \in \mathbf{Z}_y^{cal}\}} I(\hat{p}(x_i \mid y) > \hat{p}(x_{n+1} \mid y)) < 1 - \epsilon$$

Using the same function f*, we then have*

$$f(\hat{p}(x_{n+1} \mid y)) < 1 - \epsilon. \tag{3.25}$$

Let us show by contradiction that $\hat{p}(x_{n+1} \mid y) > \hat{t}_y$.
Suppose that $\hat{p}(x_{n+1} \mid y) \leqslant \hat{t}_y$. *Since* f *is a decreasing function, we have* $f(\hat{p}(x_{n+1} \mid y)) \geqslant f(\hat{t}_y)$. *By the definition of* $\hat{t}_y$*, we have* $f(\hat{t}_y) \geqslant 1 - \epsilon$. *Thus* $f(\hat{p}(x_{n+1} \mid y)) \geqslant f(\hat{t}_y) \geqslant 1 - \epsilon$. *However, this contradicts (3.25). So we proved that (3.22)* $\implies$ *(3.23), which concludes the proof.*

■

The training and prediction algorithms for density-based conformal classification are defined in the algorithms 1 and 2.

Algorithm 1 Training algorithm

Input: Training data $\mathbf{Z} = (x_i, y_i)$, $i = 1 \ldots n$, Class list $\mathbf{Y}$, Confidence level ϵ, Ratio p.
Initialize: $\hat{p}_{list} = list, \hat{t}_{list} = list$
for $y \in \mathcal{Y}$ **do**
  $\mathbf{X}_y^{tr}, \mathbf{X}_y^{cal} \leftarrow SubsetData(\mathbf{Z}, \mathbf{Y}, p)$
  $\hat{p}_y \leftarrow LearnDensityEstimator(\mathbf{X}_y^{tr})$
  $\hat{t}_y \leftarrow Quantile(\hat{p}_y(\mathbf{X}_y^{cal}), \epsilon)$
  $\hat{p}_{list}.append(\hat{p}_y); \hat{t}_{list}.append(\hat{t}_y)$
end for
return $\hat{p}_{list}, \hat{t}_{list}$

Algorithm 2 Prediction algorithm

Input: Input to be predicted x, Trained $\hat{p}_{list}, \hat{t}_{list}$, Class list $\mathbf{Y}$.
Initialize: $C = list$
for $y \in \mathbf{Y}$ **do**
  if $\hat{p}_y(x) \geqslant \hat{t}_y$ **then**
    $C.append(y)$
  end if
end for
return C

3.5.2 *Experimental setting*

To examine the effectiveness of the conformal method on different data types, three data sets for binary classification were used. They are **CelebA** [Liu et al., 2015], an image data set to determine a person's gender, **IMDb** [Maas et al., 2011], a textual data set describing film reviews for sentiment analysis, and **EGSS** [Arzamasov, 2018], a tabular data set for the study of electrical networks stability.

The overall approach[1] (presented in Figure 3.6) follows the same steps as in ICP for classification and meets the conditions listed above (the i.i.d. or exchangeability assumptions). The detailed approach is as follows:

1. Divide the data set into proper training $\mathbf{Z}^{tr} = (\mathbf{X}^{tr}, \mathbf{Y}^{tr})$, calibration $\mathbf{Z}^{cal} = (\mathbf{X}^{cal}, \mathbf{Y}^{cal})$ and test $\mathbf{Z}^{ts} = (\mathbf{X}^{ts}, \mathbf{Y}^{ts})$ sets.

2. Use $\mathbf{Z}^{tr}$ to train the DL model corresponding to each type of data.

3. Exploit the last before dense layer as a feature extractor to obtain a vector of size 50 that represents the object (image for CelebA, text for IMDb, or vector for EGSS).

4. Use a Gaussian kernel density estimator of bandwidth 1 available in Python's scikit-learn on $\mathbf{Z}_y^{tr}$ of each class y.

5. Compute the density scores $\hat{p}(x_{cal} \mid y)$ for calibration data $\mathbf{Z}_y^{cal}$ and sort them in descending order to determine the chosen ϵ threshold $\hat{t}_y$ for each class y to delimit their density regions.

6. With $\mathbf{Z}^{ts}$, compute the density score of each object for each class $\hat{p}(x^{ts} \mid y)$ and compare them to $\hat{t}_y$ to obtain the prediction set $\Gamma_d^{\epsilon}(x^{ts})$.

The architecture deployed for CelebA for computer vision is shown in Figure 3.7 and described as follows:

[1] The code is available on GitHub at https://github.com/M-Soundouss/density_based_conformal_prediction

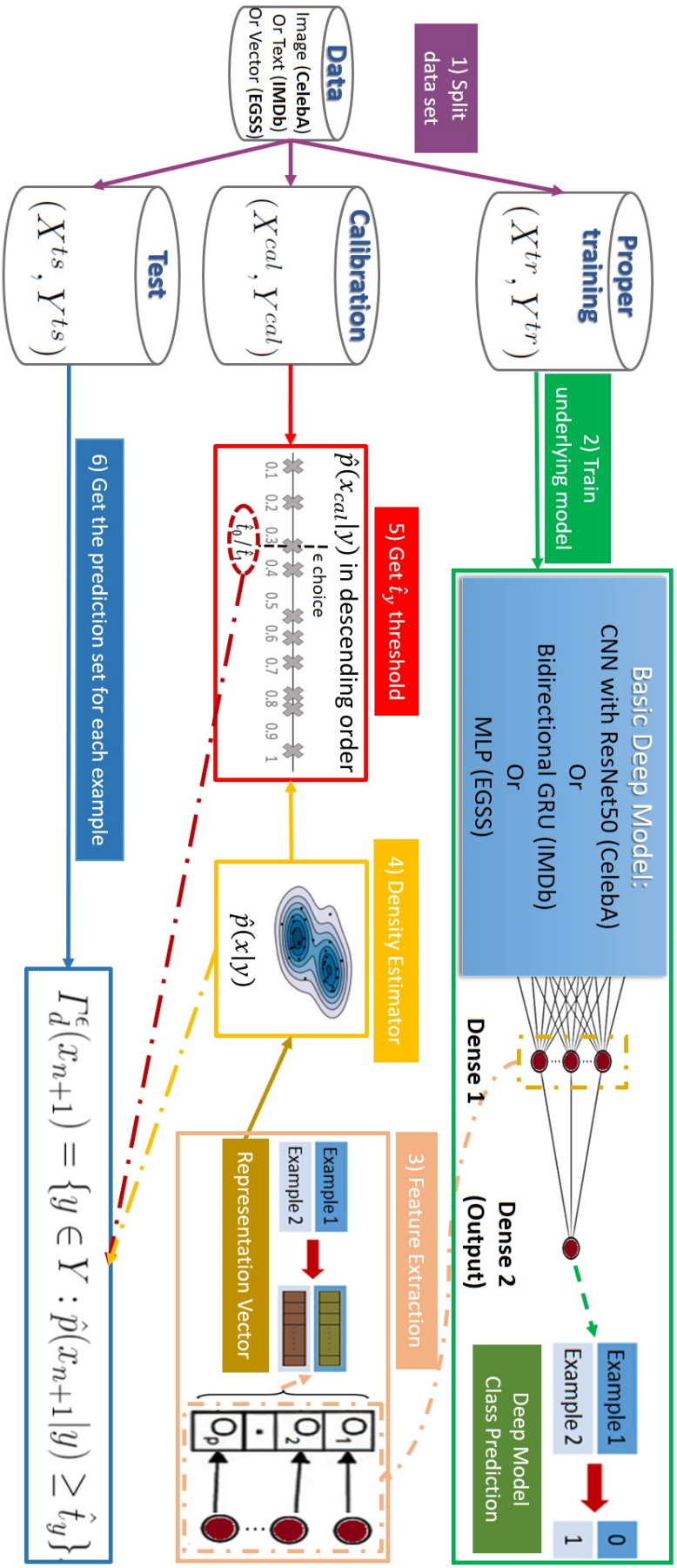


Figure 3.6: Density-based conformal prediction approach.

- Load the ResNet50 base model [He et al., 2016] pre-trained on the data set ImageNet [Deng et al., 2009].
- Fit the model to CelebA by adding dropouts and max pooling to the output.
- Use an intermediate dense layer as a feature extractor with a vector of size 50 representing the image, which will be used later for conformal prediction (image representation).
- Add a dense layer with one single neuron to obtain the class of the image (male or female).

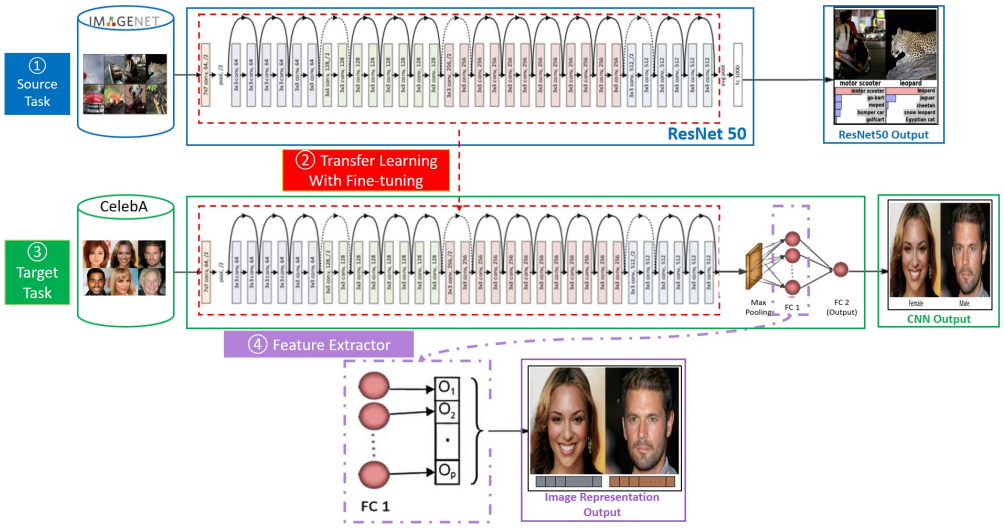


Figure 3.7: Architecture of the CNN model for CelebA.

The architecture implemented for IMDb for opinion mining is shown in Figure 3.8 and described as follows:

- Do an embedding of the input data to transform the positive integers into dense vectors of fixed size 50.
- Apply a bidirectional GRU (Gated Recurrent Unit).
- Apply max pooling and a dense layer of size 256.
- Apply a second dense layer of size 50 that will be used as a feature extractor with a vector of size 50 representing the text, which will be dedicated later for conformal prediction (text representation).
- Add a dense layer with one single neuron to obtain the class of the text (positive or negative).

The architecture used for EGSS to deal with tabular data is shown in Figure 3.9 and described as follows:

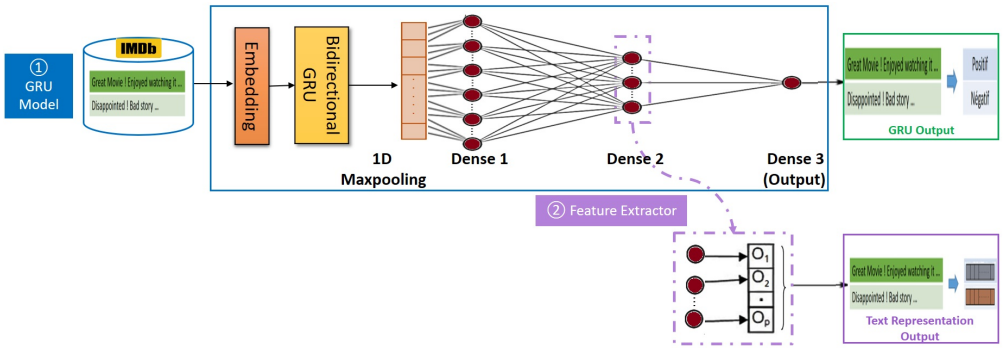


Figure 3.8: Architecture of the GRU model for IMDb.

- Apply a dense layer of size 128 on the input data.
- Fit the model to CelebA by adding dropouts and max pooling to the output.
- Use an intermediate dense layer as a feature extractor with a vector of size 50 representing the vector, which will be used later for conformal prediction (vector representation).
- Add a dense layer with one single neuron to obtain the class of the vector (stable or unstable).

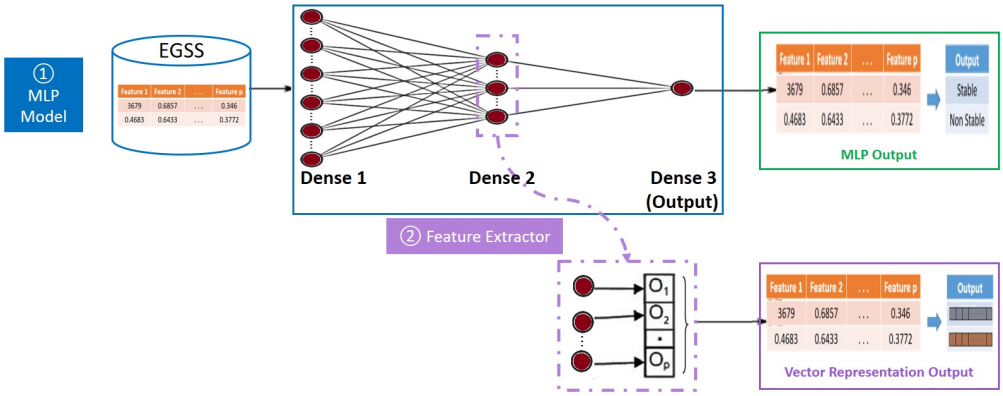
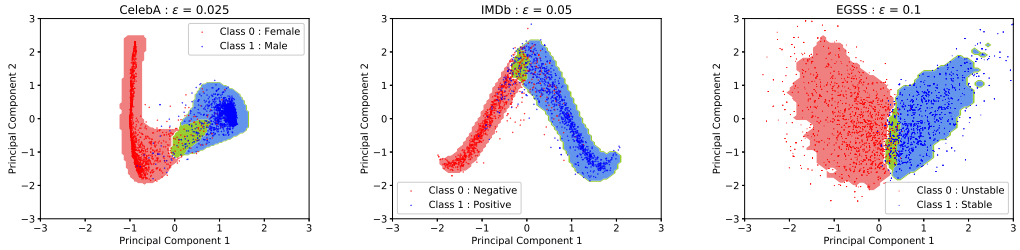


Figure 3.9: Architecture of the MLP model for EGSS.

Figure 3.10 visualizes the density regions via a Principal Component Analysis with a dimension reduction from $\mathbb{R}^{50}$ to $\mathbb{R}^2$. Results show the distinct regions of the classes 0 (in red) and 1 (in blue) with a non-empty intersection (in green) representing a region of aleatoric uncertainty, i.e. an ambiguity region with both classes overlapping. In this case, the conformal classifier returns a $\{0, 1\}$ set. The points outside these three regions belong to the region of epistemic uncertainty, meaning a poorly informed region with out-of-distribution data. In this case, the conformal classifier "does not know" and returns an empty set $\emptyset$.



(a) CelebA **(b)** IMDb **(c)** EGSS

Figure 3.10: Conformal prediction density regions for all data sets.

3.5.3 *Results on test examples*

To obtain more information, we calculated the accuracies on $\mathbf{Z}^{ts}$ with different values for $0.01 \leqslant \epsilon \leqslant 0.5$, by considering $\hat{y}_i^{DL}$ the prediction of the underlying DL model (CNN, GRU or MLP), $\hat{y}_i^{CONF}$ the prediction of the conformal pipeline, and by introducing the term $V = \{i \mid \hat{y}_i^{CONF} \in \{\{0\},\{1\}\}\}$, i.e. the conformal singleton prediction samples. These accuracies are:

- **DL accuracy:** the accuracy of the underlying DL model, meaning:

$$\text{DL accuracy} = \frac{1}{n}\sum_i I_{\hat{y}_i^{DL}=y_i}.$$

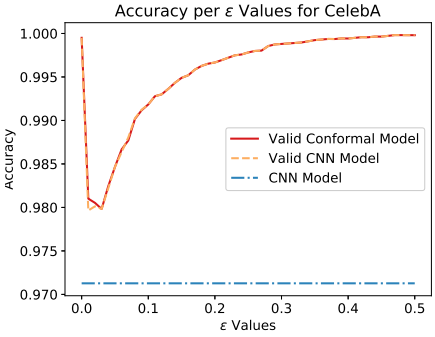
- **Valid conformal accuracy:** the accuracy of the conformal classifier when only considering the singleton predictions 0 or 1 (without taking into account the $\{0,1\}$ and the empty sets), expressed as:

$$\text{Valid conformal accuracy} = \frac{1}{|V|}\sum_{i\in V} I_{\hat{y}_i^{CONF}=y_i}.$$

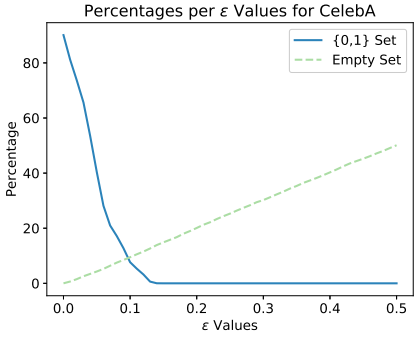
- **Valid DL accuracy:** the accuracy of the underlying DL model on test examples for which the conformal classifier predicts 0 or 1, i.e.:

$$\text{Valid DL accuracy: } = \frac{1}{|V|}\sum_{i\in V} I_{\hat{y}_i^{DL}=y_i}.$$

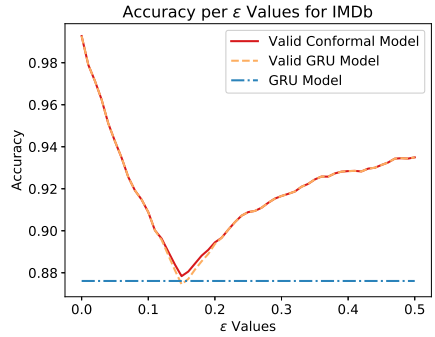
The percentage of empty sets $\emptyset$ and $\{0,1\}$ sets was also calculated from all the predictions made by the conformal classifier on $\mathbf{Z}^{ts}$. Results are shown on Figure 3.11.



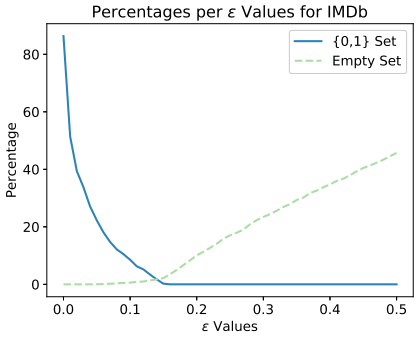
(a) Accuracy for CelebA



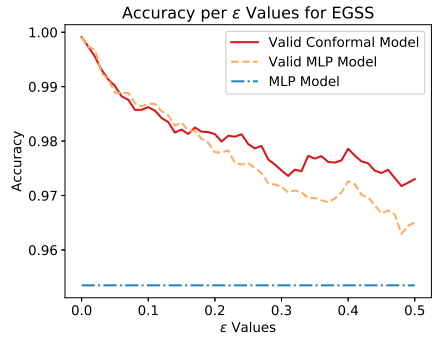
(b) Percentages for CelebA



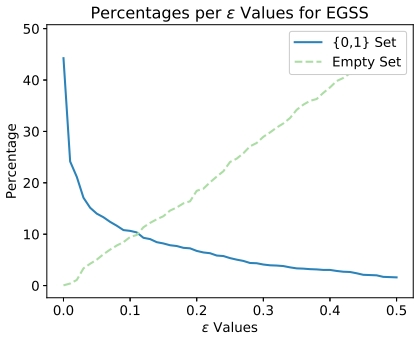
(c) Accuracy for IMDb



(d) Percentages for IMDb



(e) Accuracy for EGSS



(f) Percentages for EGSS

Figure 3.11: The accuracy and the percentages according to ϵ for CelebA (top), IMDb (middle) and EGSS (bottom).

When it comes to accuracy, results show that the addition of conformal prediction to a deep learning model does not degrade its performance when it predicts only one class, but even improves it (for EGSS). The conformal approach abstains from predicting (empty set $\emptyset$) or predicts both classes for ambiguous examples, thus making it possible to have a more reliable prediction of the label. It is also noticed that as ϵ grows, the percentage of predicted $\{0, 1\}$ sets decreases until it is no longer predicted (at $\epsilon = 0.15$ for CelebA for example). Conversely, the opposite is observed with the percentage of empty sets $\emptyset$ which escalates as ϵ increases.

3.5.4 *Illustration on noisy and foreign examples*

CELEBA Two types of noise were introduced: a noise masking parts of the face and another Gaussian on all the pixels. These perturbations and their predictions are illustrated in Figure 3.12 with "CNN" the prediction of the underlying DL model and "CNN + CP" that of the conformal classifier. This example shows that both models correctly identify the woman in image (a). However, by masking image (b), the CNN predicts it as a man with a score of 0.6 whereas the model of conformal prediction is more cautious by indicating that it does not know (∅). When applying a Gaussian noise over the whole image (c), the CNN predicts that it is a man with a larger score of 0.91, whereas the conformal classifier predicts both classes. For outliers, examples (d), (e), and (f) illustrate the ability of the conformal model to identify different outliers as such (∅) in contrast to the deep model that predicts them as men with a high score.

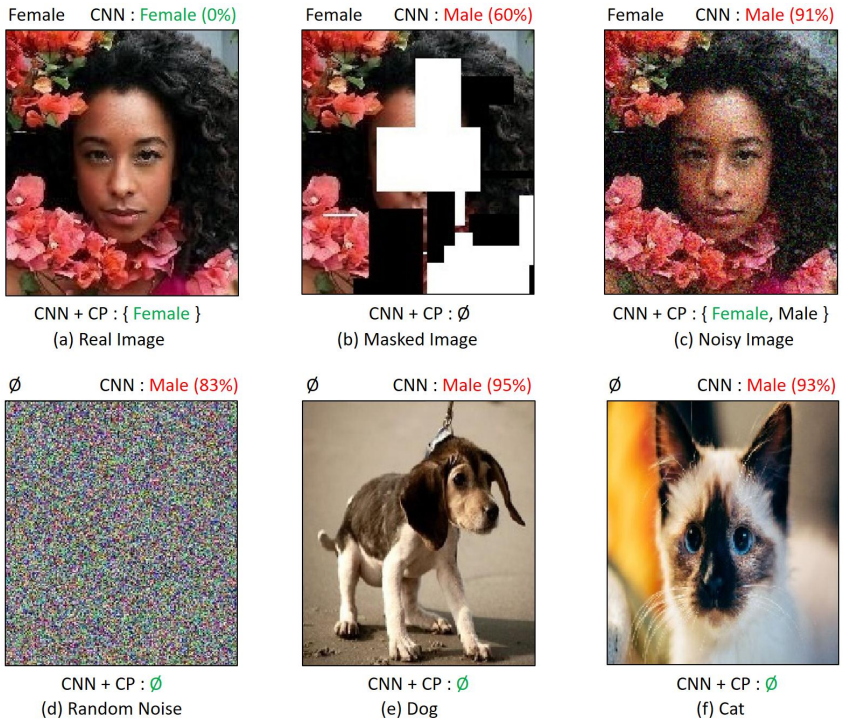


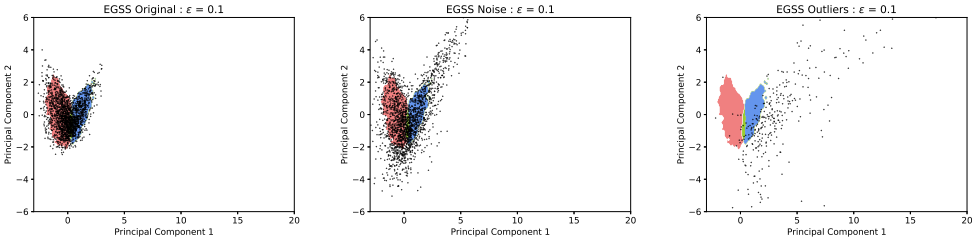
Figure 3.12: Examples of noisy and outlier images for CelebA.

IMDB Figure 3.13 displays a comparison of two texts before and after randomly changing a few words (in bold) by others in the model's vocabulary. The actual text is correctly predicted as negative by both models, but once it is noisy, it becomes positive for the GRU. However, the conformal classifier is more cautious by indicating that it can be both cases ($\{0, 1\}$). For the outlier example formed completely of vocabulary words, the GRU model predicts positive with a score of 0.99, while the conformal classifier correctly identifies it as meaningless text (∅).

Original Example	« **Every great** romantic comedy **needs** conflict between the romantic leads [...] **This story** falls completely flat in this **area** [...] suspense is flat, **there** is no anticipation, **and** there **really is no** allure [...] I was quite surprised. During the movie, I **expected them** more **to** play **a** game of checkers and chat **about** the weather than see **any** moving passion [...] While I'm a fan **of** both actors [...] The writing **was** very **weak, which** also **might** have **impacted** the **performances** [...] »	Label : Negative sentiment GRU : Negative sentiment (0.09) GRU + CP : Positive sentiment
Noisy Example	« **ambidexterous trentini** romantic comedy **dispassionately** conflict between the romantic leads [...] **fanout phoolan** falls completely flat in this **centerpiece** [...] suspense is flat, **binding** is no anticipation, **scroller** there **wiedzmin laudable thunderball** allure [...] I was quite surprised. During the movie, I **lives' 'are** more **subtextual** play **commemorations** game of checkers and chat **stratovarius** the weather than see **linking** moving passion [...] While I'm a fan **unti** both actors [...] The writing **ripoff's** very **nare releases** also **sharikov** have **maes** the **'sketching'** [...] »	Label : Negative sentiment , GRU : Positive sentiment (0.74) GRU + CP : {Positive sentiment , Negative sentiment }
Outlier Example	« wolverines 'sandwich' controversial posit homme subfunctions snowmobile symbiotic malamud challenge needle's personl witch's nonce wills' swooshes cobbled brash mcq wanky 'bought regenerated southstreet amazed ravenna 'mainly belyt hijixn shrugs deodorant mesquida anodynesprech romishness malice seldomely settling dispicable vocation [...] reduce macfarlane's disclosing officers' wiretapping balbao seagals mi3 dibnah romulan controls dolled maguire' [...] »	GRU : Positive sentiment (0.99) GRU + CP : ∅

Figure 3.13: Examples of noisy and outlier texts for IMDb.

EGSS Figure 3.14 visualizes through a Principal Component Analysis a comparison of the positions of the test examples on the density regions before (a) and after (b) the addition of a Gaussian noise. This shows that several examples are positioned outside the density regions after the introduction of the disturbances. The outlier examples (c) created by modifying some characteristics of these test examples with extreme values (to simulate a sensor failure, for instance) are even further away from the density regions, and recognized as such by the conformal classifier (∅).



(a) real **(b)** noisy **(c)** outlier

Figure 3.14: Density visualization of noisy and outlier examples for EGSS.

3.6 CONCLUSION

In this chapter, we structured some main concepts of conformal prediction and showed via an application how it can be used to have more reliable and cautious models for different data types (image, text, tabular) by detecting ambiguous examples and handling noisy and outlier ones.

In the next chapter, we will focus on applying conformal prediction to a more complex learning problem, which is multi-target regression.

Chapter 4

Conformal prediction for multi-target regression

> *Man's mind once stretched by a new idea, never regains its original dimension.*
>
> **— Oliver Wendell Holmes**

Contents

The most common supervised task in Machine Learning is to learn a single-output prediction model. However, this setting can be generalized to multi-task learning [Caruana, 1998], where the objective is to predict multiple outputs from the input features characterizing the data set. For instance, multi-label classification focuses on having numerous binary outputs [Zhang et al., 2013], label-ranking algorithms can be seen as problems with ordinal target values [Vembu et al., 2010], and multi-target regression considers real output values [Borchani et al., 2015]. A multi-output learning algorithm can be improved by providing an estimate of the confidence to be placed in its predictions. This can be a great added value when it comes to real-world applications with scarce data (for instance in the medical domain where examples are very hard to collect for specific targets, and where predictions are used for critical decisions) or with multiple, possibly correlated output variables to predict at once. It is then natural to try to leverage such correlations to improve predictions.

Most research work on conformal prediction for multi-task learning focuses on the problem of multi-label classification [Wang et al., 2015; Wang et al., 2020], where each task is a binary classification one. Conformal prediction for multi-target regression has been less explored, even though it can be quite useful in practice. To our knowledge, only a few studies deal with conformal prediction for multi-target regression: [Kuleshov et al., 2018] provide a theoretical framework to use conformal predictors within manifold (e.g., to provide a unidimensional embedding of the multivariate output), while [Neeven et al., 2018] use a straightforward multi-target extension of a conformal single-output k-nearest neighbor regressor [Papadopoulos et al., 2011a] to provide weather forecasts. However, this latter essentially verifies validity (i.e., having well-calibrated outputs) for each individual target.

In this chapter, we go through this largely unexplored area by extending single-output regression methods in conformal prediction to the multi-target problem, and come up with other non-conformity measures that take into consideration the correlation between the outputs.

The first section will briefly present the multi-target regression setting, with details about data and metrics that we take into account for our conformal approaches. The second section will focus on a naive conformal approach that treats targets independently (as in our paper [Messoudi et al., 2020a]). The third section will exploit copulas to consider the dependence structure between the targets in the conformal approach (which is the main contribution of our paper [Messoudi et al., 2021c]). The last section will concentrate on an ellipsoidal approach that gives a more flexible conformal region (which is thoroughly explained in our paper [Messoudi

et al., 2022a]). Thus, most of the content of this chapter will be taken from these cited papers.

4.1 MULTI-TARGET REGRESSION (MTR)

In this section, we will introduce the multi-target regression setting and the considered metrics used to evaluate our methods. We will also present the synthetic and real-world data sets on which our experiments are conducted.

4.1.1 *An overview on MTR*

Let us consider a multi-target regression problem. As seen before, we have the same setting of $z_1 = (x_1, y_1), z_2 = (x_2, y_2), \ldots, z_n = (x_n, y_n)$ successive examples drawn from a distribution on the example space $\mathbf{Z} := \mathbf{X} \times \mathbf{Y}$, with $x_i \in \mathbf{X}$ an object in the vector space of objects $\mathbf{X}$ and $y_i = (y_i^1, \ldots, y_i^m) \in \mathbf{Y}$ its target, where, in this case, the target space $\mathbf{Y} = \{y^1, \ldots, y^m\} \subset \mathbb{R}^m$ is made of m real-valued targets (we will use superscripts to denote the dimensions, and subscripts to denote sample indices). This simply means that the objective of multi-target regression is to predict multiple outputs based on the input features characterizing the data set, which generalizes standard regression. There are two distinct approaches to treat MTR called *algorithm adaptation* and *problem transformation* methods.

For *algorithm adaptation*, standard single-output regression algorithms are extended to the multi-target regression problem. Many models were adapted to the MTR problem, such as Support Vector Regressors [Sánchez-Fernández et al., 2004], regression trees [De'Ath, 2002], kernel methods [Baldassarre et al., 2012] and rule ensembles [Aho et al., 2009].

In *problem transformation*, one usually adapts multi-target regression data sets to the single-target regression task, building a model for each output and then regrouping all the predictions. Many renowned multi-label classification problem transformation methods were extended to the multi-target regression case by [Spyromitros-Xioufis et al., 2012], such as:

- **Binary relevance** [Li et al., 2003]: creates m data sets, one for each label. One binary classifier is then trained on each data set independently, and the prediction for a new example is the union of all the m predictions of all single-label classifiers.
- **Classifier chains** [Read et al., 2011]: decides on an order for all labels, and then each binary classifier for each label adds all previous classifier predictions as additional features.

As our goal here is not to produce a new MTR method, but rather to propose a flexible means to make the predictions reliable through

conformal inference, we will not make a more detailed review of those methods. The reader interested in different methods can consult for instance [Spyromitros-Xioufis et al., 2016].

4.1.2 *Evaluation of conformal MTR*

As seen in subsection 3.2, performance metrics usually considered for conformal prediction are validity and efficiency. These same metrics will be employed to evaluate our methods by adapting them to the MTR setting. In this case, we will define a *global significance level* ϵ_g, which is the error rate of the m-dimensional confidence region generated by conformal prediction. More details about how to define ϵ_g will be given for each method in the upcoming sections.

Validity is computed by calculating the accuracy of each nonconformity measure (NCM) and comparing it with the calibration line. This line represents the case where the error rate is exactly equal to ϵ_g for a confidence level $1-\epsilon_g$, which is the desired outcome of using conformal prediction. In multi-target regression, the accuracy is computed based on whether the observation y of object x belongs to the conformal prediction region $[\Gamma^{\epsilon_g}(x)]$ or not depending on the chosen significance level ϵ_g.

Concretely, for each considered confidence level ϵ_g and new object x_{n+1} in the test set X^{ts}, we obtain a prediction $[\Gamma^{\epsilon_g}(x_{n+1})]$. From this, we can compute the empirical validity as the percentage of times that $[\Gamma^{\epsilon_g}(x_{n+1})]$ contains the true observed value y_{n+1}, i.e.

$$Val\left([\Gamma^{\epsilon_g}(x_{n+1})]\right) = \frac{\sum_{(x_{n+1},y_{n+1})\in Z^{ts}} \mathbb{1}_{y_{n+1}\in[\Gamma^{\epsilon_g}(x_{n+1})]}}{|Z^{ts}|}.$$

Doing it for several values of ϵ_g, we obtain a calibration curve that should be as close as possible to the identity function in order for the conformal approach to be exactly valid. This enables us to introduce another metric, *area of validity*, which provides the average difference between a perfect calibration (the identity function) and the observed curve. This means, in particular, that a negative value indicates that the observed frequency is in average below the specified confidence degree.

Efficiency in single-output regression is measured by the size of the intervals, and a method is all the more efficient as predicted intervals are small. To assess efficiency in multi-target regression, we can simply compute the volume of the obtained predictions. For each experiment, we then compute the median value of those volumes (for the estimation to be robust against very large conformal prediction regions), thus we have:

$$Eff([\Gamma^{\epsilon_g}(x_{n+1})]) = Vol([\Gamma^{\epsilon_g}(x_{n+1})]).$$

Since the volume of an area depends on its shape, efficiency will be defined differently for each conformal approach in its own section.

4.1.3 *Data description*

For comparison purposes, we chose to apply our methods on the same synthetic and real data sets. Thus, it is essential to present them before giving more details on our conformal approaches to treat multi-target regression.

On synthetic data

The synthetic data set aims at assessing the performance of all our methods by controlling the dependence structure between the outputs. It contains 50000 instances and is created as follows:

- x is 2-dimensional with x^1 and x^2 generated independently using a uniform distribution with values between -5 and 5.
- y is a 2-dimensional linear transformation of x plus some Gaussian centered noise:

$$y = Ax + \varepsilon \quad \text{with } A = \begin{bmatrix} 0.7 & 0.3 \\ 0.2 & 0.8 \end{bmatrix} \text{ and } \varepsilon \sim \mathcal{N}(0, S_x).$$

The covariance matrix S_x of ε depends on x and is a weighted average of four covariance matrices at four different points μ_i:

$$S_x = \sum_{i=1}^{4} \frac{\Delta(x, \mu_i)}{\sum_{j=1}^{4} \Delta(x, \mu_j)} \times \text{Cov}_{\mu_i} \tag{4.1}$$

where Δ is an inverse distance calculated as follows:

$$\Delta(x, \mu_i) = \left(\frac{1}{d(x, \mu_i) + \xi} \right)^4 \tag{4.2}$$

with ξ a small value used to avoid a division by 0. Figure 4.1 shows the four μ_i points that are placed at the extremities of our distribution with the following coordinates:

$$\mu_1 = (-5, 5), \quad \mu_2 = (5, 5), \quad \mu_3 = (-5, -5), \quad \mu_4 = (5, -5)$$

These μ_i points have the distinct covariance matrices Cov_{μ_i}:

$$\text{Cov}_{\mu_1} = \text{Cov}_{\mu_4} = \begin{bmatrix} 0.1 & -0.09 \\ -0.09 & 0.1 \end{bmatrix}, \quad \text{Cov}_{\mu_2} = \text{Cov}_{\mu_3} = \begin{bmatrix} 0.1 & 0.09 \\ 0.09 & 0.1 \end{bmatrix}$$

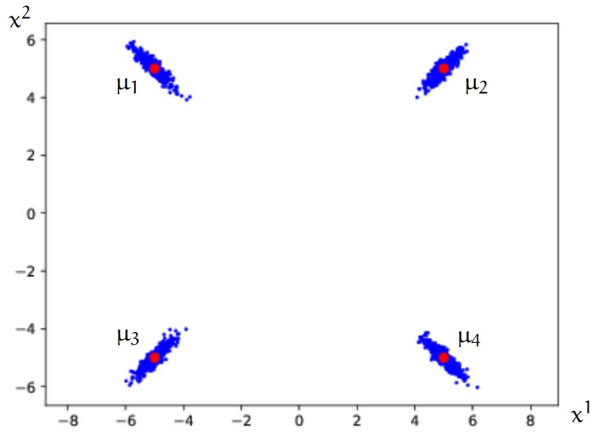


Figure 4.1: Representation of μ_i points and their covariance matrices.

On real data

We also used 19 data sets with various numbers of targets to compare between the different conformal approaches. These data sets are from Mulan and the UCI repository. Their information is summed up in Table 4.1.

Names	**Instances**	**Features**	**Targets**	**Source**
res building	372	105	2	[Rafiei et al., 2016]
enb	768	8	2	[Tsoumakas et al., 2011]
music origin	1059	68	2	[Zhou et al., 2014]
bias corr	7750	25	2	[Cho et al., 2020]
jura	359	15	3	[Tsoumakas et al., 2011]
scpf	1137	23	3	[Tsoumakas et al., 2011]
indoorloc	21049	520	3	[Torres-Sospedra et al., 2014]
sgemm	241600	14	4	[Nugteren et al., 2015]
atp1d	337	411	6	[Tsoumakas et al., 2011]
atp7d	296	411	6	[Tsoumakas et al., 2011]
rf1	9125	64	8	[Tsoumakas et al., 2011]
rf2	9125	576	8	[Tsoumakas et al., 2011]
osales	639	413	12	[Tsoumakas et al., 2011]
wq	1060	16	14	[Tsoumakas et al., 2011]
scm1d	9803	280	16	[Tsoumakas et al., 2011]
scm20d	8966	61	16	[Tsoumakas et al., 2011]
oes10	403	298	16	[Tsoumakas et al., 2011]
oes97	334	263	16	[Tsoumakas et al., 2011]
com crime	2215	125	18	[Redmond, 2011]

Table 4.1: Information on the used multi-target regression data sets.

After presenting the multi-target regression setting and the different data sets used in our experiments, we will now detail how conformal prediction and multi-target regression can be combined, starting with the naive conformal MTR approach..

4.2 NAIVE CONFORMAL MTR

Within the MTR setting, instead of having one single real-valued output, we have a multi-dimensional output $\mathbf{Y} = \{y^1, \ldots, y^m\}$ with $y^j \in \mathbb{R}, j \in \{1, \ldots, m\}$ the different individual real-valued m targets. In this first approach, we make target-wise predictions. This implies that for each individual target, a conformal inference approach will need to produce an individual interval prediction. Thus, given a new object x_{n+1} and a *global significance level* ϵ_g, we can define a *hyper-rectangle* $[\Gamma^{\epsilon_g}(x_{n+1})]$ as:

$$[\Gamma^{\epsilon_g}(x_{n+1})] = \times_{j=1}^{m} [\underline{\hat{y}}^j_{n+1}, \overline{\hat{y}}^j_{n+1}]. \tag{4.3}$$

where $\times$ is a Cartesian product, m is the number of targets and $\underline{\hat{y}}^j_{n+1}, \overline{\hat{y}}^j_{n+1}$ are respectively the lower and upper bounds of the interval predictions given by the non-conformity measure for each target in $\mathbf{Y}$. This hyper-rectangle forms the volume:

$$Vol([\Gamma^{\epsilon_g}(x_{n+1})]) = \prod_{j=1}^{m} (\overline{\hat{y}}^j_{n+1} - \underline{\hat{y}}^j_{n+1}).$$

Thus, validity for a conformal approach based on a hyper-rectangle will be calculated based on the times where a global ground-truth y_{n+1} of a new object x_{n+1} belongs to this volume, i.e. each single ground-truth y^j_{n+1} for each individual target y^j should be between the bounds $\underline{\hat{y}}^j_{n+1}, \overline{\hat{y}}^j_{n+1}$ of its interval prediction. Efficiency will consist on computing this volume that should be as tight as possible. Figure 4.2 illustrates this hyper-rectangle for a 2-dimensional multi-target regression problem.

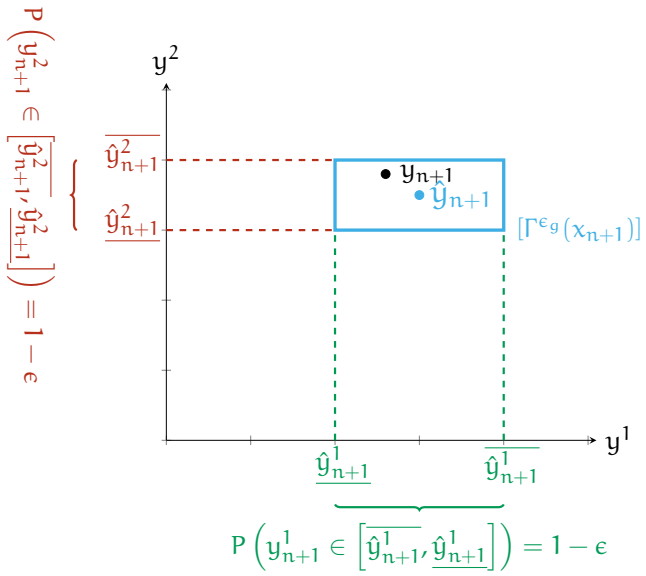


Figure 4.2: Illustration of a hyper-rectangle conformal region.

With this view, the objective of conformal inference for MTR is to satisfy the global significance level ϵ_g required by the user such that:

$$P(y_{n+1} \in [\Gamma^{\epsilon_g}(x_{n+1})]) \geqslant 1 - \epsilon_g. \tag{4.4}$$

This probability can also be expressed using the individual interval predictions for each target Y^j as follows:

$$P(y^1_{n+1} \in [\underline{y^1_{n+1}}, \overline{y^1_{n+1}}], \ldots, y^m_{n+1} \in [\underline{y^m_{n+1}}, \overline{y^m_{n+1}}]) \geqslant 1 - \epsilon_g. \tag{4.5}$$

In this case, ϵ_g comes from choosing the corresponding individual significance levels $\epsilon^1, \ldots, \epsilon^m$ that allow us to obtain the sought-after confidence level $1 - \epsilon_g$ for the whole hyper-rectangle in order to verify that a correctly predicted example must have all of its m observed values y^j in the corresponding interval predictions for each target.

To do so, we propose a naive approach that considers all individual significance levels as equal, i.e. $\epsilon^1 = \ldots = \epsilon^m = \epsilon_t$ and that all targets are independent. This means that the actual confidence level of the hyper-rectangle is estimated as:

$$1 - \epsilon_g = \prod_{j=1}^{m}(1 - \epsilon^j) = (1 - \epsilon_t)^m \tag{4.6}$$

This can be linked to the simple and conservative approach of Bonferroni correction [Weisstein, 2004] in a multiple-comparison correction case.

To adapt the conformal prediction framework to the multi-target regression problem, we can calculate the corrected value of ϵ_t that should be used in order to obtain a global ϵ_g, so as to get an overall confidence level of the hyper-rectangle $1 - \epsilon_g$. This corrected ϵ_t is defined as follows:

$$\epsilon_t = 1 - \sqrt[m]{1 - \epsilon_g} \approx \frac{\epsilon_g}{m} \quad \text{when } \epsilon_g \text{ is small.} \tag{4.7}$$

Our experiments focus on this corrected individual significance level.

Apart from this assumption, we use the possible links between y^j when proposing other extensions to existing NCMs following two main ideas. The first one is to learn the normalizing coefficients by a multi-target model. Thus, this latter will exploit the information coming from training each task and will share representations between related targets in order to generalize better and give greater performance results (see [Ruder, 2017] and [Caruana, 1993]). The second idea is to use a deep fixed-length representation of the data that was learned from trying to predict all the targets at once when calculating the normalizing coefficient in k nearest neighbors based non-conformity measures, thus embedding the correlation information in the deep network layers. This is based on the fact that using representations of the examples instead of the raw instances is a good method

to boost the performance of the neural network on tabular data by helping it generalize better (see [Guo et al., 2016] and [De Brébisson et al., 2015]).

4.2.1 *Naive non-conformity measures*

Since we will compute the non-conformity scores on each target individually and will adjust them to fit the MTR case by recalculating the needed ϵ_t, we will express non-conformity measures for a single target y_i. We use three existing normalized non-conformity measures described in Table 3.1 of Chapter 3 in the single-output regression as follows:

- **SINGLE:** uses the normalized non-conformity measure:

$$\alpha_i = \frac{|y_i - \hat{y}_i|}{\exp(\mu_i) + \beta}, \tag{4.8}$$

  where each $\mu_i = \ln(|y_i - \hat{y}_i|)$ is estimated by a single deep neural network trained on each output separately.

- **Original k-NN (O-KNN):** adopts the k-nearest neighbors non-conformity measure:

$$\alpha_i = \left| \frac{y_i - \hat{y}_i}{\exp(\gamma \lambda_i^k)} \right|, \tag{4.9}$$

  based on λ_i^k only, with the distances calculated between the original form of the examples x_i. We recall that:

$$\lambda_i^k = \frac{d_i^k}{median(d_j^k, z_j \in Z^{tr})}, \quad d_i^k = \sum_{j=1}^{k} \delta(x_i, x_{i_j}),$$

  where δ is a distance.

- **Original k-NN with Standard Deviation (OS-KNN):** adopts the k-nearest neighbors non-conformity measure:

$$\alpha_i = \left| \frac{y_i - \hat{y}_i}{\exp(\gamma \lambda_i^k) + \exp(\rho \xi_i^k)} \right|, \tag{4.10}$$

  based on λ_i^k and ξ_i^k, with the distances calculated between the original form of the examples x_i. We recall that:

$$\xi_i^k = \frac{s_i^k}{median(s_j^k, z_j \in Z^{tr})},$$

  where $s_i^k = \sqrt{\frac{1}{k} \sum_{j=1}^{k} (y_{i_j} - \overline{y_{i_{1\ldots k}}})^2}$ and $\overline{y_{i_{1\ldots k}}} = \frac{1}{k} \sum_{j=1}^{k} y_{i_j}$.

To these existing non-conformity measures, we add three new ones defined as:

- **MULTI:** trains a single deep neural network to estimate the normalizing coefficients μ_i in (4.8) for all outputs at the same time.
- **Representation k-NN (R-KNN):** instead of using the original form of the data, it employs the learned deep representations of the examples extracted from the before last dense layer of the underlying algorithm's neural network to compute d_i^k in λ_i^k for the k-nearest neighbors non-conformity measure (4.9).
- **Representation k-NN with Standard Deviation (RS-KNN):** uses the k-nearest neighbors non-conformity measure (4.10) with λ_i^k and ξ_i^k where the learned deep representations of the examples are used to calculate d_i^k.

4.2.2 *Experimental setting*

Since Transductive Conformal Prediction (Section 3.1) is not computationally efficient when using deep learning architectures, we use the Inductive Conformal Prediction (ICP) framework in order to only train the underlying deep neural network model once. Hence, we follow the steps in Figure 4.3:

1. Split the data set into proper training $\mathbf{Z}^{tr}$, calibration $\mathbf{Z}^{cal}$ and test $\mathbf{Z}^{ts}$.
2. Use $\mathbf{Z}^{tr}$ to train the underlying algorithm, which is a deep neural network, and get the output predictions and the representations of each example.
3. For each non-conformity measure, learn the appropriate normalizing algorithm (deep neural network for SINGLE and MULTI NCMs, and kNN on the original examples for O-KNN and OS-KNN NCMs and their representations for R-KNN and RS-KNN NCMs) using $\mathbf{Z}^{tr}$.
4. After choosing ϵ_g, calculate the corrected ϵ_t as in Equation (4.7) and compute the individual non-conformity scores for all targets for each NCM by using $\mathbf{Z}^{cal}$ in order to obtain their individual α_s^j for $j \in \{1, \ldots, m\}$.
5. For each new object x_{n+1} in $\mathbf{Z}^{ts}$, predict $\hat{y}$ and its representation using the underlying neural network, and compute all of its individual interval predictions to form the hyper-rectangle depending on the chosen significance level ϵ_g for each NCM.

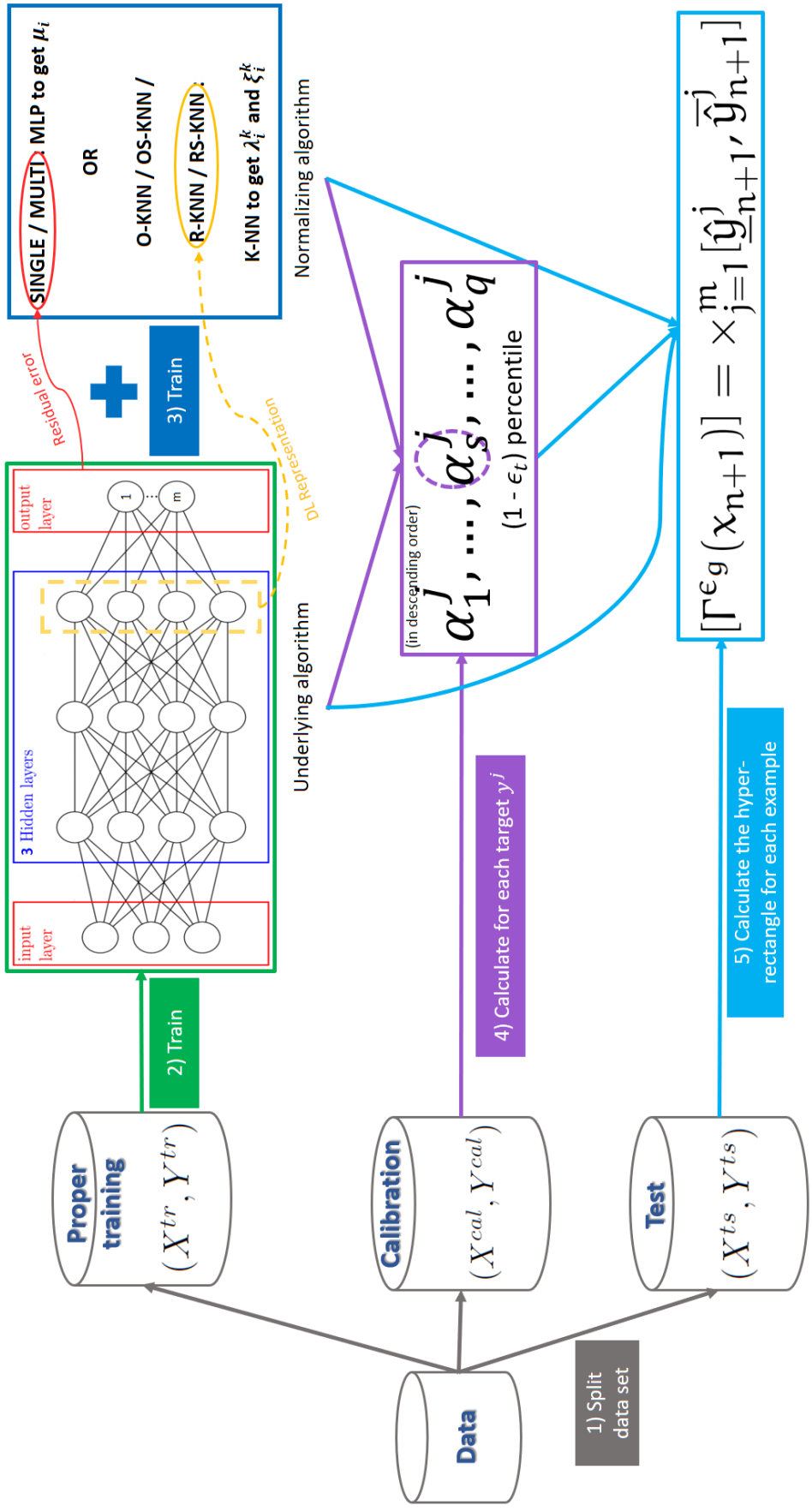


Figure 4.3: Naive MTR conformal prediction approach.

Since we are working with neural networks, we normalize all the features and targets to have a mean of 0 and a standard deviation of 1 as a pre-processing step in order to make the deep neural network optimization easier. Then, we conduct all our experiments with 10-folds cross validation, meaning that each data set is split into 10 equally-sized folds and the experiments are repeated for each k fold as the test set and the remaining $k-1$ sets as the training set. This procedure is necessary in order to eliminate biased results caused by a specific split of the data or the examples chosen in the calibration set. The results are thus averaged on all 10 folds.

The overall focus of this naive approach is to compare between the different non-conformity measures presented above. Thus, for all experiments on all data sets, we keep the same amount of examples in the calibration set, which is 10% of the training examples. We also do not optimize the sensitivity parameters for each data set and use the same values, which are $\beta = 0.1$ for SINGLE and MULTI NCMs, and $\gamma = \rho = 0.5$ for the remaining NCMs. Moreover, we keep the same underlying Machine Learning algorithm for all experiments, which is a deep neural network. Its architecture is as follows:

- Apply a dense layer to the input (with the number of the continuous features and the number of one hot values for the categorical features), with "selu" activation (scaled exponential linear units [Klambauer et al., 2017]).
- Add three other hidden dense layers with dropouts and "selu" activation.
- Add a dense layer with "selu" activation and use it as a feature extractor to produce a representation vector with a fixed size.
- Add a final dense layer with m neurons representing the targets and a linear activation to get the outputs predicted by the model.

The results of this deep neural architecture will enable us to calculate values of the normalizing coefficients for the corresponding non-conformity measure:

- μ_i: estimate the error $\ln(|y_i - \hat{y}_i|)$ which will be learned by the normalizing neural network. This Multi-Layer Perceptron (MLP) is composed of three hidden dense layers with "selu" activation and dropouts and a final dense layer with one output for each target separately in the case of the SINGLE non-conformity measure, or with m neurons for all targets at once in the case of the MULTI non-conformity measure.
- λ_i^k and ξ_i^k: use the deep representations of each example to compute d_i^k for the k-nearest neighbors based non conformity measures R-KNN and RS-KNN.

4.2.3 *Results on synthetic data*

Using the synthetic data set described above, we computed the mean validity and efficiency (surface) of all non-conformity measures for different values of ϵ_g. Results in Table 4.2 show that they all have similar performances when it comes to validity, i.e. they are valid, but become a little bit over-conservative as ϵ_g grows. For efficiency, the MULTI NCM slightly outperforms the others, giving the tighter conformal prediction regions.

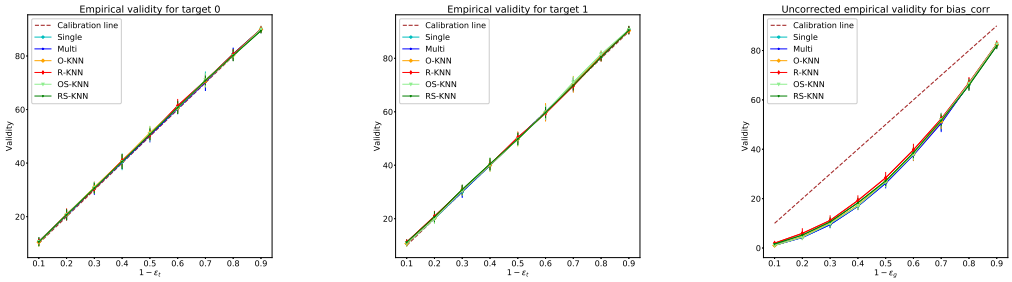
synthetic (k = 2)		$\epsilon_g = 0.01$	$\epsilon_g = 0.05$	$\epsilon_g = 0.1$	$\epsilon_g = 0.15$	$\epsilon_g = 0.2$
Validity	SINGLE	99.14 ± 0.09	95.60 ± 0.20	91.29 ± 0.37	86.71 ± 0.46	82.37 ± 0.65
	MULTI	99.16 ± 0.08	95.51 ± 0.24	91.20 ± 0.32	86.79 ± 0.41	82.39 ± 0.56
	O-KNN	99.15 ± 0.10	95.65 ± 0.19	91.23 ± 0.41	86.81 ± 0.44	82.43 ± 0.60
	R-KNN	99.17 ± 0.08	95.62 ± 0.28	91.34 ± 0.28	87.07 ± 0.51	82.62 ± 0.63
	OS-KNN	99.16 ± 0.08	95.69 ± 0.18	91.29 ± 0.38	86.78 ± 0.49	82.52 ± 0.46
	RS-KNN	99.18 ± 0.08	95.71 ± 0.17	91.31 ± 0.34	86.91 ± 0.52	82.60 ± 0.51
Efficiency	SINGLE	3.60 ± 0.12	2.27 ± 0.09	1.72 ± 0.07	1.41 ± 0.06	1.19 ± 0.05
	MULTI	$\mathbf{3.59 \pm 0.12}$	$\mathbf{2.24 \pm 0.08}$	$\mathbf{1.71 \pm 0.06}$	$\mathbf{1.40 \pm 0.05}$	$\mathbf{1.18 \pm 0.04}$
	O-KNN	3.84 ± 0.19	2.35 ± 0.11	1.78 ± 0.09	1.44 ± 0.07	1.21 ± 0.06
	R-KNN	4.12 ± 0.22	2.43 ± 0.14	1.80 ± 0.09	1.45 ± 0.07	1.20 ± 0.07
	OS-KNN	3.80 ± 0.15	2.35 ± 0.11	1.77 ± 0.08	1.44 ± 0.07	1.21 ± 0.06
	RS-KNN	3.86 ± 0.20	2.36 ± 0.12	1.76 ± 0.08	1.43 ± 0.07	1.20 ± 0.06

Table 4.2: Validity and efficiency results for synthetic data with naive conformal MTR.

4.2.4 *Results on real data*

To motivate the choice of using a corrected naive conformal MTR approach, we compute the empirical validity for each target of the "bias corr" data set, as well as its uncorrected hyper-rectangle validity (i.e. by choosing ϵ_t equal to the desired hyper-rectangle's significance level ϵ_g) in Figure 4.4. The results show that for each target, all NCM lines are in agreement, or slightly above the calibration line. However, when computing the validity for the hyper-rectangle, the performance of the conformal predictors for multi-target regression (computed without our naive correction) is less than the calibration line, due to the observation made earlier. This proves the utility of using our corrected naive approach to compute the actual values of ϵ_t and obtain a $1 - \epsilon_g$ confidence level on all the hyper-rectangle.

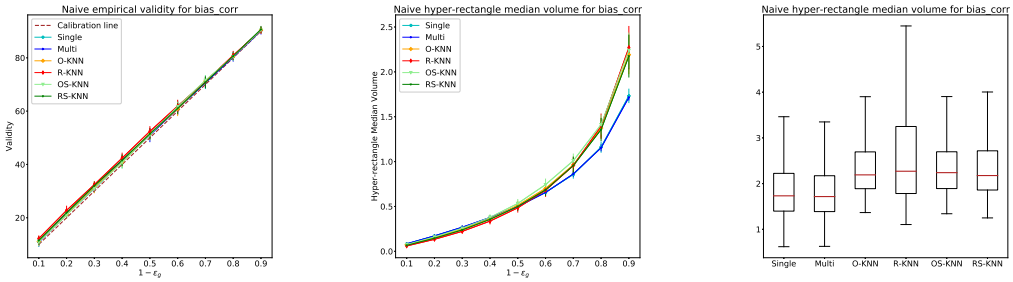
Results of the corrected validity and efficiency using our naive conformal MTR approach are summarized in Tables 4.3 and 4.4 for $\epsilon_g = 0.1$. Other results in a figure form for more ϵ_g values are shown in Figures 4.5 and 4.6 for "bias corr" and "sgemm" data sets. Figures for the remaining data sets can be found in Appendix A. Note that for data sets with more than four targets, we use a logarithmic scale to plot the median volume,



(a) Target 0 **(b)** Target 1 **(c)** Uncorrected hyper-rectangle

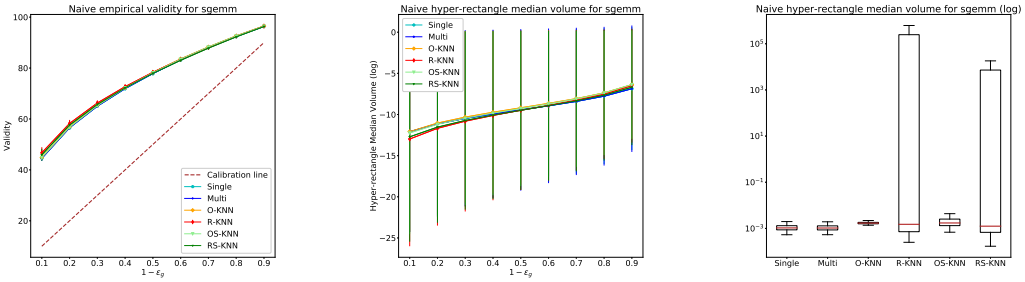
Figure 4.4: Uncorrected empirical validity results per target for "bias corr".

as hyper-rectangle volumes quickly decrease when lowering the required confidence.



(a) Empirical validity **(b)** Empirical efficiency **(c)** Volume for $\epsilon_g = 0.1$

Figure 4.5: Naive conformal results for "bias corr".



(a) Empirical validity **(b)** Empirical efficiency **(c)** Volume for $\epsilon_g = 0.1$

Figure 4.6: Naive conformal results for "sgemm".

In the case of validity, results are summarized in Table 4.3 and shown in sub-figures (a), i.e. sub-figures 4.5a, 4.6a, A.1a, A.2a, A.3a, A.4a, A.5a, A.6a, A.7a, A.8a, A.9a, A.10a, A.11a, A.12a, A.13a.

Data set	SINGLE	MULTI	O-KNN	R-KNN	OS-KNN	RS-KNN
res building	88.97 ± 5.48	88.67 ± 6.51	90.85 ± 3.86	90.86 ± 5.15	91.39 ± 4.50	93.02 ± 5.15
enb	91.55 ± 4.62	91.54 ± 3.04	89.85 ± 3.48	91.02 ± 3.22	90.11 ± 3.86	90.76 ± 4.12
music origin	92.17 ± 2.83	90.47 ± 2.57	89.62 ± 4.32	90.47 ± 4.01	89.90 ± 4.07	89.81 ± 5.02
bias corr	90.33 ± 0.98	90.19 ± 1.43	90.41 ± 1.22	90.31 ± 1.33	90.37 ± 1.17	90.65 ± 1.31
jura	91.67 ± 4.48	90.82 ± 5.55	91.09 ± 4.61	91.66 ± 4.96	91.93 ± 3.60	89.13 ± 5.62
scpf	92.88 ± 3.15	92.26 ± 2.82	91.20 ± 3.76	92.61 ± 3.71	92.61 ± 3.76	91.72 ± 3.59
indoorloc	90.94 ± 0.53	91.08 ± 0.73	91.70 ± 0.86	91.61 ± 0.98	90.74 ± 0.97	90.56 ± 0.65
sgemm	96.34 ± 0.13	96.40 ± 0.17	96.53 ± 0.19	96.48 ± 0.18	96.50 ± 0.18	96.32 ± 0.19
rf1	92.38 ± 0.87	92.55 ± 0.95	91.88 ± 1.35	91.69 ± 1.19	91.23 ± 1.29	91.48 ± 1.08
rf2	91.48 ± 1.18	91.44 ± 1.42	91.44 ± 1.12	91.51 ± 1.33	91.06 ± 1.75	91.94 ± 1.54
osales	92.97 ± 4.03	93.59 ± 4.81	95.46 ± 2.26	93.42 ± 3.43	94.37 ± 4.75	92.33 ± 3.09
wq	92.83 ± 2.84	91.89 ± 3.11	93.49 ± 2.88	92.26 ± 2.63	93.58 ± 2.76	92.74 ± 2.98
scm1d	94.25 ± 1.29	93.97 ± 1.15	93.53 ± 1.41	93.81 ± 1.20	92.51 ± 1.06	94.47 ± 1.11
scm20d	94.89 ± 1.18	95.16 ± 0.99	95.57 ± 0.61	94.91 ± 0.78	93.99 ± 0.84	94.62 ± 1.05
com crime	93.54 ± 2.31	93.68 ± 2.45	94.81 ± 1.71	93.95 ± 1.24	94.76 ± 1.70	94.13 ± 2.32

Table 4.3: Naive conformal empirical validity results for all data sets with $\epsilon_g = 0.1$.
In red are mean values greater than 94% and in orange are mean values between 92% and 94%.

Data set	SINGLE	MULTI	O-KNN	R-KNN	OS-KNN	RS-KNN
res building	19.36 ± 8.39	18.86 ± 10.58	25.19 ± 14.89	18.01 ± 7.46	19.35 ± 9.73	$\mathbf{16.64 \pm 7.66}$
enb	$\mathbf{2.24 \pm 0.74}$	2.27 ± 0.58	2.97 ± 0.54	3.06 ± 0.83	2.86 ± 0.56	2.37 ± 0.49
music origin	21.45 ± 4.92	19.30 ± 5.92	13.94 ± 2.38	15.23 ± 2.93	$\mathbf{13.59 \pm 2.26}$	14.26 ± 2.36
bias corr	1.73 ± 0.08	$\mathbf{1.72 \pm 0.03}$	2.19 ± 0.22	2.27 ± 0.24	2.24 ± 0.19	2.18 ± 0.24
jura	$1.72^{2} \pm 1.26^{2}$	$1.19^{2} \pm 65.51$	$\mathbf{75.80 \pm 43.64}$	$1.15^{2} \pm 68.92$	87.58 ± 66.63	91.95 ± 67.66
scpf	4.98 ± 4.69	4.83 ± 4.99	$\mathbf{0.38 \pm 0.39}$	11.95 ± 19.66	0.51 ± 0.42	5.50 ± 5.04
indoorloc	0.15 ± 0.06	0.16 ± 0.06	0.35 ± 0.17	0.22 ± 0.17	0.20 ± 0.11	$\mathbf{0.14 \pm 0.11}$
sgemm	$1.05^{-3} \pm 5.74^{-4}$	$\mathbf{1.04^{-3} \pm 4.61^{-4}}$	$1.70^{-3} \pm 1.31^{-3}$	$1.51^{-3} \pm 9.87^{-4}$	$1.72^{-3} \pm 1.38^{-3}$	$1.25^{-3} \pm 8.81^{-4}$
rf1	0.06 ± 0.04	$\mathbf{7.02^{-3} \pm 6.09^{-3}}$	1.85 ± 3.22	1.44 ± 2.84	1.48 ± 3.31	1.27 ± 2.99
rf2	$4.99^{-3} \pm 2.94^{-3}$	$\mathbf{3.18^{-3} \pm 1.80^{-3}}$	2.29 ± 3.71	3.82 ± 9.38	2.80 ± 6.28	3.09 ± 8.08
osales	$3.43^{19} \pm 1.01^{20}$	$7.76^{22} \pm 2.33^{23}$	$\mathbf{1.18^{10} \pm 1.65^{10}}$	$3.97^{11} \pm 1.19^{12}$	$5.09^{10} \pm 9.43^{10}$	$1.49^{10} \pm 4.42^{10}$
wq	$1.71^{14} \pm 2.52^{14}$	$8.55^{13} \pm 7.60^{13}$	$6.67^{12} \pm 7.61^{12}$	$1.78^{13} \pm 2.90^{13}$	$\mathbf{2.59^{12} \pm 1.89^{12}}$	$6.43^{12} \pm 5.22^{12}$
scm1d	$6.17^{4} \pm 3.76^{4}$	$4.26^{4} \pm 2.74^{4}$	$5.95^{5} \pm 9.89^{5}$	$8.73^{5} \pm 1.11^{6}$	$\mathbf{2.13^{4} \pm 4.33^{4}}$	$1.21^{5} \pm 1.92^{5}$
scm20d	$3.68^{6} \pm 3.87^{6}$	$5.05^{6} \pm 4.95^{6}$	$2.13^{8} \pm 1.29^{8}$	$1.83^{7} \pm 1.15^{7}$	$7.96^{5} \pm 4.38^{5}$	$\mathbf{6.37^{5} \pm 6.88^{5}}$
com crime	$5.68^{14} \pm 1.69^{15}$	$3.85^{14} \pm 1.15^{15}$	$3.55^{12} \pm 1.03^{13}$	$\mathbf{8.51^{7} \pm 1.17^{8}}$	$1.15^{11} \pm 2.68^{11}$	$1.58^{8} \pm 2.73^{8}$

We note X^{Y} the value $X \times 10^{Y}$.

Table 4.4: Naive conformal empirical efficiency results for all data sets with $\epsilon_g = 0.1$.
Tighter volumes are in bold. Reported results are mean values of medians over all folds.

Validity results prove that using our naive approach to get corrected values of ϵ_t provides valid (in a conservative way) conformal predictions in the case of multi-target regression. All NCMs seem to perform well on all data sets, without any obvious best non-conformity measure for all data sets. However, we notice two distinct cases. The first one has a validity close to the calibration line, as for "bias corr" (sub-figure 4.5a). The second one has a validity that is higher than the confidence level defined by ϵ_g values, as for "sgemm" (sub-figure 4.6a), which indicates that the correction may be too strong. This distinct behavior can also be shown in Table 4.3 when comparing different validity results of all NCMs, especially as the number of targets m grows.

In the case of efficiency, results of the median hyper-rectangle volume are summarized in Table 4.4 and illustrated in sub-figures (b), i.e. sub-figures 4.5b, 4.6b, A.1b, A.2b, A.3b, A.4b, A.5b, A.6b, A.7b, A.8b, A.9b, A.10b, A.11b, A.12b, A.13b.

Efficiency sub-figures show that the overall volume decreases as ϵ_g grows, and significantly faster as the number of targets m increases. This can be explained by the fact that the confidence level becomes smaller, which means that we allow for tighter prediction intervals (as we approach a point prediction), and therefore the conformal regressor tends to give prediction intervals that are smaller and smaller for each target (values less than 1). Then after multiplying these intervals to compute the volume, we find that the hyper-rectangle volume approaches 0 faster when m is large (which justifies the use of a logarithmic scale for data sets with more than four targets).

Table 4.4 shows that, for most data sets, using a MULTI NCM gives predictions with tighter volumes compared to a SINGLE NCM. However, when comparing non-conformity measures based on k-nearest neighbors, we cannot observe a best NCM overall as the results differ from one data set to another.

Since graphs mostly collide and have high standard deviation at times in sub-figures (b), we also drew box plots of median hyper-rectangle volumes in sub-figures (c), i.e. sub-figures 4.5c, 4.6c, A.1c, A.2c, A.3c, A.4c, A.5c, A.6c, A.7c, A.8c, A.9c, A.10c, A.11c, A.12c, A.13c. These confirm all observations previously made. Moreover, they show that using a MULTI NCM reduces the volumes' variability compared to the SINGLE NCM. However, using the deep learning representation for kNN-based NCMs (R-KNN and RS-KNN) does not always give smaller box plots, as it in half of the times adds variability (as for "bias corr" in sub-figure 4.5c).

4.2.5 *Computation time*

During the experiments, we also computed the time taken in seconds for each non-conformity measure to train and predict for each fold on all data

sets. Note that since O-KNN and OS-KNN (respectively R-KNN and RS-KNN) share the same values of parameter λ_i^k, the training of the k-NN is done at the same time for both of them. Thus, the computation time is grouped for both of them. The results of these computation times averaged on 10 folds are shown in the Table 4.5.

Data set	SINGLE	MULTI	O/OS-KNN	R/RS-KNN
res building	18.59	8.63	0.18	0.55
enb	19.18	11.23	0.10	0.24
music origin	39.20	19.78	0.13	0.31
bias corr	224.67	151.02	0.82	3.93
jura	17.65	7.74	0.23	0.66
scpf	30.41	15.88	0.17	0.30
indoorloc	140.64	53.64	33.85	32.19
sgemm	2831.40	729.09	33.89	31.12
rf1	124.46	93.12	1.91	2.15
rf2	274.16	96.56	2.36	2.23
osales	205.04	49.24	0.97	1.57
wq	187.13	49.31	0.70	0.88
scm1d	245.47	22.28	7.13	4.25
scm20d	193.92	23.54	4.06	4.14
com crime	239.56	45.90	1.38	1.51

Table 4.5: Average computation time in seconds for naive conformal NCMs.

From these results, the non-conformity measures based on k-nearest neighbors O/OS-KNN and R/RS-KNN have slightly equal computation time, showing that using the original form of the examples or a deep representation of them does not affect the computation time. However, when comparing SINGLE to MULTI non-conformity measures, we notice that the first one is much slower than the second one, with an increasing difference between them as the number of targets m grows. This is due to the fact that the SINGLE NCM needs to train additional $m-1$ models compared to the MULTI NCM, with each time corresponding to a training done on one single target separately. This shows another advantage of using a MULTI NCM approach instead of a SINGLE one.

4.3 COPULA-BASED CONFORMAL MTR

This section introduces our copula-based approach to improve the naive conformal prediction in the multi-variate regression setting by considering the possible relationships between the outputs. We first recall some basics of copulas and refer to Nelsen [Nelsen, 1999] for a full introduction, before detailing how we apply them to MTR conformal purposes.

4.3.1 *An overview on copulas*

A copula is a mathematical function that can describe the dependence between multiple random variables. The term "copula" was first introduced by Sklar [Sklar, 1959] in his famous theorem, which is one of the fundamentals of copula theory, now known as Sklar's theorem. However, these tools have already been used before, as for instance in Fréchet's paper [Fréchet, 1951] and Höffding's work [Höffding, 1940] [Höffding, 1941] (reprinted as [Höffding, 1994]). Copulas are popular in the statistical and financial fields [Embrechts et al., 2002], but they're nowadays more and more used in other applied scientific domains as well, such as hydrology [Favre et al., 2004], medicine [Nikoloulopoulos et al., 2008], etc.

Let $Y = (Y^1, \dots, Y^m)$ be an m-dimensional vector composed of the random variables $Y^1, \dots, Y^m$. Let its cumulative distribution function (c.d.f.) be $F = F_Y : \mathbb{R}^m \to [0,1]$. This c.d.f. carries two important information:

- The c.d.f. of each random variable Y^j s.t. $F_{Y^j}(y) = P(Y^j \leqslant y)$, $\forall i \in [1, m]$.
- The dependence structure between them.

The objective of copulas is to isolate the dependence structure from the c.d.f.s F_{Y^j} by transforming them into uniform distributed functions U_j and then expressing the dependence structure between U_j. In other words, an m-dimensional copula $C : [0,1]^m \to [0,1]$ is a c.d.f. with standard uniform $U(0,1)$ marginals. It is characterized by the following properties:

1. C is grounded, i.e. if $u_j = 0$ for at least one $j \in \{1, \dots, m\}$, then $C(u_1, \dots, u_m) = 0$.
2. If all components of C are equal to 1 except u_j for all $u_j \in [0,1]$ and $j \in \{1, \dots, m\}$, then $C(1, \dots, 1, u_j, 1, \dots, 1) = u_j$.
3. C is m-increasing, i.e., $\forall a, b \in [0,1]^m$, $a \leqslant b$:

$$\Delta_{(a,b]} C = \sum_{i \in \{0,1\}^m} (-1)^{\sum_{j=1}^m i_j} C(a_1^{i_1} b_1^{1-i_1}, \dots, a_m^{i_m} b_m^{1-i_m}) \geqslant 0. \quad (4.11)$$

The idea of copulas is based on probability and quantile transformation [McNeil et al., 2015] defined in the following proposition:

Proposition 2 *Let* F *be a cumulative distribution function and* $F^{\leftarrow}$ *its generalized inverse, i.e. the function* $F^{\leftarrow}(t) := \inf\{y \in \mathbb{R} : F(y) \geqslant t\}$.

1. ***Quantile transformation.*** *If* $U \sim U(0,1)$ *has a standard uniform distribution, then* $P(F^{\leftarrow}(U) \leqslant y) = F(y)$.
2. ***Probability transformation.*** *If* Y *has a univariate continuous c.d.f.* F, *then* $F_Y \sim U(0,1)$.

Using the above proposition, we can see that all multivariate distribution functions include copulas and that we can use a mixture of univariate marginal distributions and a suitable copula to produce a multivariate distribution function. This is described in Sklar's theorem [Sklar, 1959] as follows:

Theorem 4.3.1 (Sklar's theorem) *For any m-dimensional c.d.f.* F *with marginal distributions* $F_1, \ldots, F_m$, *there exists a copula* $C : [0,1]^m \to [0,1]$ *such that:*

$$F(y_1, ..., y_m) = C(F_1(y_1), \ldots, F_m(y_m)), y \in \mathbb{R}^m. \tag{4.12}$$

If F_j *is continuous for all* $j \in \{1, \ldots, m\}$, *then* C *is unique.*

With the equation 4.12 and $F_j \circ F_j^{\leftarrow}(z) \geqslant z$, we can get:

$$C(u_1, ..., u_m) = F(F_1^{\leftarrow}(u_1), \ldots, F_m^{\leftarrow}(u_m)). \tag{4.13}$$

Fréchet-Höffding bounds

One can show that all dependency structures, i.e., all copulas, lie between the Fréchet-Höffding bounds that correspond to extreme dependencies called comonotonicity and countermonotonicity. In the case of two uniform random variables u_1 and u_2, theses random variables are comonotonic if $u_1 = u_2$ (positive dependency), and countermonotonic if $u_2 = 1 - u_1$ (negative dependency) [Schmidt, 2007].

Theorem 4.3.2 (Fréchet-Höffding bounds) *Let* $M(U) = \min_{1 \leqslant j \leqslant m}\{u_j\}$ *and* $W(U) = \max\{\sum_{j=1}^{m} u_j - m + 1, 0\}$.

1. *For any d-dimensional copula* C, $W(U) \leqslant C(U) \leqslant M(U), U \in [0,1]^m$.
2. W *is a copula if and only if* $m = 2$.
3. M *is a copula for all* $m \geqslant 2$.

Another special instance of dependency that is in between these two bounds is the independence, whose copula is expressed as $\Pi(U) = \prod_{j=1}^{m} u_j$ for $U \in [0,1]^m$.

Archimedean copulas

Archimedean copulas are a special class of copulas that are widely used because they are easy to construct, many copula families belong to this class, and they have a variety of interesting properties (typically explicit, useful in calculations, simple sampling in most of the cases...). The main idea behind Archimedean copulas is based on a generator function ϕ and its pseudo-inverse $\phi^{[-1]}$, defined in [McNeil et al., 2015] as follows:

Definition 1 (Archimedean copula generator) *A continuous, strictly decreasing, convex function $\phi : [0,1] \to [0,\infty]$ satisfying $\phi(1) = 0$ is known as an Archimedean copula generator. It is known as a strict generator if $\phi(0) = \infty$.*

Definition 2 (Pseudo-inverse) *Let $\phi : [0,1] \to [0,\infty]$ be a continuous, strictly decreasing function with $\phi(1) = 0$ and $\phi(0) \leqslant \infty$. We define a pseudo-inverse of ϕ with domain $[0,\infty]$ by:*

$$\phi^{[-1]}(t) = \begin{cases} \phi^{-1}(t) & \text{for} \quad 0 \leqslant t \leqslant \phi(0) \\ 0 & \text{for} \quad \phi(0) \leqslant t \leqslant \infty \end{cases} \tag{4.14}$$

Based on these definitions, the bivariate Archimedean copula is described in the following theorem [Nelsen, 1999]:

Theorem 4.3.3 (Bivariate Archimedean copula) *Let ϕ be a continuous, strictly decreasing function from $[0,1]$ to $[0,\infty]$ such that $\phi(1) = 0$, and let $\phi^{[-1]}$ be the pseudo-inverse of ϕ defined as in 4.14. Then the function $C : [0,1]^2 \to [0,1]$ given by:*

$$C(u,v) = \phi^{[-1]}(\phi(u) + \phi(v)) \tag{4.15}$$

is a copula if and only if ϕ is convex.

In the case of multivariate random variables, the construction of an m-dimensional copula requires that the Archimedean copula generator ϕ should be strict. Then:

$$C(u_1,\ldots,u_m) = \phi^{[-1]}(\phi(u_1) + \ldots + \phi(u_m)) \tag{4.16}$$

gives a copula in any dimension m if and only if the generator inverse $\phi^{-1} : [0,\infty] \to [0,1]$ is completely monotonic [McNeil et al., 2015].

Table 4.6 summarizes characteristics of three one parameter Archimedean copula families [McNeil et al., 2015].

Empirical copulas

Parametric copula approaches, such as choosing within the Archimedean copula families, can have an important weakness when estimating the

Family	Gumbel [Gumbel, 1960]	Clayton [Genest et al., 1993]	Frank [Frank, 1979]
Generator $\phi(t)$	$(-\ln t)^\theta$	$\frac{1}{\theta}(t^{-\theta}-1)$	$-\ln(\frac{e^{-\theta t}-1}{e^{-\theta}-1})$
Parameter θ **range**	$\theta \geqslant 1$	$\theta \geqslant -1$	$\theta \in \mathbb{R}$
Strict	Yes	$\theta \geqslant 0$	Yes
Lower	Π	W	W
Upper	M	M	M

Table 4.6: Archimedean copula families.

marginals and the parameters, especially when strong assumptions about the dependence structure are made. The Empirical copula presents a non-parametrical way of estimating the marginals directly from the observations [Ruschendorf, 1976; Ruymgaart, 1978]. It is defined as follows [Hofert et al., 2019]:

$$C_n(u) = \frac{1}{n}\sum_{i=1}^{n} 1(U_i \leqslant u) = \frac{1}{n}\sum_{i=1}^{n}\prod_{j=1}^{m} 1(U_i^j \leqslant u_j), u \in [0,1]^m. \tag{4.17}$$

where $U_{i,n}$ are the pseudo-observations that replace the unknown marginal distributions, which are defined as:

$$U_i = (U_i^1, \ldots, U_i^m) = (F_1(Y_i^1), \ldots, F_m(Y_i^m)), i \in \{1, \ldots, n\}. \tag{4.18}$$

where F_j for $j \in \{1, \ldots, m\}$ is estimated by:

$$F_j(y) = \frac{1}{n+1}\sum_{i=1}^{n} 1(Y_i^j \leqslant y), y \in \mathbb{R}. \tag{4.19}$$

The idea behind empirical copulas is to make rank transformations of data samples of size n drawn for each dimension d from its corresponding estimated marginal distribution.

4.3.2 *Copula-based non-conformity measures*

As seen before, the objective of the conformal prediction framework for MTR in the normalized setting is to satisfy a global significance level ϵ_g required by the user such that:

$$P(y_{n+1} \in [\Gamma^{\epsilon_g}(x_{n+1})]) \geqslant 1 - \epsilon_g. \tag{4.20}$$

This probability can also be written as follows:

$$P(y_{n+1}^1 \in [\underline{y_{n+1}^1}, \overline{y_{n+1}^1}], \ldots, y_{n+1}^m \in [\underline{y_{n+1}^m}, \overline{y_{n+1}^m}])$$

$$= P\left(\frac{|y^1_{n+1} - \hat{y}^1_{n+1}|}{\sigma^1_{n+1}} \leqslant \alpha^1_s, \ldots, \frac{|y^m_{n+1} - \hat{y}^m_{n+1}|}{\sigma^m_{n+1}} \leqslant \alpha^m_s\right) \geqslant 1 - \epsilon_g. \tag{4.21}$$

Thus, we need to find the individual non-conformity scores $\alpha^1_s, \ldots, \alpha^m_s$, defined for instance by target-wise confidence levels ϵ^j, such that we ensure a global confidence level $1 - \epsilon_g$. Extending

$$P(Q \leqslant \alpha_s) = 1 - \epsilon_g := F_Q(\alpha_s),$$

where F denotes here the joint cumulative distribution induced by P, and considering the random variables $Q^j = |y^j - \hat{y}^j|/\sigma^j$, $j \in \{1, \ldots, m\}$, we get:

$$P(Q^1 \leqslant \alpha^1_s, \ldots, Q^m \leqslant \alpha^m_s) \geqslant 1 - \epsilon_g. \tag{4.22}$$

Should we know the joint distribution in (4.22), and therefore the dependence relations between target predictions, it would be relatively easy to get the individual significance levels[1] ϵ^j associated to the individual non-conformity scores α^j_s such that we satisfy the chosen confidence level $1 - \epsilon_g$. Yet, such a joint distribution is usually unknown. We propose a simple and efficient method to do so, leveraging the connection between (4.22) and copulas. Before doing that, note again that under the assumption that we are well calibrated, we can transform (4.22) into

$$F(\alpha^1_s, \ldots, \alpha^m_s) = 1 - \epsilon_g. \tag{4.23}$$

Considering (4.23), and following Sklar's theorem, we have

$$\begin{aligned} F(\alpha^1_s, \ldots, \alpha^m_s) &= C(F_1(\alpha^1_s), \ldots, F_m(\alpha^m_s)) \\ &= C(1 - \epsilon^1, \ldots, 1 - \epsilon^m) \\ &= 1 - \epsilon_g \end{aligned}$$

where the second line is obtained from (3.16). Clearly, if we knew the copula C, then we could search for values ϵ^j providing the desired global confidence.

A major issue is then to obtain or estimate the copula modelling the dependence structure between the targets and their confidence levels. As copulas are classically estimated from multi-variate observations, a simple means that we will use here is to estimate them from the non-conformity scores generated from the calibration set Z^{cal}. Namely, if α^j_i is the non-conformity score corresponding to the j^{th} target of the z_i example of Z^{cal} for $i \in \{l + 1, \ldots, n\}$, we simply propose to estimate a copula C from the matrix:

[1] Note that there may be multiple choices for such individual levels. Here we will fix them to be equal for simplicity.

$$A = \begin{bmatrix} \alpha_{l+1}^1 & \alpha_{l+1}^2 & \cdots \\ \vdots & \ddots & \\ \alpha_n^1 & & \alpha_n^m \end{bmatrix}. \tag{4.24}$$

On three specific copulas

We will now provide some detail about the copulas we performed experiments on. They have been chosen to go from the one requiring the most assumptions to the one requiring the least assumptions.

THE INDEPENDENT COPULA The Independent copula means that the m targets are considered as being independent, with no relationship between them. It is a strong assumption, but it does not require any estimation of the copula. In this case, (4.22) becomes:

$$\begin{aligned} \Pi(F_1(\alpha_s^1), \ldots, F_m(\alpha_s^m)) &= \prod_{j=1}^{m} F_j(\alpha_s^j) = \prod_{j=1}^{m} P(Q^j \leqslant \alpha_s^j) \\ &\geqslant \prod_{j=1}^{m} (1 - \epsilon^j) = 1 - \epsilon_g, \end{aligned}$$

If we assume that all $\epsilon^1, \ldots, \epsilon^m$ equal the same value ϵ_t, then:

$$\prod_{j=1}^{m} (1 - \epsilon^j) = (1 - \epsilon_t)^m = 1 - \epsilon_g.$$

Thus, we simply obtain (4.7) seen in the naive conformal approach (Section 4.2), which is:

$$\epsilon_t = 1 - \sqrt[m]{1 - \epsilon_g}.$$

This individual significance level ϵ_t is then used to calculate the different non-conformity scores α_s^j for each target in the multi-target regression problem for the Independent copula.

THE GUMBEL COPULA The Gumbel copula is a member of the Archimedean copula family which depends on only one parameter, and in this sense is a good representative of parametric copulas. It comes down to applying the generator function $\phi(F_j(\alpha_s^j)) = (-\ln F_j(\alpha_s^j))^\theta$ and its inverse $\phi^{[-1]}(F_j(\alpha_s^j)) = \exp -(F_j(\alpha_s^j))^{1/\theta}$ to (4.16), resulting in the expression

$$C_G^\theta(F_1(\alpha_s^1), \ldots, F_m(\alpha_s^m)) = \exp - \left(\sum_{j=1}^{m} \left(-\ln F_j(\alpha_s^j) \right)^\theta \right)^{1/\theta}. \tag{4.25}$$

In this case, we need to estimate the parameter θ. Since the marginals $F_j(\alpha^j)$ are unknown, we also need to estimate them. In our case, we will simply use the empirical c.d.f. induced by the non-conformity scores α_i^j of matrix A. An alternative would be to also assume a parametric form of the F_j, but this seems in contradiction with the very spirit of non-conformity scores. In particular, we will denote by $\hat{F}_j$ the empirical cumulative distribution such that

$$\hat{F}_j(\beta) = \frac{|\{\alpha_i^j : \alpha_i^j \leqslant \beta, i \in \{l+1, \dots, n\}\}|}{n-l}, \quad \beta \in \mathbb{R}.$$

The parameter θ can then be estimated from matrix A using for instance the Maximum Pseudo-Likelihood Estimator [Hofert et al., 2019] with a numerical optimization, which can be done by using the Python library "copulae"[2]. Once this is obtained, we then get for a particular choice of ϵ^j that

$$C_G^{\hat{\theta}} = \exp - \left(\sum_{j=1}^{m} \left(-\ln(1-\epsilon^j) \right)^{\hat{\theta}} \right)^{1/\hat{\theta}} \tag{4.26}$$

$$= \exp - \left(\sum_{j=1}^{m} \left(-\ln F_j(\alpha_s^j) \right)^{\hat{\theta}} \right)^{1/\hat{\theta}} \tag{4.27}$$

And we can search for values ϵ^j that will make this equation equal to $1-\epsilon_g$, using the estimations $\hat{F}_j$. The solution is especially easy to obtain analytically if we consider that $\epsilon^1 = \dots = \epsilon^m = \epsilon_t$, as we then have that

$$\epsilon_t = 1 - (1-\epsilon_g)^{1/\sqrt[\theta]{m}},$$

and one can then obtain the corresponding non-conformity scores $\alpha_s^1, \dots, \alpha_s^m$ by replacing F_j by $\hat{F}_j$.

We chose this particular family of Archimedean copulas because its lower bound is the Independent copula (as seen in Table 4.6). We can easily verify this by taking $\hat{\theta} = 1$. Thus, we can capture independence if it is verified, and otherwise search in the direction of positive dependence. One reason for such a choice is that the previous experiments in our naive approach indicate that the product copula gives overly conservative results.

THE EMPIRICAL COPULA Parametric copulas, as all parametric models, have the advantage of requiring less data to be well estimated, while having the possibly important disadvantage that they induce some bias in the estimation, that is likely to grow as the number of target increases. The Empirical copula presents a non-parametric way of estimating the marginals

[2] https://pypi.org/project/copulae/

directly from the observations [Ruschendorf, 1976; Ruymgaart, 1978]. It is defined as follows [Hofert et al., 2019]:

$$C_E(\mathbf{u}) = \frac{1}{n-l}\sum_{i=l+1}^{n} \mathbb{1}_{\mathbf{u}_i \leqslant \mathbf{u}} = \frac{1}{n-l}\sum_{i=l+1}^{n}\prod_{j=1}^{m} \mathbb{1}_{u_i^j \leqslant u^j}, \quad \mathbf{u} \in [0,1]^m, \tag{4.28}$$

where $\mathbb{1}_A$ is the indicator function of event A, and for $i \in \{l+1,\dots,n\}$ the inequalities $\mathbf{u}_i \leqslant \mathbf{u}$ need to be understood component-wise. $\mathbf{u}_i$ are the pseudo-observations that replace the unknown marginal distributions, which are defined as:

$$\mathbf{u}_i = (u_i^1,\dots,u_i^m) = (\hat{F}_1(\alpha_i^1),\dots,\hat{F}_m(\alpha_i^m)), \quad i \in \{l+1,\dots,n\}, \tag{4.29}$$

where distributions $\hat{F}_j$ are defined as before. Simply put, the Empirical copula corresponds to consider as our joint probability the empirical joint cumulative distribution. We then have that

$$C_E(F_1(\alpha_s^1),\dots,F_m(\alpha_s^m)) = \frac{1}{n-l}\sum_{i=l+1}^{n}\prod_{j=1}^{m} \mathbb{1}_{u_i^j \leqslant F_j(\alpha_s^j)}. \tag{4.30}$$

Using that $F_j(\alpha_s^j) = 1-\epsilon^j$, we can then search for values of ϵ^j, $j = 1,\dots,m$ that will make (4.30) equal to $1-\epsilon_g$. Note that in this case, even assuming that $\epsilon^1 = \dots = \epsilon^m = \epsilon_t$ will require an algorithmic search, which is however easy as C_E is an increasing function, meaning that we can use a simple dichotomic search.

4.3.3 *Experimental setting*

We choose to work with a Neural Network (NN) and a Random Forest (RF) as the underlying algorithms, and compare between the three copula functions to show that adding copulas to the non-conformity measures works with any underlying algorithm. However, our approach can be easily adapted to any multi-variate regression model.

To compute the non-conformity scores over the calibration set, we use the MULTI NCM presented in the naive approach, i.e. the normalized non-conformity score given by (4.8) as described in [Papadopoulos et al., 2011b], and predict $\mu_i = \ln(|y_i - \hat{y}_i|)$ simultaneously for all targets by a single multivariate MLP. We chose this NCM among the six previous tested NCMs since all of them nearly performed the same, but with the MULTI NCM being better than the SINGLE one, and giving hyper-rectangle median volumes with low variability.

Experiments are conducted on normalized data with a mean of 0 and a standard deviation of 1, with a 10-fold cross validation to avoid the impact of biased results, and with a calibration set equal to 10% of the training

examples for all data sets. We take the value $\beta = 0.1$ for the sensitivity parameter and do not optimize it when calculating the normalizing coefficient μ_i. We follow the steps in Figure 4.7 as described below:

1. Get the proper training data $\mathbf{Z}^{tr}$, calibration data $\mathbf{Z}^{cal}$ and test data $\mathbf{Z}^{ts}$

2. Train the underlying algorithm (NN or RF) on the proper training data $\mathbf{Z}^{tr}$. The Neural Network's architecture is composed of a first dense layer applied to the input with "selu" activation (scaled exponential linear units [Klambauer et al., 2017]), three hidden dense layers with dropouts and "selu" activation, and a final dense layer with m outputs and a linear activation. The Random Forest is trained for each target alone using Python sklearn's implementation, then each target is predicted independently to get the results. Then, predict $\hat{\mathbf{Y}}^{cal}$ and $\hat{\mathbf{Y}}^{ts}$ for calibration and test data respectively using the underlying algorithm.

3. Train the normalizing Multi-Layer Perceptron on the proper training data $(\mathbf{X}^{tr}, \mu_{tr} = \ln(|\mathbf{Y}^{tr} - \hat{\mathbf{Y}}^{tr}|)$, corresponding to the error estimation of the underlying algorithm. The normalizing MLP consists of three hidden dense layers with "selu" activation and dropouts and a final dense layer with m outputs for predicting all targets simultaneously. This approach was chosen since it proved to be more efficient than a single target approach that we experimented in the naive approach. Then, predict μ_{cal} and μ_{ts} for calibration and test data respectively using the normalizing MLP.

4. If needed, get an estimation[3] of the copula C from the matrix A of calibration non-conformity scores.

5. For a chosen global significance level ϵ_g, get the individual significance level $\epsilon^j = \epsilon_t$ for $j \in \{1, \ldots, m\}$ and calculate $\alpha_s = \{\alpha_s^1, \ldots, \alpha_s^m\}$ for all targets using calibration data, according to the methods mentioned before.

6. Get the interval predictions for each new object x_{n+1} of $\mathbf{Z}^{ts}$ with:

$$[\Gamma^{\epsilon_g}(x_{n+1})] = [\hat{y}_{n+1} - \alpha_s(\exp(\mu_{n+1}) + \beta), \hat{y}_{n+1} + \alpha_s(\exp(\mu_{n+1}) + \beta)]\,. \tag{4.31}$$

Remark 1 *We choose $\epsilon^j = \epsilon_t$ for $j \in \{1, \ldots, m\}$ as we have no indication that individual targets should be treated with different degree of cautiousness. However, since copulas are functions from $[0,1]^m$ to $[0,1]$, there is in principle no problem in considering different confidence degrees for different tasks, if an application calls for it.*

[3] In the case of the Gumbel copula, we use a Maximum Pseudo-Likelihood Estimator with a numerical optimization using the BFGS algorithm

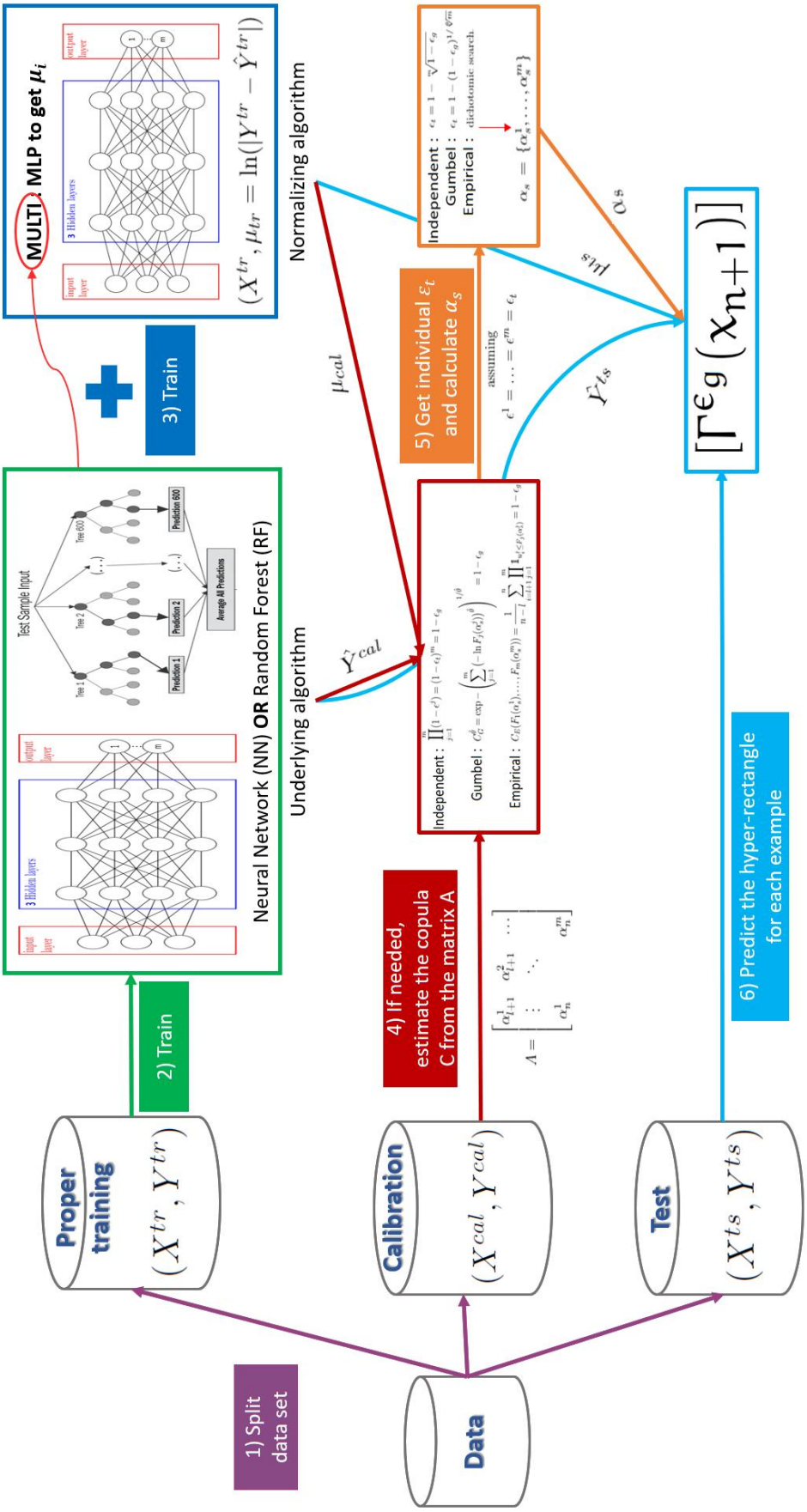


Figure 4.7: Copula-based MTR conformal prediction approach.

The implementation was done using Python and Tensorflow. The copula part of our experiments was based on the book [Hofert et al., 2019] and the Python library "copulae". This work's code is available on GitHub[4].

4.3.4 *Results on synthetic data*

Using the synthetic data set described in section 4.1.3, we calculated the mean validity and efficiency (surface) of all non-conformity measures for different values of ϵ_g. Results in Table 4.7 show that all NCMs have almost the same performance when it comes to validity, i.e. they are valid, but the Independent copula becomes a little bit overly conservative as ϵ_g grows, while both Gumbel and Empirical copulas stay exactly valid. For efficiency, we can clearly see that the RF NCMs give tighter volumes when compared to NN ones, but overall Gumbel and Empirical copula approaches give the best results, with nearly unnoticeable difference between them.

synthetic (k = 2)			$\epsilon_g = 0.01$	$\epsilon_g = 0.05$	$\epsilon_g = 0.1$	$\epsilon_g = 0.15$	$\epsilon_g = 0.2$
Validity	Independent	NN	$\mathbf{99.04 \pm 0.13}$	95.52 ± 0.35	91.14 ± 0.59	86.87 ± 0.69	82.42 ± 0.62
		RF	99.04 ± 0.18	95.70 ± 0.28	91.37 ± 0.46	87.09 ± 0.33	82.69 ± 0.52
	Gumbel	NN	99.16 ± 0.11	94.99 ± 0.26	$\mathbf{89.95 \pm 0.43}$	$\mathbf{85.03 \pm 0.38}$	80.35 ± 0.43
		RF	99.12 ± 0.16	$\mathbf{95.00 \pm 0.29}$	90.07 ± 0.43	84.87 ± 0.42	80.36 ± 0.49
	Empirical	NN	99.16 ± 0.11	95.08 ± 0.41	89.98 ± 0.45	84.91 ± 0.39	$\mathbf{79.96 \pm 0.30}$
		RF	99.12 ± 0.16	95.26 ± 0.32	89.96 ± 0.53	84.87 ± 0.42	79.82 ± 0.47
Efficiency	Independent	NN	3.58 ± 0.27	2.28 ± 0.15	1.72 ± 0.11	1.41 ± 0.09	1.18 ± 0.08
		RF	$\mathbf{3.32 \pm 0.12}$	2.10 ± 0.06	1.58 ± 0.04	1.28 ± 0.03	1.08 ± 0.02
	Gumbel	NN	3.70 ± 0.28	2.18 ± 0.16	1.62 ± 0.11	1.31 ± 0.10	1.10 ± 0.08
		RF	3.41 ± 0.12	$\mathbf{1.98 \pm 0.04}$	$\mathbf{1.47 \pm 0.03}$	$\mathbf{1.17 \pm 0.02}$	0.99 ± 0.01
	Empirical	NN	3.69 ± 0.28	2.20 ± 0.15	1.62 ± 0.12	1.30 ± 0.10	1.08 ± 0.08
		RF	3.41 ± 0.11	2.03 ± 0.05	1.47 ± 0.04	$\mathbf{1.17 \pm 0.02}$	$\mathbf{0.97 \pm 0.02}$

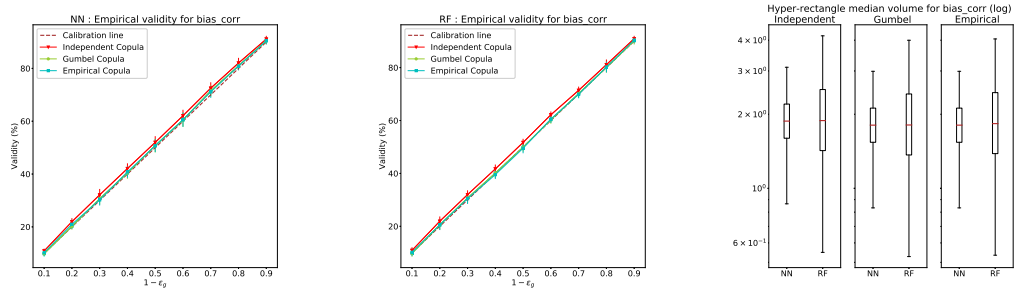
Table 4.7: Validity and efficiency results for synthetic data with copula-based conformal MTR.

4.3.5 *Results on real data*

Analysis on validity and efficiency

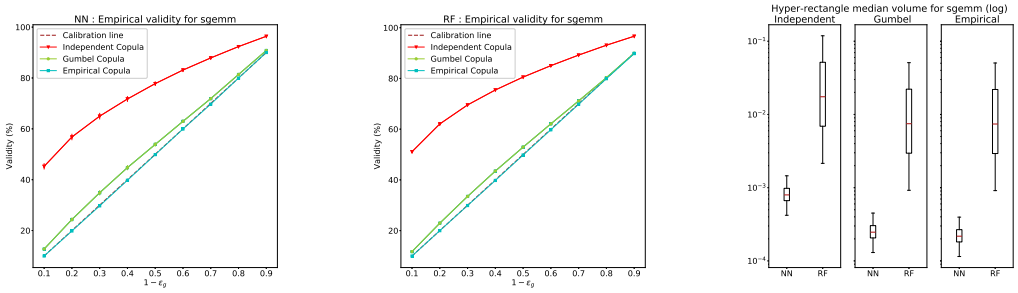
This section presents the results of our experiments, investigating in particular the validity and efficiency of the proposed approaches. Figures 4.8, 4.9 and 4.10 detail these results for "bias corr", "sgemm" and "wq". Complementary results for the remaining data sets can be found in Appendix B. Note that for data sets with more than four targets, we use a logarithmic scale to plot the median volume.

[4] https://github.com/M-Soundouss/CopulaConformalMTR



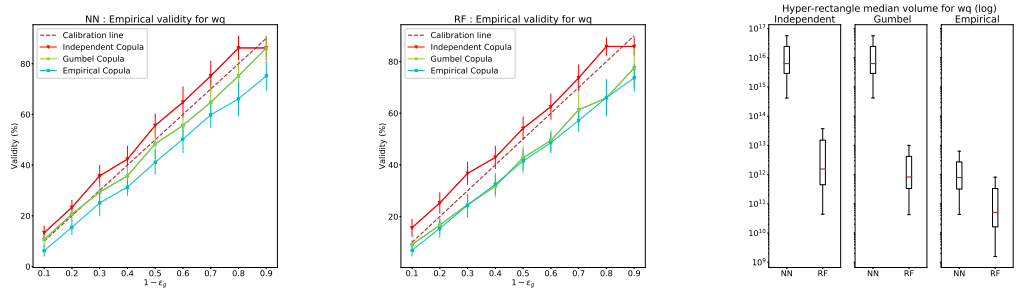
(a) Validity (NN) **(b)** Validity (RF) **(c)** Efficiency (both)

Figure 4.8: Copula conformal results for "bias corr".



(a) Validity (NN) **(b)** Validity (RF) **(c)** Efficiency (both)

Figure 4.9: Copula conformal results for "sgemm".



(a) Validity (NN) **(b)** Validity (RF) **(c)** Efficiency (both)

Figure 4.10: Copula conformal results for "wq".

The results of the error rate or validity curves are shown in sub-figures (a) for the Neural Network (i.e. sub-figures 4.8a, 4.9a, 4.10a, B.1a, B.2a, B.3a, B.4a, B.5a, B.6a, B.7a, B.8a, B.9a, B.10a, B.11a, B.12a) and (b) for the Random Forest (i.e. sub-figures 4.8b, 4.9b, 4.10b, B.1b, B.2b, B.3b, B.4b, B.5b, B.6b, B.7b, B.8b, B.9b, B.10b, B.11b, B.12b) for each data set.

Validity results clearly show that the best performance is obtained by using the Empirical copula, where the model is well calibrated. For most of the studied data sets, the Empirical copula curve is almost perfectly

aligned with the calibration line, and thus almost exactly valid. This is due to the fact that Empirical copula functions use non-parametric estimate of the marginals based on the observations, which enables the model to better adapt to the dependence structure of each data set. This dependence structure is neglected when using an Independent copula-based non-conformity measure, since the m targets are treated as if they were independent, and so the link between them is not exploited when computing ϵ_t. This also means that the difference between the Empirical and the Independent copula-based NCMs is bigger when there is a strong dependence between the non-conformity scores, and is an indication of the strength of this dependence. For instance, we can deduce that the targets are strongly related for "sgemm" by the big gap between the Independent and Empirical curves (Figures 4.9a and 4.9b). For the Gumbel copula, the accuracy curve is generally closer to the calibration line than the one for the Independent copula. This supports the existence of a dependence structure between the targets, since the lower bound of the Gumbel copula is the Independent copula, which means that if the targets were in fact independent, the two curves would perfectly match. This can be seen in Figures 4.8a and 4.8b for "bias corr", where the curves almost overlap all the time, meaning that the targets are likely to be independent.

From the empirical validity results, we also noticed that the Empirical copula NCM can be slightly invalid sometimes (Figures 4.10a and 4.10b for "wq"). We explain this by the fewer number of examples, in which case one could use a more regularized form than the Empirical copula. However, when a lot of examples are available (for instance, more than 200000 observations for "sgemm"), the validity curve of the Empirical copula NCM is perfectly aligned with the calibration line, meaning that this measure is exactly valid (Figures 4.9a and 4.9b).

These conclusions concerning the empirical validity are the same for both underlying algorithms, which suggests that the difference regarding the validity performance mainly comes from the chosen copula-based NCM.

Efficiency results are illustrated in sub-figure (c) for all data sets (i.e. sub-figures 4.8c, 4.9c, 4.10c, B.1c, B.2c, B.3c, B.4c, B.5c, B.6c, B.7c, B.8c, B.9c, B.10c, B.11c, B.12c) for $\epsilon_g = 0.1$.

Efficiency results show that, for each underlying algorithm, the Independent copula NCM has a bigger median hyper-rectangle volume compared to the Gumbel and Empirical copula NCMs, especially in those cases where the existence of a dependence structure is confirmed by the calibration curves as for "sgemm" (Figure 4.9c). This is due to the fact that using an Independent copula ignores the dependence between the non-conformity scores, which leads to an over-estimation of the global hyper-rectangle error. This impact is avoided when using the Empirical copula because it takes advantage of the dependence structure to construct better

interval predictions. However, when the individual targets seem to be independent as given by the calibration curves, we can no longer notice a difference between using the different copula-based NCMs when measuring the efficiency, as for "bias corr" (Figure 4.8c). Another remark concerning efficiency is that the box plots for Empirical copula are tighter than the other two, which shows that the values are homogeneous on all folds compared to the Independent copula for instance, where the variability is much more visible.

When comparing between the underlying algorithms, we can see that the Neural Network gives tighter volumes for "sgemm" (Figure 4.9c). We can explain this by the fact that "sgemm" has more data, and the strong dependence structure is taken into consideration when training the Neural Network model that is learned on all targets simultaneously, as opposed to the Random Forest that is trained on each target individually. For "wq" (Figure 4.10c), the Random Forest gives better results since this data set has 14 targets but a few examples, which hinders the training process of the Neural Network, thus giving an advantage to the Random Forest underlying algorithm.

The empirical validity and efficiency results are summarized in Tables 4.8 and 4.9. The validity simply provides the average difference between a perfect calibration (the identity function) and the observed curve for each copula. This means, in particular, that a negative value indicates that the observed frequency is in average below the specified confidence degree.

Data set	Independent		**Gumbel**		**Empirical**	
	NN	RF	NN	RF	NN	RF
res building	4.36 ± 8.38	7.33 ± 7.01	-0.94 ± 9.34	$\mathbf{0.05 \pm 7.30}$	-2.26 ± 9.49	-1.59 ± 8.63
enb	2.95 ± 6.31	2.53 ± 4.48	-0.12 ± 6.32	-3.72 ± 4.41	$\mathbf{-0.11 \pm 6.52}$	-5.42 ± 4.22
music origin	2.19 ± 4.89	3.32 ± 4.68	-0.93 ± 4.66	$\mathbf{0.17 \pm 4.93}$	-1.41 ± 4.84	-0.56 ± 5.14
bias corr	1.87 ± 1.82	1.54 ± 1.48	0.59 ± 1.88	0.19 ± 1.64	0.66 ± 1.98	$\mathbf{0.15 \pm 1.69}$
jura	2.49 ± 8.77	$\mathbf{2.22 \pm 8.05}$	-3.55 ± 8.95	-3.56 ± 8.45	-4.63 ± 8.74	-6.05 ± 8.43
scpf	22.33 ± 4.79	18.56 ± 4.32	15.60 ± 4.70	11.57 ± 5.01	-3.47 ± 4.87	$\mathbf{0.48 \pm 5.79}$
indoorloc	3.77 ± 1.11	3.89 ± 1.56	1.80 ± 1.16	1.09 ± 1.39	$\mathbf{0.03 \pm 1.13}$	0.12 ± 1.40
sgemm	25.14 ± 0.84	28.07 ± 0.40	3.06 ± 0.68	1.99 ± 0.39	$\mathbf{-0.14 \pm 0.39}$	-0.15 ± 0.39
rf1	6.01 ± 1.44	4.99 ± 1.28	2.98 ± 1.38	1.98 ± 1.33	-0.40 ± 1.49	$\mathbf{-0.34 \pm 1.48}$
rf2	5.78 ± 2.68	4.94 ± 1.76	3.08 ± 2.37	1.98 ± 1.89	-0.30 ± 1.60	$\mathbf{0.24 \pm 1.68}$
osales	11.54 ± 6.29	15.94 ± 7.17	$\mathbf{1.54 \pm 6.65}$	3.86 ± 7.28	-8.54 ± 6.90	-3.86 ± 6.71
wq	3.65 ± 4.64	3.63 ± 4.28	$\mathbf{-2.59 \pm 4.53}$	-7.87 ± 4.84	-8.82 ± 4.71	-9.31 ± 4.49
scm1d	14.77 ± 2.84	14.58 ± 2.89	10.66 ± 2.67	9.79 ± 2.84	$\mathbf{-0.57 \pm 1.85}$	-0.79 ± 2.30
scm20d	14.44 ± 2.06	14.97 ± 2.02	10.52 ± 2.33	9.39 ± 2.10	$\mathbf{-1.16 \pm 2.01}$	-1.54 ± 2.09
com crime	17.08 ± 3.60	18.27 ± 3.22	10.71 ± 3.56	9.11 ± 4.15	-2.45 ± 3.36	$\mathbf{-2.38 \pm 3.76}$

Table 4.8: Area of validity summarized results for all data sets.
Best results are in bold. In red are mean values with a gap greater than 10 *and in orange are mean values with a gap between* 5 *and* 10.

	Independent		Gumbel		Empirical	
	NN	RF	NN	RF	NN	RF
res building	$8.86^{1} \pm 5.46^{1}$	1.01 ± 5.77^{-1}	$3.22^{1} \pm 1.03^{1}$	$3.97^{-1} \pm 2.18^{-1}$	$2.33^{1} \pm 7.15$	$\mathbf{2.54^{-1} \pm 1.03^{-1}}$
enb	3.43 ± 1.44	$1.14^{-1} \pm 3.03^{-2}$	2.76 ± 1.02	$\mathbf{8.45^{-2} \pm 2.88^{-2}}$	2.71 ± 1.23	$8.64^{-2} \pm 2.92^{-2}$
music origin	$4.02^{1} \pm 1.54^{1}$	$2.47^{1} \pm 1.18^{1}$	$3.27^{1} \pm 1.5^{1}$	$2.07^{1} \pm 1.1^{1}$	$3.08^{1} \pm 1.46^{1}$	$\mathbf{1.81^{1} \pm 7.82}$
bias corr	1.88 ± 2.86^{-1}	1.88 ± 2.43^{-1}	1.81 ± 2.73^{-1}	$\mathbf{1.81 \pm 2.26^{-1}}$	1.81 ± 2.73^{-1}	1.83 ± 2.25^{-1}
jura	$4.41^{2} \pm 4.94^{2}$	$3.72^{1} \pm 5.51^{1}$	$1.77^{2} \pm 1.61^{2}$	$\mathbf{9.66 \pm 7.64}$	$1.77^{2} \pm 1.61^{2}$	$\mathbf{9.66 \pm 7.64}$
scpf	$1.03^{11} \pm 3.02^{11}$	$8.72^{10} \pm 2.06^{11}$	$1.02^{11} \pm 3.02^{11}$	$7.56^{10} \pm 2.08^{11}$	$1.12^{7} \pm 2.05^{7}$	$\mathbf{5.71^{6} \pm 1.54^{7}}$
indoorloc	$1.31^{-1} \pm 7.77^{-2}$	$4.76^{-1} \pm 7^{-1}$	$1.15^{-1} \pm 7.95^{-2}$	$4.13^{-1} \pm 6.02^{-1}$	$\mathbf{1.03^{-1} \pm 7.65^{-2}}$	$4.26^{-1} \pm 6.5^{-1}$
sgemm	$7.97^{-4} \pm 4.81^{-4}$	$1.75^{-2} \pm 2.58^{-3}$	$2.47^{-4} \pm 1.45^{-4}$	$7.48^{-3} \pm 7.91^{-4}$	$\mathbf{2.17^{-4} \pm 1.25^{-4}}$	$7.4^{-3} \pm 8.15^{-4}$
rf1	$7.19^{-3} \pm 1.23^{-2}$	$5.64^{-5} \pm 4.87^{-5}$	$5.15^{-3} \pm 9.11^{-3}$	$3.56^{-5} \pm 3.44^{-5}$	$4.49^{-3} \pm 9.23^{-3}$	$\mathbf{2.81^{-5} \pm 1.67^{-5}}$
rf2	$2.17^{-3} \pm 2.89^{-3}$	$2.67^{-4} \pm 3.54^{-4}$	$1.67^{-3} \pm 2.45^{-3}$	$\mathbf{1.42^{-4} \pm 1.71^{-4}}$	$1.52^{-3} \pm 2.42^{-3}$	$\mathbf{1.42^{-4} \pm 1.71^{-4}}$
osales	$4.75^{17} \pm 1.42^{18}$	$7.18^{10} \pm 2.12^{11}$	$4.75^{17} \pm 1.42^{18}$	$7.18^{10} \pm 2.12^{11}$	$1.31^{10} \pm 2.02^{10}$	$\mathbf{8.89^{5} \pm 9.43^{5}}$
wq	$6.32^{15} \pm 1.88^{16}$	$1.55^{12} \pm 1.65^{12}$	$6.32^{15} \pm 1.88^{16}$	$8.29^{11} \pm 1.86^{12}$	$7.96^{11} \pm 8.97^{11}$	$\mathbf{5.09^{10} \pm 6.56^{10}}$
scm1d	$1.08^{5} \pm 1.04^{5}$	$1.72^{5} \pm 1.66^{5}$	$1.67^{4} \pm 1.33^{4}$	$2.11^{4} \pm 1.58^{4}$	$\mathbf{2.31^{3} \pm 1.93^{3}}$	$3.43^{3} \pm 2.27^{3}$
scm2od	$2.18^{6} \pm 4.26^{6}$	$5.77^{6} \pm 5.38^{6}$	$2.14^{5} \pm 2.88^{5}$	$1.02^{6} \pm 7.3^{5}$	$\mathbf{2.73^{4} \pm 2.58^{4}}$	$2.01^{5} \pm 1.06^{5}$
com crime	$1.28^{13} \pm 2.912^{13}$	$3.29^{11} \pm 6.25^{11}$	$5.11^{9} \pm 1.08^{10}$	$2.08^{8} \pm 3.59^{8}$	$5.39^{7} \pm 8.18^{7}$	$\mathbf{2.17^{6} \pm 4.25^{6}}$

We note X^{Y} the value $X \times 10^{Y}$.

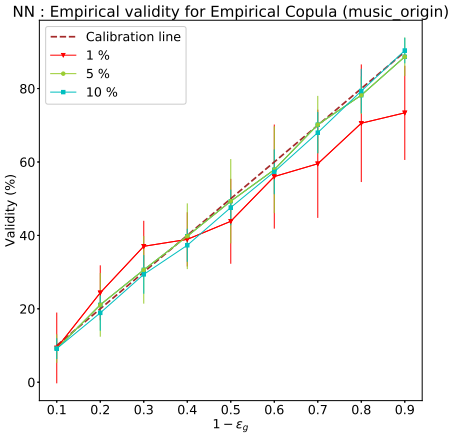
Table 4.9: Efficiency (hyper-rectangle median volume for $\epsilon_g = 0.1$) summarized results for all data sets.
Tighter volumes are in bold. Reported results are mean values of medians over all folds.

The numbers confirm our previous observations on the graphs, as the average gap is systematically higher for the Independent copula and lower for the Empirical one, with Gumbel in-between. In the few cases where the Independent copula gives better results, data is often scarce, as seen for "wq". We can however notice that while the Empirical copula provides the best results, it is also often a bit under the calibration line, indicating that if conservativeness is to be sought, one should maybe prefer the Gumbel copula. These outcomes are the same for both NN and RF, without one algorithm being overall better than the other. About the same conclusions can be given regarding efficiency, with the Empirical copula giving the best results and the Independent one the worst.

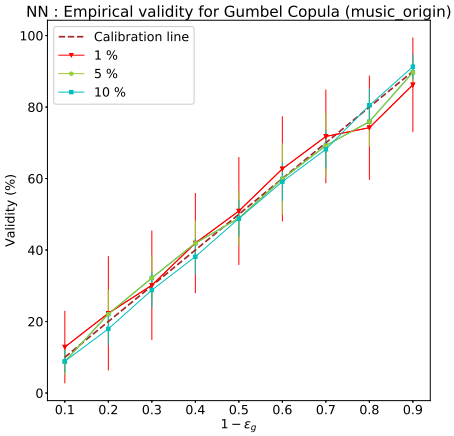
Analysis on the effect of calibration data size

To complete our experiments and analyze the sensitivity of our approach to the size of the calibration set, we conducted the same experiments on two data sets, where we retained only 1% and 5% of the whole data set: "indoorloc" which has a lot of examples (21049) and "music origin" which has fewer examples (1059). We only used Neural Networks as the underlying algorithm with the Empirical and Gumbel copulas non-conformity measures to compare between them. Figures 4.11 and 4.12 show the results for both data sets.

Results clearly show that with fewer examples for "music origin", the Empirical non-conformity measure is often invalid and also unstable (has larger variability) with 1% of the examples as calibration data (Figure 4.11a). This can also be seen in the difference of variance between

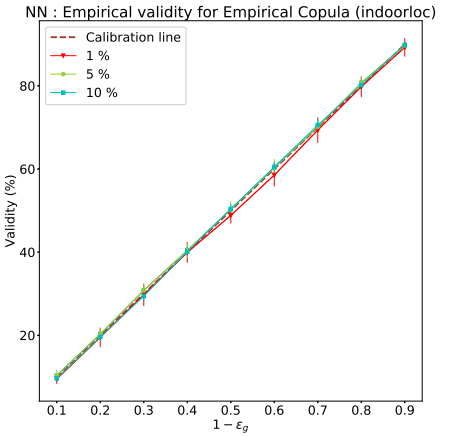


(a) Validity for the Empirical copula

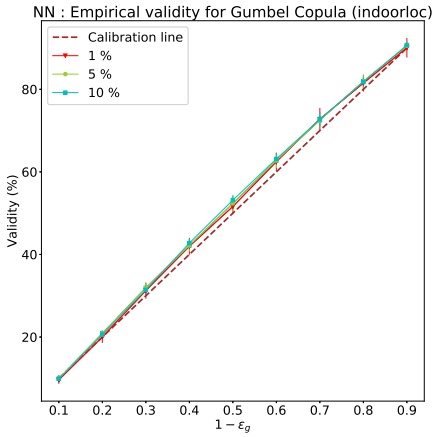


(b) Validity for the Gumbel copula

Figure 4.11: Copula conformal results for different calibration data sizes for "music origin".



(a) Validity for the Empirical copula



(b) Validity for the Gumbel copula

Figure 4.12: Copula conformal results for different calibration data sizes for "indoorloc".

values for the empirical validity, with 10% having more homogeneous values as compared to 5% and 1% respectively. Using the Gumbel copula, which is semi-parametric, helps to attenuate the effect, with more consistent results even for 1% (Figure 4.11b). For "indoorloc", the impact of the percentage of data used is insignificant, since the validity curves overlap for 10%, 5% and 1% of data used for calibration, mainly because 1% of the whole data set is still quite large (about 200 samples, to be compared with the 10 samples of "music origin"). This is the case for both

Empirical and Gumbel copulas, giving the same results as earlier, i.e. the Empirical copula being exactly valid and better than the Gumbel copula.

4.3.6 *Computation time*

Table 4.10 shows computation time for all data sets and for both underlying algorithms, since the normalizing MLP model is the same for all NCMs. Results clearly shows that Random Forests are faster than Neural Networks, with the exception of "scm1d".

Time (s)	**Neural Networks**	**Random Forests**
res building	32.96	11.44
enb	55.91	16.23
music origin	62.95	20.17
bias corr	611.35	135.80
jura	27.80	10.39
scpf	90.39	12.93
indoorloc	627.16	223.31
sgemm	6966.29	5229.56
rf1	480.45	141.96
rf2	463.85	238.86
osales	161.13	115.15
wq	108.99	25.77
scm1d	349.37	1032.2
scm20d	329.81	230.04
com crime	273.24	194.81

Table 4.10: Computation time for all real data sets.

4.4 ELLIPSOIDAL CONFORMAL MTR

This section will present our work on ellipsoidal conformal MTR, which tries to exploit the shared information between the different targets of a multi-target regression problem with another more flexible confidence region shape: ellipsoids.

An ellipsoid E is a set of the form:

$$E = \{x \in \mathbb{R}^n : (x-\mu)^T \Sigma (x-\mu) \leqslant 1\}, \tag{4.32}$$

where Σ is a (positive) definite matrix and $\mu \in \mathbb{R}^n$ is the center of the ellipsoid. Its volume is defined by:

$$\text{Vol}(E) = (\det(\Sigma))^{-1/2} \, \text{Vol}(B_n), \tag{4.33}$$

where the unit ball $B_n = \{x \in \mathbb{R}^n : ||x||_2 \leqslant 1\}$ is an ellipsoid and its volume depending on the dimension n is calculated as:

$$\begin{aligned} &\text{Vol}(B_2) = \pi, \qquad \text{Vol}(B_3) = \frac{4\pi}{3}, \\ &\text{Vol}(B_n) = \text{Vol}(B_{n-2}) \times \left(\frac{2\pi}{n}\right). \end{aligned} \tag{4.34}$$

4.4.1 *Ellipsoidal non-conformity measures*

Knowing that the ellipsoid[5] is one of the most flexible geometrical shapes, [Johnstone et al., 2021] proposed for a robust optimization problem an ellipsoidal NCM:

$$\alpha_i = \sqrt{(y_i - \hat{y}_i)^T \hat{\Sigma}^{-1} (y_i - \hat{y}_i)}, \tag{4.35}$$

where $y_i - \hat{y}_i$ is a vector of univariate non-conformity scores (here, the regressor's error rate) and $\hat{\Sigma}^{-1}$ is the sample inverse-covariance matrix of the observed errors.

The standard global ellipsoidal NCM in (4.35) uses a sample inverse-covariance matrix $\hat{\Sigma}^{-1}$ globally-estimated from the training data errors. Therefore, it resembles the standard approach in the univariate case where the non-conformity score is not tailored to each instance. We propose to use instead a normalized inverse-covariance matrix $\hat{\Sigma}_i^{-1}$ for each instance x_i, to take into consideration a locally-estimated covariance matrix of the instance. The normalized NCM in this case is:

$$\alpha_i = \sqrt{(y_i - \hat{y}_i)^T \hat{\Sigma}_i^{-1} (y_i - \hat{y}_i)} \tag{4.36}$$

[5] Note that the term "ellipsoid" used in this paper refers to a k-dimensional ellipsoid.

This NCM enables us to define the conformal prediction region as an ellipsoid E_i given by the following theorem:

Theorem 4.4.1 *The ellipsoid* E_i *given by the NCM*

$$\alpha_i = \sqrt{(y_i - \hat{y}_i)^T \hat{\Sigma}_i^{-1} (y_i - \hat{y}_i)},$$

which center is the regressor's prediction $\hat{y}_i$*, and covariance matrix is* $\frac{\hat{\Sigma}_i^{-1}}{\alpha_s^2}$*, is a conformal valid prediction.*

Proof 2 *Let* E_i *be the ellipsoid to which the ground truth* y_i *of an instance* x_i *should belong. To satisfy the validity of a conformal predictor considering a significance level* ϵ*, we have:*

$$\begin{aligned}
&P(y_i \in E_i) \geqslant 1 - \epsilon \\
&\iff P(\alpha_i \leqslant \alpha_s) \geqslant 1 - \epsilon. \\
&\iff P\left(\sqrt{(y_i - \hat{y}_i)^T \hat{\Sigma}_i^{-1} (y_i - \hat{y}_i)} \leqslant \alpha_s\right) \geqslant 1 - \epsilon. \\
&\iff P\left(\sqrt{(y_i - \hat{y}_i)^T \frac{\hat{\Sigma}_i^{-1}}{\alpha_s^2} (y_i - \hat{y}_i)} \leqslant 1\right) \geqslant 1 - \epsilon.
\end{aligned}$$

Following an ellipsoid's definition, we can see that

$$E_i = \left\{ y_i \in \mathbb{R}^k : (y_i - \hat{y}_i)^T \frac{\hat{\Sigma}_i^{-1}}{\alpha_s^2} (y_i - \hat{y}_i) \leqslant 1 \right\}.$$

E_i *is the ellipsoid given by the center* $\hat{y}_i$ *and the matrix* $\frac{\hat{\Sigma}_i^{-1}}{\alpha_s^2}$*.* ■

The volume of E_i, its efficiency, is equal to the volume of an ellipse:

$$\text{Vol}(E_i) = \alpha_s^k \det\left(\hat{\Sigma}_i\right)^{1/2} \text{Vol}(B_k), \tag{4.37}$$

where $B_k = \{y \in \mathbb{R}^k : \|y\|_2 \leqslant 1\}$ is the unit ball.

Figure 4.13 illustrates ellipsoidal conformal regions for a 2-dimensional MTR problem compared to a hyper-rectangle conformal region for the same object x_{n+1}. It shows that they can be different depending on the relationship between the outputs.

Local covariance matrix estimation

As mentioned above, the NCM given by (4.36) is mainly composed of a normalized inverse-covariance matrix $\hat{\Sigma}_i^{-1}$ for each instance x_i, meaning that we need to estimate locally a covariance matrix $\hat{\text{Cov}}_i$ for x_i. Thus, we propose to calculate $\hat{\Sigma}_i$ by:

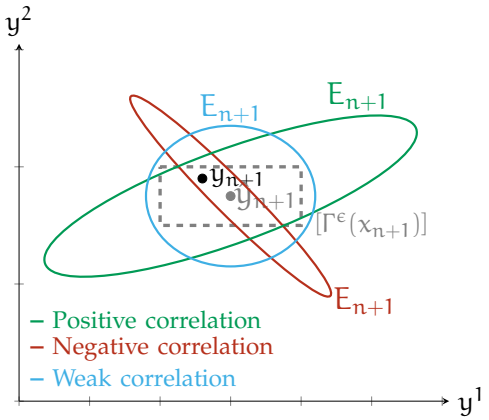


Figure 4.13: Illustration of an ellipsoidal conformal prediction.

$$\hat{\Sigma}_i = \lambda \hat{\text{Cov}}_i + (1-\lambda)\hat{\Sigma}, \tag{4.38}$$

where $\hat{\Sigma}$ is the global covariance matrix estimated from error rates over all training instances in $\mathbf{Z}^{tr}$, that is from the observed errors $(y_i - \hat{y}_i)$, $\hat{\text{Cov}}_i$ is the local covariance matrix of instance x_i, and λ is a parameter to control the trade-off between $\hat{\Sigma}$ and $\hat{\text{Cov}}_i$ in order to get a positive definite matrix $\hat{\Sigma}_i$ and avoid numerical instability. Its value should be high enough to ensure that the influence of $\hat{\text{Cov}}_i$ is predominant compared to $\hat{\Sigma}$.

To estimate $\hat{\text{Cov}}_i$, we propose to use a kNN model to get the neighboring instances of x_i from $\mathbf{Z}^{tr}$, and then use the observed error rates on these instances to estimate $\hat{\text{Cov}}_i$. That is, given x_i, we consider the k nearest instances $x_j \in \mathbf{Z}^{tr}$ from x_i, and estimate $\hat{\text{Cov}}_i$ from the observed errors $(y_j - \hat{y}_j)$ for instances x_j.

4.4.2 *Experimental setting*

Our experiments are conducted using Python and the code is available in GitHub [6]. Their objective is to compare between four NCMs:

- **Standard Empirical Copula (SEC):** produces same-size hyper-rectangle conformal regions by non-parametrically estimating the dependence structure from calibration data. Although not present in Section 4.3, we can easily derive it by considering the standard NCM $\alpha_i = |y_i - \hat{y}_i|$.

- **Normalized Empirical Copula (NEC):** constructs personalized hyper-rectangle conformal regions with a difficulty estimator σ_i learned by a Multi-Layer Perceptron.

[6] https://github.com/M-Soundouss/EllipsoidalConformalMTR

- **Standard Global Ellipsoid (SGE):** generates same-size ellipsoidal conformal regions by exploiting the global inverse-covariance matrix $\hat{\Sigma}^{-1}$ from training data [Johnstone et al., 2021].
- **Normalized Local Ellipsoid (NLE):** gets individual ellipsoidal conformal regions using the local covariance matrix of each instance.

Table 4.11 summarizes the features of the different methods. We can see from it that local ellipses can capture both covariance structures and local variations, hence providing possibly better results.

	Is Local	Captures Covariance
SEC	x	x
NEC	✓	x
SGE	x	✓
NLE	✓	✓

Table 4.11: Properties of the used non-conformity measures.

The experiments are conducted with a 10-fold cross validation following the steps in Figure 4.14 described below:

1. Split data into proper training $\mathbf{Z}^{tr}$, calibration $\mathbf{Z}^{cal}$ and test $\mathbf{Z}^{ts}$ sets, by allocating 10% of the data to $\mathbf{Z}^{ts}$, and splitting the remaining 90% of the instances into $\mathbf{Z}^{tr}$ and $\mathbf{Z}^{cal}$, with $|\mathbf{Z}^{cal}| = 10\%$ of training data.
2. Train the underlying algorithm on $\mathbf{Z}^{tr}$, here a multi-output random forest using Scikit Learn's "MultiOutputRegressor", calculate $\hat{\Sigma}$ of the errors made on $\mathbf{Z}^{tr}$ instances, and get the regressor's predictions for $\mathbf{Z}^{cal}$ and $\mathbf{Z}^{ts}$.
3. For the normalized NCMs, train the normalizing models on $\mathbf{Z}^{tr}$: an MLP for NEC to get σ_i by learning μ_i, and a kNN for our method NLE to get the neighboring instances of x_i, which number is equal to 5% of the number of $\mathbf{Z}^{tr}$ instances, to calculate $\hat{\Sigma}_i$.
4. For each instance in $\mathbf{Z}^{cal}$ and $\mathbf{Z}^{ts}$, get σ_i values for NEC using the MLP, and $\hat{\Sigma}_i$ values for NLE using the kNN with $\lambda = 0.95$, a high value that favors the impact of $\hat{\text{Cov}}_i$ in (4.38).
5. Choose the significance level's value ϵ.
6. Obtain the non-conformity scores for the different NCMs for $\mathbf{Z}^{cal}$, sort $\alpha_1, \ldots, \alpha_q$ in a descending order and get the index s of the $(1-\epsilon)$-percentile of the non-conformity score α_s.
7. For a new object x_{n+1} in $\mathbf{Z}^{ts}$, get its hyper-rectangle or ellipsoidal prediction region depending on the used NCM.

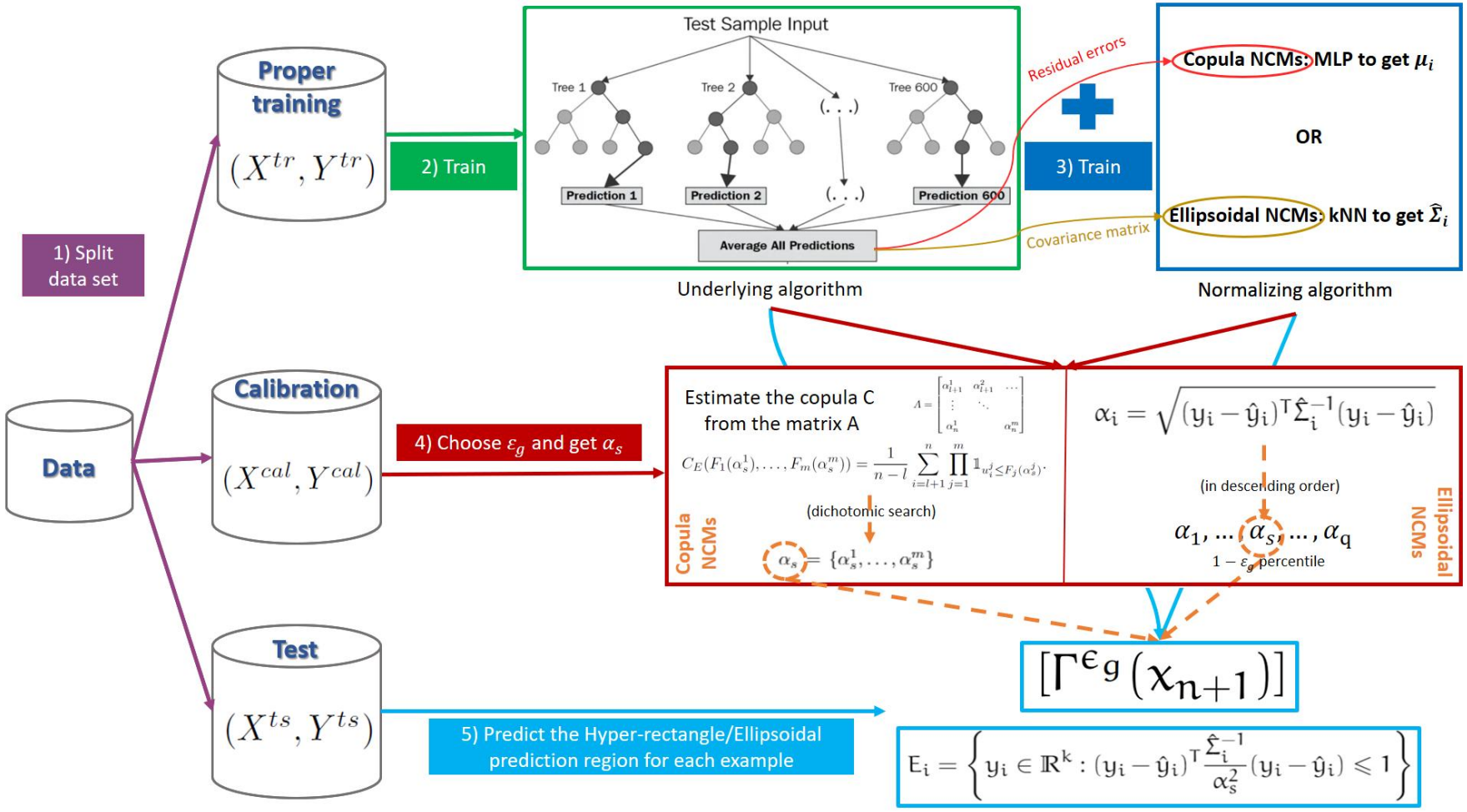


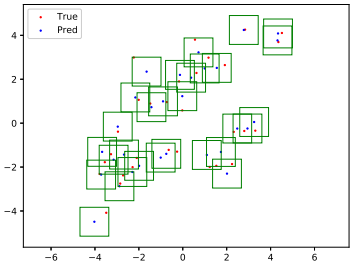
Figure 4.14: Ellipsoidal vs copula-based MTR conformal prediction approach.

4.4.3 *Results on synthetic data*

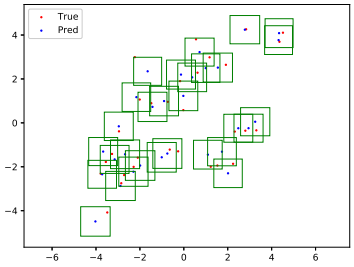
For synthetic data, we calculated the mean validity and efficiency (surface) of all non-conformity measures for different ϵ values. Results are summed up in Table 4.12. They show that all NCMs are exactly valid, with a slight difference between them. However, when it comes to efficiency, our local ellipsoid NCM outperforms the others, giving the best results with tighter conformal prediction regions.

synthetic (k = 2)		$\epsilon = 0.01$	$\epsilon = 0.05$	$\epsilon = 0.1$	$\epsilon = 0.15$	$\epsilon = 0.2$
Validity	SEC	99.21 ± 0.12	95.11 ± 0.23	90.25 ± 0.35	85.07 ± 0.45	80.07 ± 0.79
	NEC	99.24 ± 0.11	95.08 ± 0.31	90.26 ± 0.39	84.99 ± 0.41	80.12 ± 0.81
	SGE	98.97 ± 0.17	95.02 ± 0.37	90.00 ± 0.50	84.95 ± 0.49	79.94 ± 0.73
	NLE	99.01 ± 0.15	94.89 ± 0.28	90.02 ± 0.48	84.91 ± 0.33	80.00 ± 0.45
Efficiency	SEC	3.95 ± 0.13	2.32 ± 0.04	1.75 ± 0.03	1.39 ± 0.02	1.16 ± 0.02
	NEC	3.99 ± 0.14	2.32 ± 0.04	1.76 ± 0.03	1.38 ± 0.02	1.16 ± 0.02
	SGE	4.19 ± 0.17	2.51 ± 0.05	1.84 ± 0.04	1.46 ± 0.02	1.20 ± 0.02
	NLE	$\mathbf{2.85 \pm 0.07}$	$\mathbf{1.81 \pm 0.02}$	$\mathbf{1.39 \pm 0.02}$	$\mathbf{1.14 \pm 0.01}$	$\mathbf{0.97 \pm 0.01}$

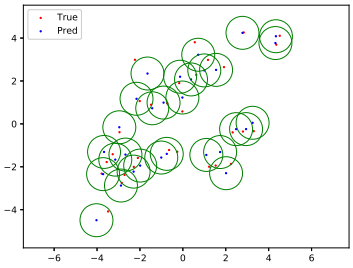
Table 4.12: Validity and efficiency results for synthetic data.



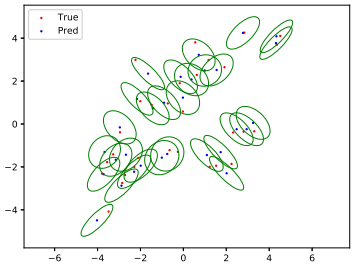
(a) SEC



(b) NEC



(c) SGE



(d) NLE

Figure 4.15: Results' visualization for synthetic data with $\epsilon = 0.1$.

For visualization purposes, we drew prediction regions (a rectangle for the empirical copula NCMs and an ellipse for the ellipsoid NCMs) in

Figure 4.15. We notice that both empirical copula NCMs give almost the same results, which explains why their efficiency results in Table 4.12 are almost equal. Another remark is that the normalized ellipse NCM in sub-figure 4.15d shows that the drawn ellipses follow the choice of μ_i values and their covariance matrices used to generate synthetic data, which shows that our method respects the local covariance, thus explaining the gain in efficiency going from a rectangle to an ellipse.

4.4.4 *Results on real data*

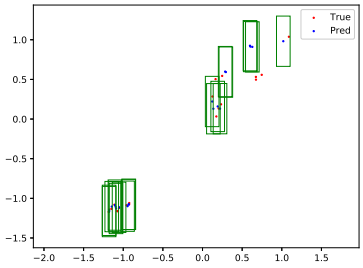
Following the same experiment protocol described above, we calculate validity and efficiency values for all NCMs with $\epsilon = 0.1$. These results are summarized in Table 4.13.

$\epsilon = 0.1$	Validity				Efficiency			
	SEC	NEC	SGE	NLE	SEC	NEC	SGE	NLE
res building	83.59 ± 6.73	85.48 ± 4.02	85.75 ± 5.08	89.54 ± 5.11	0.30 ± 0.07	0.22 ± 0.06	0.31 ± 0.08	**0.17 ± 0.05**
enb	83.98 ± 6.37	84.89 ± 5.99	87.23 ± 4.15	89.57 ± 5.22	0.13 ± 0.03	0.08 ± 0.02	0.11 ± 0.02	**0.04 ± 0.02**
music origin	88.76 ± 3.59	88.66 ± 2.64	89.70 ± 4.80	89.05 ± 4.30	10.98 ± 1.11	16.54 ± 3.30	10.60 ± 1.41	**9.55 ± 0.73**
bias corr	89.98 ± 0.93	90.42 ± 1.32	90.24 ± 1.06	90.29 ± 1.48	1.47 ± 0.06	1.32 ± 0.05	1.37 ± 0.05	**1.19 ± 0.07**
jura	86.93 ± 6.20	85.81 ± 4.69	86.64 ± 5.22	88.30 ± 8.03	24.29 ± 14.14	12.79 ± 7.57	12.89 ± 4.26	**10.17 ± 5.60**
scpf	84.52 ± 5.18	84.26 ± 4.95	87.86 ± 5.02	87.87 ± 4.75	$3.77^{10} \pm 6.83^{10}$	$1.26^{10} \pm 2.81^{10}$	$4.87^{7} \pm 8.71^{6}$	**69.59 ± 89.96**
indoorloc	90.32 ± 0.48	90.32 ± 0.58	90.37 ± 0.96	90.11 ± 0.89	0.06 ± 0.01	**0.05 ± 0.01**	0.07 ± 0.01	0.29 ± 0.03
sgemm	90.04 ± 0.17	90.05 ± 0.27	89.98 ± 0.20	90.03 ± 0.17	$8.45^{-5} \pm 2.56^{-6}$	$7.34^{-5} \pm 3.15^{-6}$	$1.84^{-5} \pm 4.93^{-7}$	$\mathbf{1.15^{-5} \pm 3.41^{-7}}$
atp1d	72.09 ± 11.19	66.74 ± 10.73	85.18 ± 7.08	85.78 ± 5.50	6.25 ± 3.47	1.02 ± 1.15	8.17 ± 5.63	**0.47 ± 0.45**
atp7d	72.29 ± 11.82	68.54 ± 12.96	81.82 ± 10.41	86.13 ± 10.83	8.07 ± 10.73	**0.64 ± 0.35**	6.84 ± 9.23	4.11 ± 7.57
rf1	89.10 ± 2.14	89.13 ± 1.99	90.04 ± 1.50	90.20 ± 1.53	$5.75^{-7} \pm 3.79^{-7}$	$3.60^{-7} \pm 2.16^{-7}$	$6.20^{-7} \pm 5.27^{-7}$	$\mathbf{4.14^{-8} \pm 3.34^{-8}}$
rf2	90.18 ± 1.53	89.79 ± 1.76	90.26 ± 1.31	89.98 ± 1.36	$7.50^{-7} \pm 4.40^{-7}$	$3.13^{-7} \pm 2.38^{-7}$	$6.49^{-7} \pm 4.96^{-7}$	$\mathbf{4.44^{-8} \pm 2.41^{-8}}$
osales	81.39 ± 7.02	78.86 ± 7.88	86.71 ± 3.95	88.12 ± 4.64	$5.60^{6} \pm 9.55^{6}$	$1.20^{15} \pm 3.45^{15}$	$1.47^{5} \pm 2.46^{5}$	$\mathbf{8.08^{4} \pm 1.26^{5}}$
wq	72.74 ± 4.43	78.49 ± 2.76	89.15 ± 4.91	87.55 ± 3.91	$1.31^{10} \pm 6.27^{9}$	$3.93^{17} \pm 1.18^{18}$	$\mathbf{1.16^{9} \pm 7.06^{8}}$	$1.35^{9} \pm 1.08^{9}$
scm1d	89.70 ± 1.27	89.60 ± 1.61	90.07 ± 1.27	90.42 ± 1.25	$6.86^{4} \pm 2.92^{4}$	$1.12^{3} \pm 6.22^{2}$	$4.21^{2} \pm 1.79^{2}$	**8.86 ± 3.42**
scm20d	88.23 ± 1.14	88.72 ± 1.64	89.26 ± 1.02	89.45 ± 1.15	$7.98^{5} \pm 6.57^{5}$	$7.02^{4} \pm 3.85^{4}$	$3.21^{3} \pm 2.27^{3}$	$\mathbf{5.52^{2} \pm 3.54^{2}}$
oes10	74.15 ± 13.92	66.13 ± 19.95	87.57 ± 8.13	88.58 ± 6.84	$7.62^{5} \pm 2.23^{6}$	$1.05^{5} \pm 2.86^{5}$	$1.83^{6} \pm 4.83^{6}$	$\mathbf{8.25^{3} \pm 1.26^{4}}$
oes97	69.11 ± 8.18	62.94 ± 15.06	89.22 ± 6.47	87.99 ± 7.41	$\mathbf{1.45^{5} \pm 4.16^{5}}$	$3.86^{6} \pm 8.38^{6}$	$6.90^{6} \pm 1.53^{7}$	$1.18^{7} \pm 1.82^{7}$
com crime	86.18 ± 3.51	84.61 ± 3.04	90.21 ± 3.66	89.03 ± 2.34	$5.19^{9} \pm 9.74^{9}$	$7.05^{4} \pm 7.61^{4}$	$1.17^{5} \pm 1.49^{5}$	**2.80 ± 7.01**

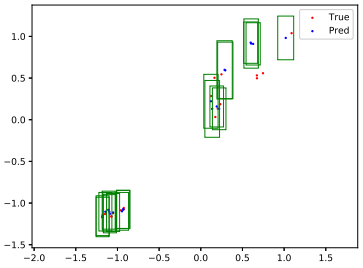
We note X^Y the value $X \times 10^Y$.

Table 4.13: Validity and efficiency results for all real data sets.
For validity, in red are mean values lower than 80% and in orange are mean values between 80% and 85%. For efficiency, tighter volumes are in bold. Reported results are mean values of medians over all folds.

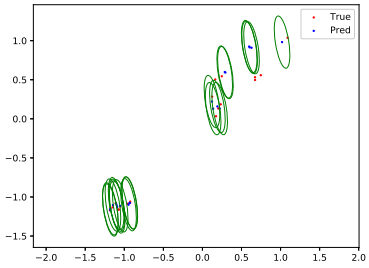
Validity results show that the ellipsoid NCMs perform better than the empirical copula NCMs, especially when it comes to small data sets (with less than 1000 instances). For these data sets, the empirical copula NCMs are invalid with higher variance. This can be explained by the fact that these NCMs are non-parametric, mostly relying on the size of calibration data which is insufficient for smaller data sets. This simply confirms that good performances are achieved when there is a good trade-off between the inductive bias (the amount of made hypothesis) of the method and the amount of data at disposal. Empirical copulas, that make very little assumptions, perform badly when having only a small amount of $\mathbf{Z}^{\text{cal}}$. Even with our theoretical guaranties, the effect of low calibration data can still be noticed for our method (as in "atp7d"), however its impact is less visible.



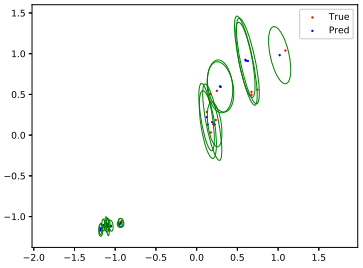
(a) SEC



(b) NEC

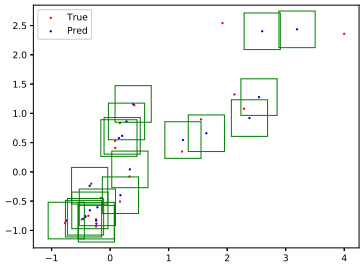


(c) SGE

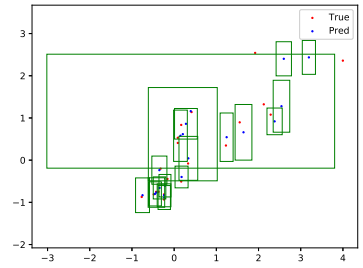


(d) NLE

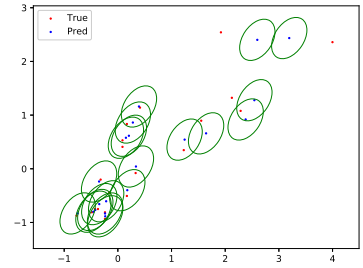
Figure 4.16: Results' visualization for "enb" with $\epsilon = 0.1$.



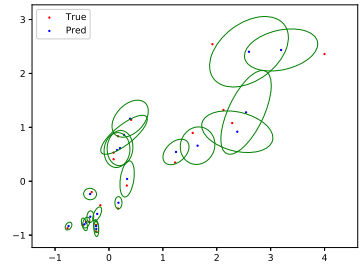
(a) SEC



(b) NEC



(c) SGE



(d) NLE

Figure 4.17: Results' visualization for "res building" with $\epsilon = 0.1$.

Efficiency results show that our method gives the tightest volumes most times, with a massive difference when compared to other NCMs in some cases such as "scpf" and "osales", "scm1d", "scm20d" and "community crime". This confirms that our NCM takes advantage of the flexibility of an ellipsoid shape in order to give tailored prediction regions for each instance depending on its local covariance. For the few data sets where an empirical copula approach gives a tighter volume, this NCM tends to be invalid (except for "indoor localization").

Figures 4.16 and 4.17 illustrate prediction regions for all NCMs for "enb" and "residential building" data sets. For "enb", we can clearly see that even if the dependence structure is roughly the same for all instances, our method NLE enables us to have adjustable ellipsoid sizes. For "residential building", the different dependence structures in each space region emphasizes the importance of using an approach that takes into consideration local covariance matrices for each instance, and thus, give a noticeable advantage to our method, especially when compared to NEC.

4.4.5 *Computation time*

Table 4.14 shows computation time for all data sets and for all NCMs.

Time (s)	Proper Train	SEC	NEC	SGE	NLE
res building	1.58 ± 0.17	0.22 ± 0.17	5.85 ± 1.48	0.10 ± 0.02	0.13 ± 0.06
enb	0.37 ± 0.01	0.16 ± 0.02	5.79 ± 0.78	0.10 ± 2.03^{-3}	0.13 ± 0.03
music origin	5.58 ± 0.51	0.17 ± 0.03	9.63 ± 2.03	0.11 ± 0.03	0.15 ± 0.02
bias corr	13.07 ± 1.02	0.21 ± 0.03	56.53 ± 1.16^{2}	0.14 ± 0.05	0.89 ± 0.11
jura	0.60 ± 0.04	0.07 ± 0.01	8.48 ± 1.68	$5.99^{-4} \pm 1.20^{-3}$	$9.94^{-3} \pm 8.94^{-4}$
scpf	2.49 ± 0.37	0.35 ± 0.05	6.31 ± 7.52	$1.30^{-3} \pm 6.39^{-4}$	0.07 ± 0.01
indoorloc	$2.25^{2} \pm 7.98$	0.35 ± 0.05	11.95 ± 0.41	$1.28^{-3} \pm 4.58^{-4}$	82.40 ± 3.91
sgemm	$2.62^{2} \pm 13.51$	4.84 ± 0.47	$9^{2} \pm 1.28^{2}$	0.01 ± 1.69^{-3}	$1.20^{3} \pm 15.07$
atp1d	15.74 ± 0.39	0.13 ± 0.01	10.82 ± 1.39	$5.98^{-4} \pm 4.88^{-4}$	0.04 ± 3.64^{-3}
atp7d	13.27 ± 0.56	0.12 ± 0.01	9.01 ± 4.16	$2.98^{-4} \pm 4.55^{-4}$	0.02 ± 7.84^{-4}
rf1	$1.79^{2} \pm 16.74$	0.52 ± 0.04	6.38 ± 0.79	$1.58^{-3} \pm 4.87^{-4}$	1.90 ± 0.12
rf2	$6.01^{2} \pm 28.05$	0.48 ± 0.05	7.91 ± 1.67	$1.57^{-3} \pm 4.97^{-4}$	16.56 ± 1.16
osales	20.62 ± 2.15	0.27 ± 0.03	12.83 ± 8.82	$9.83^{-4} \pm 4.38^{-4}$	0.15 ± 0.02
wq	9.26 ± 0.89	0.36 ± 0.05	12.05 ± 5.17	$1.08^{-3} \pm 2.94^{-4}$	0.06 ± 0.01
scm1d	$2.91^{3} \pm 9.82^{2}$	1.24 ± 0.51	53.24 ± 34.83	$3.13^{-3} \pm 1.01^{-3}$	15.99 ± 5.36
scm20d	$4.57^{2} \pm 9.06$	0.99 ± 0.14	62.22 ± 6.50	$2.87^{-3} \pm 6.96^{-4}$	2.82 ± 0.34
oes10	45.35 ± 3.42	0.32 ± 0.04	14.83 ± 3.18	$1.09^{-3} \pm 6.98^{-4}$	0.05 ± 0.01
oes97	33.52 ± 2.51	0.33 ± 0.04	19.87 ± 1.07	$1.08^{-3} \pm 6.88^{-4}$	0.04 ± 9.27^{-3}
com crime	$2.18^{2} \pm 8.46$	0.50 ± 0.07	28.76 ± 2.82	$1.69^{-3} \pm 7.79^{-4}$	0.41 ± 0.06

We note X^{Y} the value $X \times 10^{Y}$.

Table 4.14: Computation time for all real data sets.

When comparing both standard methods, we can see that the standard global ellipsoid NCM is faster. This also applies on normalized NCMs

with the normalized local ellipsoid NCM outperforming the normalized empirical copula NCM (except for "indoorloc", "sgemm" and "rf2"). This can be explained by the fact that the MLP model used to get σ_i takes longer to train than the kNN used for our ellipsoidal method. Of course, the computation time for the normalized empirical copula approach can be smaller if we change the normalizing model, but we decided to use the same model as in our previous work for comparison purposes. We can also explain the longer computation time for "indoorloc", "sgemm" and "rf2" with the fact that these data sets are larger, providing us with a bigger number of neighbors (which size is equal to 5% of training data) to take into consideration for each time.

4.5 CONCLUSION

In this chapter, we proposed three approaches to produce conformal regions while dealing with multi-target regression problems.

In our first approach, we applied inductive conformal prediction to multi-target regression using deep neural networks. We extended non-conformity measures from the single-output regression problem to the multi-target regression case and proposed new non-conformity measures. We also introduced a naive method to compute the necessary significance levels for each target to get the desired overall confidence level, which was proven to work with our empirical study. However, this naive approach produces over-conservatively valid NCMs (as expected from a Bonferroni-like correction), that often give big hyper-rectangle regions with high variability.

Our second approach dealt with these issues by exploiting a link between non-conformity scores and copulas, a commonly used tool to model multi-variate distribution, in order to obtain valid NCMs for multi-target regression. Experiments on various data sets for a small choice of representative copulas show that the method indeed allows to improve upon the naive independence assumption for different underlying algorithms (Neural Networks and Random Forests). Those first results indicate in particular that while parametric, simple copulas may provide valid results for some data sets, more complex copulas may be needed in general to obtain well calibrated predictions, with the cost that good estimations of such copulas require a lot of calibration data.

In our third and final approach, we proposed a more flexible conformal MTR method that provides normalized ellipsoidal uncertainty regions based on the local covariance matrix of the instance. This method showed that it can give tighter volumes compared to our copula-based approach, and thus better efficiency results while maintaining a validity defined by the required confidence level.

Chapter 5

Conformal prediction applied to real estate management

> *Torture the data, and it will confess to anything.*
>
> **— Ronald Coase**

Contents

This thesis is part of a partnership between the University of Technology of Compiègne and Sopra/Steria. One of the objectives of this partnership is to help the company enrich its real estate management services offer and enhance the value of the numerous building data (BIM: Building Information Modeling, IoT: Internet of Things and ERP: Enterprise Resource Planning) that are not yet exploited to their full potential. As a result, tenants' debt prediction is a project that was conducted during this thesis in order to apply conformal prediction methods to one of Sopra/Steria customers' problems. In this chapter, we will present this work, mainly based on our paper [Messoudi et al., 2021a].

5.1 Tenants' debt prediction: problem description

Social housing is housing intended for people with modest incomes who would have difficulty finding housing on the private market. The social

lease is granted under conditions of income or family composition. Even if these rents are smaller than the average rents in the geographic sector, the social housing remains a victim of unpaid rent. The generating facts that explain these unpaid situations originate from predictable reasons (job insecurity, multiple consumer loans, etc.) or unpredictable (tight budget, health problems, change in family situations such as a birth or a divorce...). In addition, the number of households in debt is likely to increase with issues such as the health crisis due to the COVID-19 pandemic [Manville et al., 2020]. In each case, social property owners must study the particular situation of the household in difficulty of payment and hire social counsellors and litigation managers to help them resolve their problems before reaching an unfortunate eviction phase.

Thus, it is essential for social property owners to anticipate tenants that are likely to fall into debt, and more importantly, limit the number of tenants that are misclassified as in debt in order to avoid paying unnecessary costs, or wasting some social agent time that could otherwise have been used to the benefit of tenants really in need. On the other side, while it is important to maintain a good overall accuracy, the accuracy on the debt class is not so important, as misclassifying a tenant as having no debt only delays the recovering procedure. To control all these factors, we propose a class-wise confidence approach based on Mondrian conformal prediction which was previously presented in 3.4. Before explaining our conformal class-wise approach, we will briefly recall classic techniques for error control in binary classification, and detail the data preparation approach.

5.2 SOME ERROR CONTROL METHODS FOR BINARY CLASSIFICATION

In a supervised learning setting, binary classification consists of classifying an object $x_i \in \mathbf{X}$ into one of two labels: the negative class 0 or the positive class 1, such that the target space is $\mathbf{Y} = \{0, 1\}$. To evaluate a binary classifier, a confusion matrix can be drawn based on its predictions as follows:

	Predicted Positive	**Predicted Negative**
Ground-truth Positive	TP (True Positive)	FN (False Negative)
Ground-truth Negative	FP (False Positive)	TN (True Negative)

Table 5.1: Confusion matrix for a binary classification problem.

This confusion matrix is used to assess the number of examples that were correctly classified (i.e. TP and TN), and those who were misclassified (i.e. FP and FN). These values are then exploited to compute the *accuracy* and its complement the *error rate* of the classifier as:

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + TN + FP} \quad \text{Error rate} = 1 - \text{Accuracy}. \tag{5.1}$$

In most cases, traditional classification techniques aim at minimizing the error rate by giving the same importance to both types of errors FP and FN. However, this can be problematic, particularly when data is imbalanced. Some methods emerged to treat the asymmetry that can occur in errors by differentiating between FPR, i.e. the *False Positive Rate* also referred to as type I error, and FNR, meaning the *False Negative Rate* also called type II error. They are defined as follows:

$$FPR(\hat{y}) = P(\hat{y} = 1 \mid y = 0) \quad FNR(\hat{y}) = P(\hat{y} = 0 \mid y = 1). \tag{5.2}$$

COST-SENSITIVE CLASSIFICATION Cost-sensitive classification (CSC) [Domingos, 1999] is a framework implemented to control asymmetric error by taking into account different costs for both FPR and FNR. Many algorithms were proposed to consider these costs from the start, such as a hybrid genetic decision tree [Turney, 1994] and BEE-Miner which is based on bees algorithm [Tapkan et al., 2016]. Other methods modify existing cost-insensitive algorithms to make them cost-sensitive such as:

- Thresholding [Sheng et al., 2006]: seeks the best classifier's probability estimate from training data as the threshold, and uses it to predict the label of test data.

- Rebalancing [Zadrozny et al., 2003]: changes the proportion of the labels in training data either by assigning weights to instances or by over-sampling/under-sampling.

NEYMAN-PEARSON CLASSIFICATION Neyman-Pearson classification (NPC) is a method that applies the Neyman-Pearson lemma [Lann, 1959] to find a classifier that solves, during training, the following minimization problem:

$$\min_{FPR(\hat{y}) \leqslant \zeta} FNR(\hat{y}), \tag{5.3}$$

where $\zeta \in (0, 1)$ is a small user-specified level. This is done by following an empirical risk minimization or a plug-in approach. Unlike other methods, NPC gives a probabilistic guarantee over the type I error bound. For more details, the reader can refer to [Tong et al., 2016].

5.3 CLASS-WISE MCP

Our primary goal in this work and the associated application is to control the error rate of a given class, while preserving a relatively low global error rate. A specificity of this case is that the error rate of the remaining class is of marginal importance, and to some extent what really matters for this second class is the efficiency, i.e., the ability to identify some samples belonging to it. In practice, this means that we need to specify different significance levels for the classes and for the global data set. For this purpose, we adapt the class-conditional Mondrian conformal prediction described in Section 3.4.

Let $\epsilon_g \in (0,1)$ be the global error rate for all the data set, $\epsilon_0 \in (0,1)$ be the one specified for the label $y = 0$, meaning that the person is not in debt, and $\epsilon_1 \in (0,1)$ be the one related to the class $y = 1$, i.e. the person is in debt. ϵ_0 is therefore the variable over which we wish to have a strong control (in order not to send unnecessary social agents to the tenants). We have

$$\epsilon_g = \epsilon_0 P(y = 0) + \epsilon_1 P(y = 1). \tag{5.4}$$

With ϵ_g and ϵ_0 chosen by the user, and provided we have reasonable estimations of $P(y = 0)$ and $P(y = 1)$, we can calculate ϵ_1 by:

$$\epsilon_1 = \frac{\epsilon_g - \epsilon_0 P(y = 0)}{P(y = 1)}. \tag{5.5}$$

When defining ϵ_g and ϵ_0, and since $\epsilon_1 \in (0,1)$, we need to respect the condition:

$$\epsilon_0 P(y = 0) < \epsilon_g < \epsilon_0 + (1 - \epsilon_0) P(y = 1), \tag{5.6}$$

otherwise we would obtain an unfeasible ϵ_1.

This enables us to have individual significance levels for each class that will guarantee an overall confidence level for the data set. Apart from this step, the other steps of class-conditional MCP remain the same. Thus, in our approach, we have the following procedure:

1. Split the original data set into a proper training set, a calibration set per class and a test set.
2. Use the proper training set to train the underlying algorithm, get the output predictions for the calibration and test sets and calculate their non-conformity scores based on the standard non-conformity measure in Equation (3.8).
3. Estimate the class prior probabilities $P(y = 0)$ and $P(y = 1)$ from the training set.
4. Fix ϵ_g and ϵ_0 and compute ϵ_1 based on the Equation (5.5), with respect to the specified condition in (5.6).

5. For each example in the test set, and for each class, compute the p-values using Equation (3.18) for class-conditional Mondrian conformal prediction.

6. For each example in the test set, get its set prediction using Equation (3.11).

Some comments are now in order. From Equation (5.5), it is clear that whenever ϵ_0 is fixed, the value of ϵ_1 increases as ϵ_g grows. Now, if our goal were to maximize our accuracy, we would set $\epsilon_g = \epsilon_0 = \epsilon_1$, as usual. However, in our setting, what is important is to identify a suitable number of persons that will be in debt rather than being too cautious. Hence, our goal for class 1 is to have a reasonable, if not maximal specificity, that is to ensure that a high amount of prediction sets (3.11) will contain only one value for class 1. As the size of Γ^ϵ decreases when ϵ increases, it is then desirable to have a high ϵ_1, hence to pick $\epsilon_g > \epsilon_0$.

5.4 GENERAL APPROACH

Our approach to predicting which tenants are likely to fall into debt followed several steps in order to move from a data warehouse provided by Sopra Steria to a binary prediction (indebted, not indebted) data set based on tenant information. These steps, summarized in Figure 5.1, are:

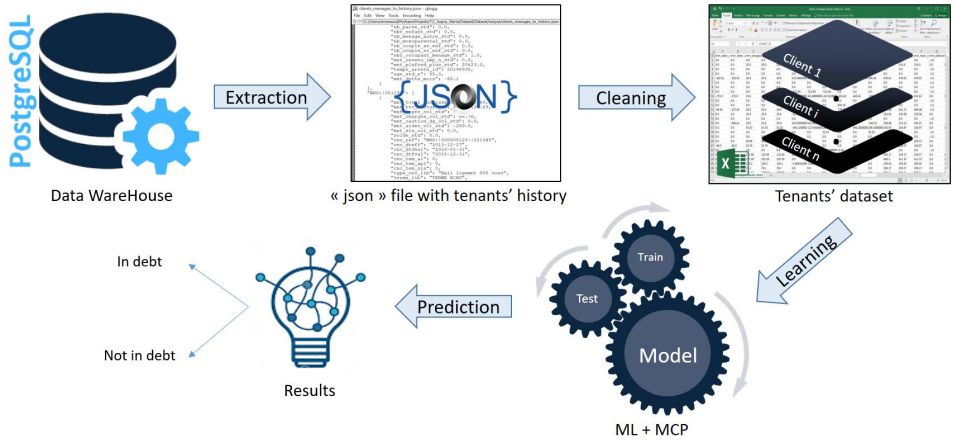


Figure 5.1: The global approach to tenants' debt prediction.

- Data extraction from the DWH database provided by Sopra Steria through the joins between the tables of interest for our problem.

- Data cleaning through transformation, addition, and selection of important variables for our problem in order to obtain a data set exploitable for Machine Learning with characteristics and a target (indebted, not indebted).

- Training the Machine Learning model with a class confidence approach to control the overall error of the model as well as the error of the class of interest (not indebted) to limit costs.

- Prediction of a new tenant as indebted or not indebted by the Machine Learning model and validation of the results on a test set.

5.4.1 *Data extraction and cleaning*

The origin of the data set comes from a data warehouse of one of Sopra Steria's customers that contains monthly historical records of tenant activity from January 2018 to December 2019. This data has been anonymized in accordance with the EU General Data Protection Regulation (GDPR) to protect the tenants' privacy. Figure 5.2 shows some of the relationships between the tables used for our application which were chosen because they contain information about the tenants, their personal situation (age, marital status, ...), their financial situation (job, salary, ...), their rental property (number of rooms, geographical location, ...), and the payment operations related to the rent (rent, bills, amounts collected, ...). In agreemen with experts, we judged that this information was the most interesting and relevant for our problem, i.e. the prediction of tenants likely to fall into a debt situation.

The extraction of the data consisted of the following steps:

1. Read the csv files of the mentioned DWH tables, reformat the missing values, and select the important variables determined by Sopra/Steria (according to the Excel 'table description' file).

2. Prepare the data of the rented element ('elo info') through the joins between the dimension table 'rented element' and the lower hierarchy tables linked to it.

3. Prepare the contract data ('contract info') through joins between the the dimension table 'contract details' and the lower hierarchy tables linked to it.

4. Prepare the household data ('household info') through joins between the fact table 'household' and the lower hierarchy tables linked to it.

5. Prepare the household data ('household detail info') through the joins between the fact table 'household details' and the lower hierarchy tables linked to it.

6. Prepare the data of the client account ('tenant info') through the joins between the fact table 'client account' and the lower hierarchy tables linked to it.

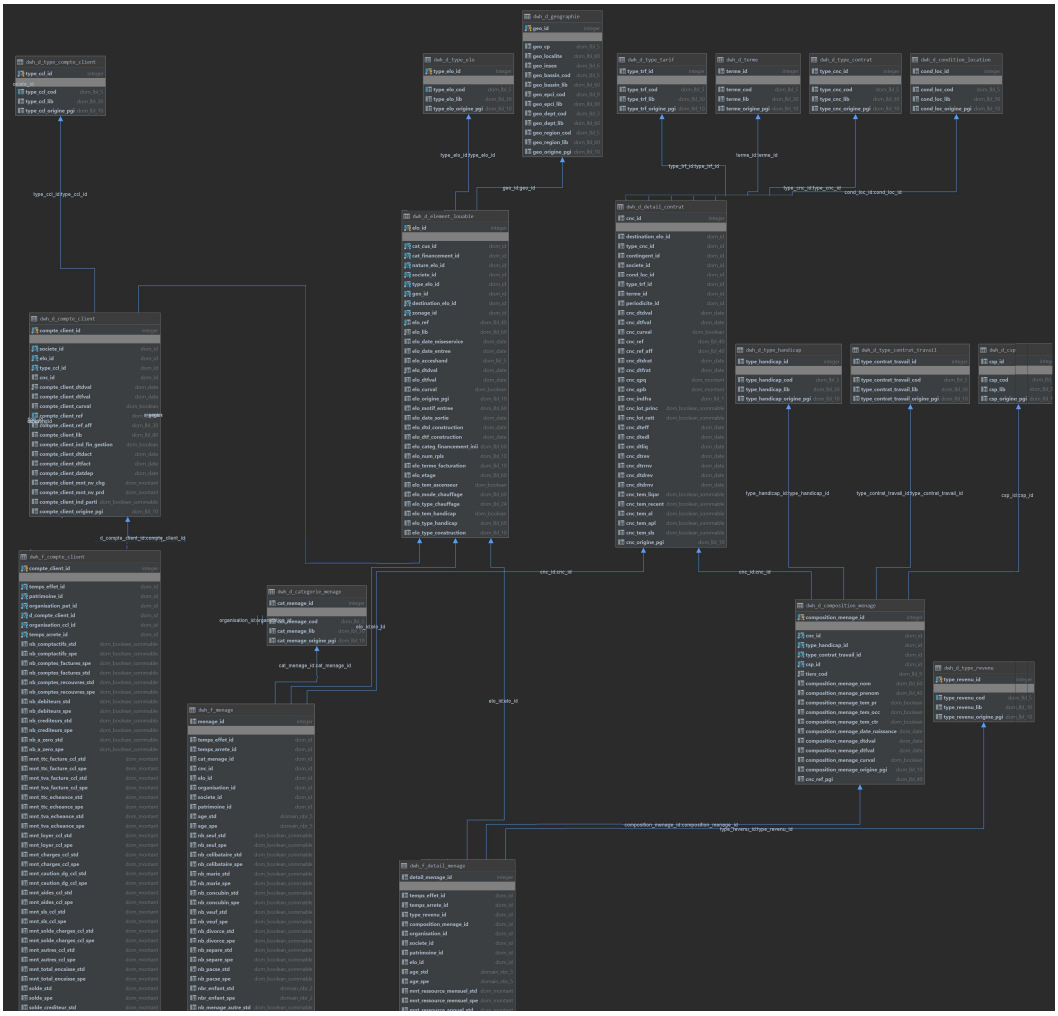


Figure 5.2: Physical Data Model (PDM) of the tables concerned by our application.

7. Merge the data 'household info' and 'household detail info' in 'all household info'.

8. Merge the data 'all household info' and remaining 'info' tables in 'tenants households'.

9. Save the 'tenants households' data set in historical 'json' format.

Figure 5.3 shows the result obtained which is the file 'tenants households to history.json' containing a dictionary of the tenants whose key is the 'tenant account ref' of the tenant and the value is its history (several lines historized according to a 'time id' field).

The steps for preparing the final data set are as follows:

1. Read the file 'tenants household to history.json' in historical format in a dictionary.

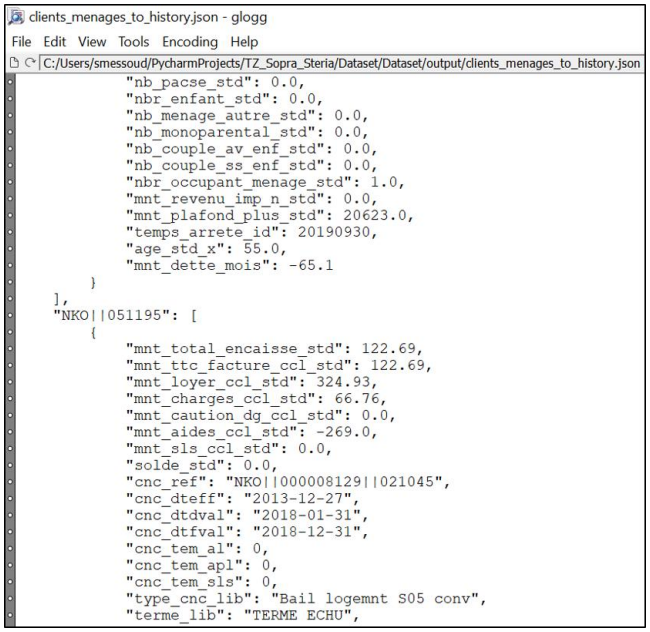


Figure 5.3: Extract from the file 'tenants households to history.json'.

2. Based on the monthly payment transactions, add a new variable calculating the cumulative debt amount, which is a sum of the difference between the collected amount and the invoice amount for each month.

3. Based on the amount of accumulated debt, add a Boolean variable "indebted" which is equal to 0 if the person is not indebted, and 1 if they are indebted for each month. Note that a person is considered indebted if the amount of their cumulative debt is greater than or equal to twice the gross monthly rent.

4. Transform the data set into a tabular format (one row per 'tenant account ref' where each original DWH field whose value varies over time is replaced by n fields corresponding to the number of months of available history).

5. For each 'tenant account ref', check if it has been indebted for the whole duration of the available history. If the person has not been indebted, take a period of three months at random and record the start date of the kept history in a 'hist start' variable. If not, take a period of three months at $m-5$ of the month m of the debt occurrence (to have a latency of one month) and record the start date of the history kept in a variable 'hist start'. Note that for indebted people, and since for each example we take a history of 3 months over a period of almost 2 years, we can have more than one example for each person corresponding to different periods of time, in which this

person can be indebted or not. This extraction strategy was chosen to have a larger data set.

6. Rename all history variables according to the order of the month (1, 2 or 3) in the saved history.

7. Keep only the maximum, minimum or average of variables containing amounts or dates (for example, keep the minimum of 'cnc dtdval', which corresponds to the start date of the validity of a contract).

8. Remove redundant values (for example, keep only one value of 'age std').

9. Delete the business key columns.

10. Delete columns with more than 99.5% missing data and columns with a single value.

11. Save the data set in tabular "csv" format in the file 'tenants households tabular.csv' in the folder 'output'.

Figure 5.4 shows the result obtained which is the file 'tenants households tabular.csv' containing thus a tabular data set with each example (line) representing a client and its history of 3 months, with a target variable 'indebted' (= 1 if indebted and = 0 if not). It indicates whether the tenant was in debt two months after the 3 month history.

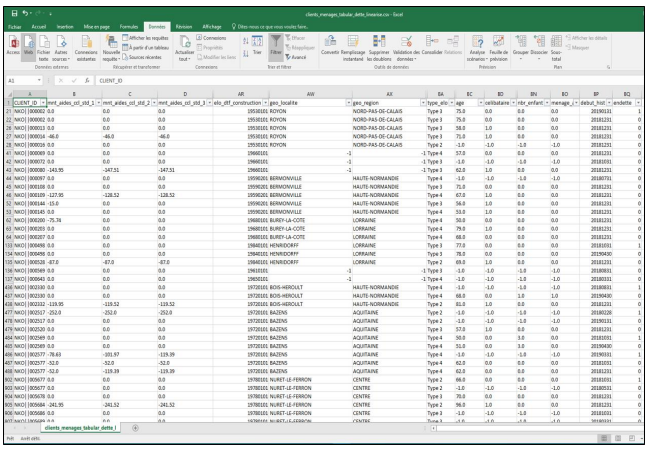


Figure 5.4: Extract from the file 'tenants households tabular.csv'.

The resulting data set contains 28566 examples, 44 variables, and a Boolean class with 1 being in debt and 0 being out of debt. It is also highly imbalanced with only 7.89% of renters having debt and has many missing values due to the fact that some of the data is collected through annual surveys.

5.4.2 *Experimental setting*

In our study, we focused on two main experiments, the first one being a comparison between ICP and MCP with the same ϵ_g, and the second one being the adaptation of class-conditional MCP to control ϵ_g and ϵ_0 values.

We chose the gradient boosting algorithm "LightGBM" as the underlying algorithm since it can handle missing values and both categorical and numerical features. We used the standard non-conformity measure $f(z) = 1 - \hat{P}_h[y \mid x]$. We also tried three types of splitting for the data set, with a test set equal to 20% of the data set, and with a calibration set equal to 20% of the training examples. These splitting approaches are as follows:

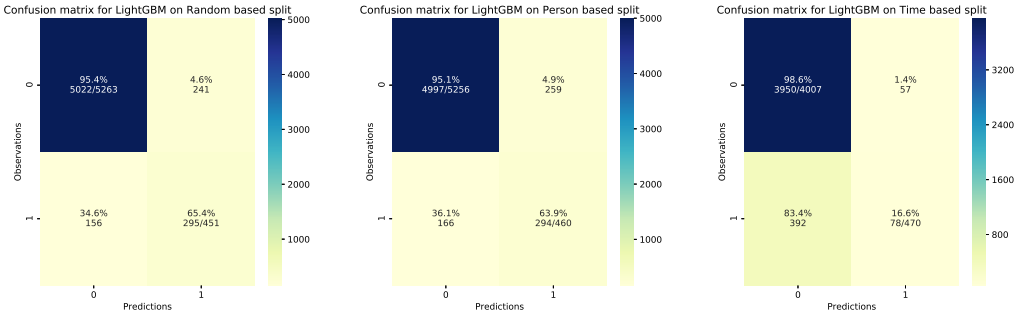
- **Random:** examples are split randomly on all training, calibration and test sets. From the application perspective, this is clearly unrealistic, as future events will be used to predict past ones (which is in practice impossible), and as the same tenant will possibly be in different sets, thus making unclear whether exchangeability holds. However, we would argue that this is true for most actual applications, where a task typically involves predicting future observations from past ones, and this does not stop most ML (including those on conformal prediction) papers to consider random splits on benchmarks to validate approaches. Since this is the standard splitting strategy that is used in all classic benchmarks, a random split therefore seems a good point to provide a proof-of-concept.

- **Person:** examples are split according to the tenant's ID, since we can have many examples for the same person. This means that the examples in the test set are those of people the algorithm did not see before in the training and calibration phases. This seems a more realistic scenario, as in practice tenants used in the training phase will often be past tenants, and tenants used in the operational phase will be new ones, different from the past ones.

- **Time:** the test predictions are done on examples from the future based on a training and calibration done on examples from the past. This is done by using randomised 2018 data for the training and calibration, and keeping 2019 data for the test based on the feature "history start", so that the test set was made of approximately 20% of the data set. This is undoubtedly the most realistic splitting strategy with respect to the application goal, however as we shall see this is also the one in which standard conformal prediction does not work very well, for potential reasons we will discuss.

For comparison purposes, Figure 5.5 shows the confusion matrix results for all splitting protocols using the underlying algorithm "LightGBM" with a threshold of 0.3 (so as to settle a low percentage of tenants wrongly

predicted as defaulting). This first figure shows how badly the underlying algorithm performs when it comes to handling imbalanced data, especially in the case of time-based splitting.



(a) Random **(b)** Person **(c)** Time

Figure 5.5: Confusion matrices for the underlying algorithm for all splitting protocols.

The first experiment was conducted based on the steps for the inductive conformal classifier as described in Section 3.4, for values of ϵ ranging from 0.1 to 0.9, where $\epsilon_g = \epsilon_0 = \epsilon_1$ in the case of MCP.

The second experiment was conducted by adapting class-conditional MCP to specify the error rate of the class 0 and the overall error rate, in order to limit the number of misclassified people that are predicted as in debt when in fact they are not. To do so, we followed our class-wise confidence approach as described in Section 5.3, with different ϵ_g and ϵ_0 values.

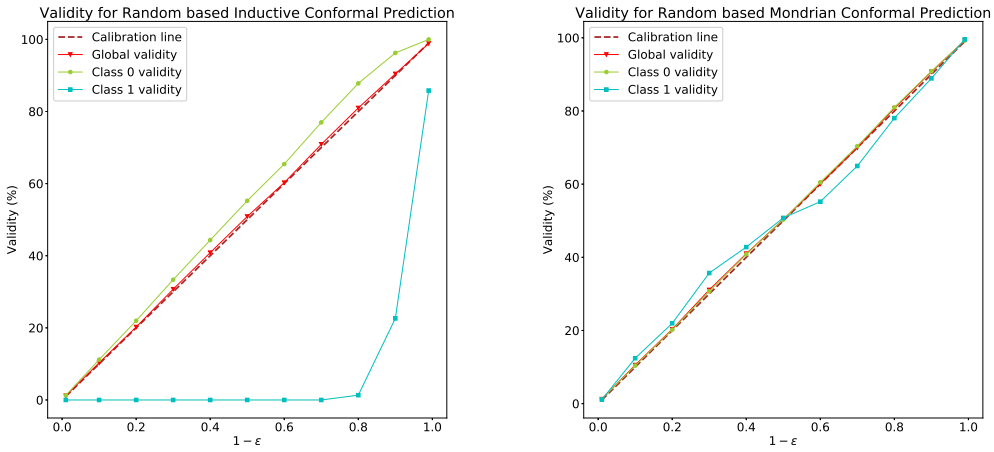
5.4.3 *What did work*

This section presents the promising results of our experiments, investigating in particular the difference between ICP and MCP, and the validity and efficiency of the proposed approach.

For our first experiment, and to verify the validity of ICP and MCP, we calculate the global accuracy and the accuracy of each class, and compare them with the calibration line. This line represents the case where the error rate is exactly equal to ϵ for a confidence level $1 - \epsilon$, which is what we seek to obtain in an exactly valid conformal predictor. Results are shown in Figures 5.6 and 5.7 for each Random and Person types of splitting.

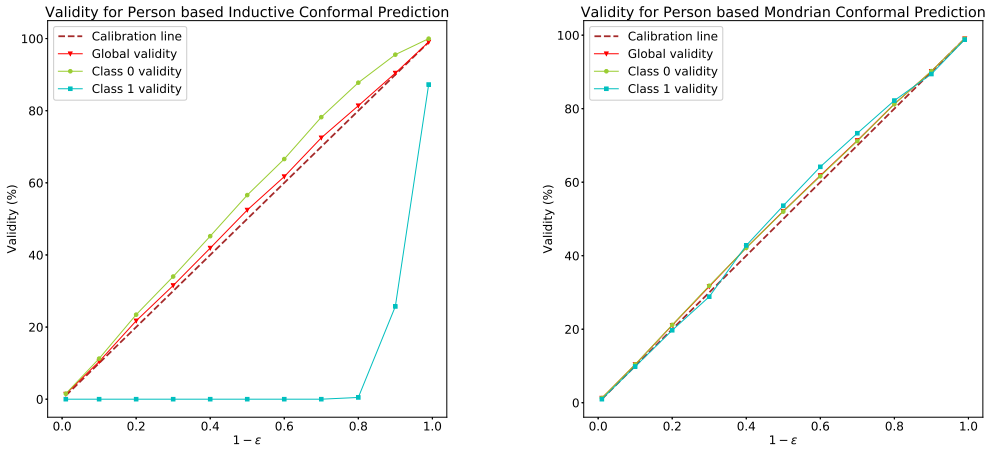
In the case of ICP (Figures 5.6a and 5.7a), results show that the global validity for all the data set is reached. However, it is not respected for the classes, more importantly for the class 1 corresponding to in debt tenants, since we have fewer examples for this class. This shows the problem of having an imbalanced data set in terms of categories or classes, and how the least represented area of the observations' space suffers the most

when a simple conformal prediction method is used. Indeed, a very bad validity for the minority class can be compensated by a slightly conservative validity for the majority class. This problem is resolved in the case of MCP (Figures 5.6b and 5.7b), which gives better validity results that are almost exactly valid for the global data set and also for each individual class, including the minority class 1. In our case, a person-based split slightly outperforms a random-based split since this latter is invalid for the minority class 1 at ϵ values between 0.1 and 0.5.



(a) ICP validity **(b)** MCP validity

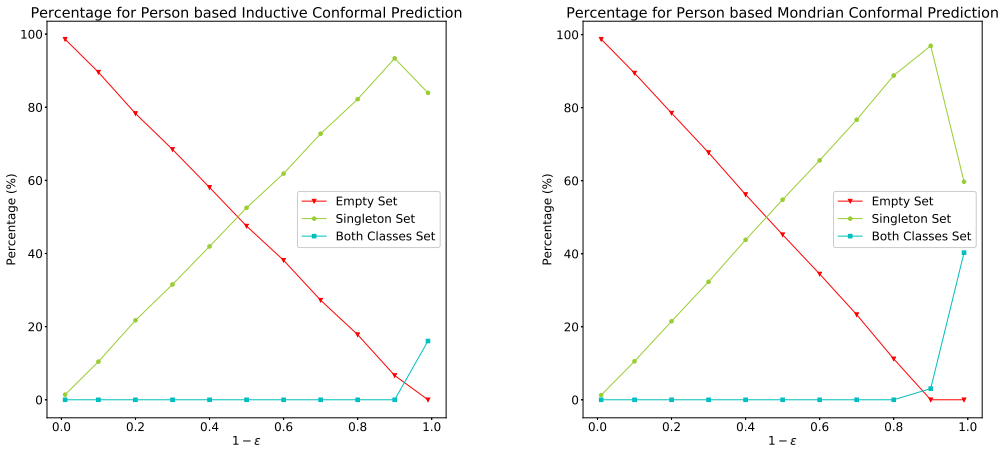
Figure 5.6: Validity results for Random-based split.



(a) ICP validity **(b)** MCP validity

Figure 5.7: Validity results for Person-based split.

To evaluate the efficiency of ICP and MCP, we calculated the percentage of singletons, empty sets $\emptyset$ and $\{0, 1\}$ sets from all the predictions of the test examples. Results are shown in Figure 5.8 for person-based splits, and are similar for the random-based split.



(a) ICP efficiency

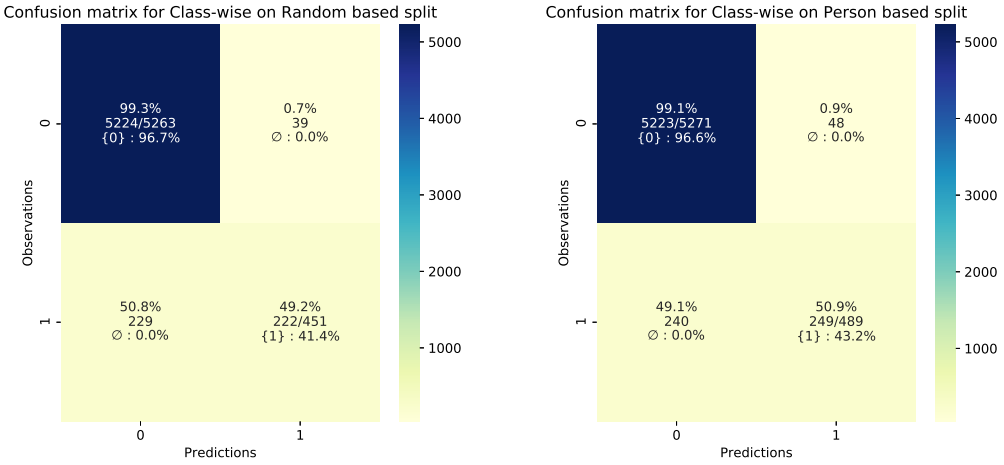
(b) MCP efficiency

Figure 5.8: Percentage results for Person-based split.

For efficiency results, it is noticed that as ϵ decreases, the percentage of predicted empty sets $\emptyset$ lessens. It is even no longer predicted (at $\epsilon = 0.1$ for MCP). Conversely, the opposite is observed with the percentage of singleton sets which grows constantly as ϵ decreases until $\epsilon = 0.1$ for ICP and $\epsilon = 0.2$ for MCP. From that moment, we notice a mirror effect between the percentage of singletons and the percentage of $\{0, 1\}$ sets, which was null until now, and that grows whereas the percentage of singletons declines, with bigger values in the case of MCP. This can be explained by the fact that in MCP, the confidence level is guaranteed for each class as well as the global data set, meaning that the model predicts more $\{0, 1\}$ sets to have a more reliable prediction, even for the minority class. The same observations are made for both splitting methods.

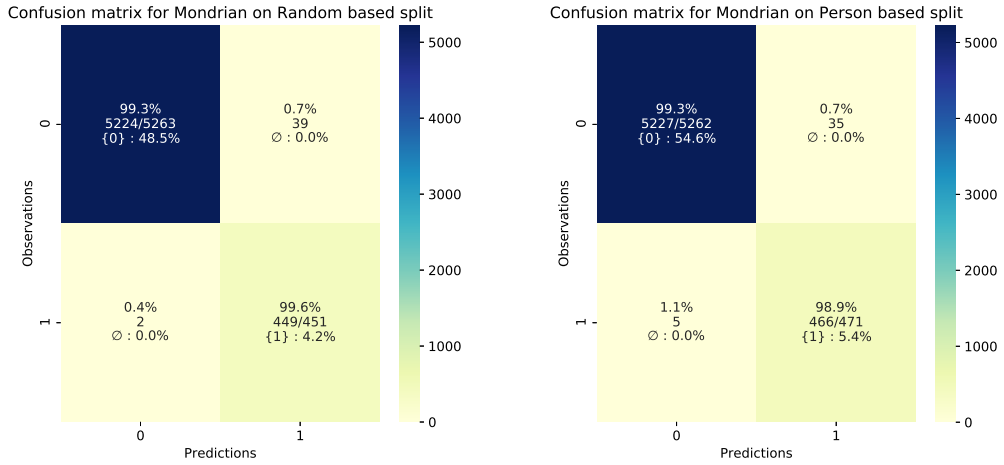
For the second experiment, we used our class-wise confidence approach and specified different values for the significance levels ϵ_g and ϵ_0. Figures 5.9 and 5.10 show the confusion matrix for both random and person splits with $\epsilon_g = 0.05$ and $\epsilon_0 = 0.01$ and with $\epsilon_g = \epsilon_0 = 0.01$, corresponding to a classical MCP classifier.

In addition to having the percentage and amount of data falling in each cell, we have added an extra element, which is either the proportion of singleton predictions in the case of correct predictions (that is, the ratio of $\{0\}$ or $\{1\}$ among the predictions), or the proportion of empty set predictions in case of incorrect predictions.



(a) Random-based split **(b)** Person-based split

Figure 5.9: Confusion matrix for class-wise confidence with $\epsilon_g = 0.05$ and $\epsilon_0 = 0.01$.



(a) Random-based split **(b)** Person-based split

Figure 5.10: Confusion matrix for class-wise confidence with $\epsilon_g = \epsilon_0 = 0.01$.

For our approach, Figure 5.9 shows that the error rate for class 0 is approximately equal to the chosen ϵ_0, and that the error rate for class 1 is also approximately 0.52, the result of Equation (5.5) when $\epsilon_0 = 0.01$ and $\epsilon_g = 0.05$. Also, the global validity is approximately equal to 95% as it was chosen by the social property owners, which shows that our method achieves a class-wise confidence while keeping a global confidence, both chosen by the user. Similarly, Figure 5.10 shows expected results for the more classical choice $\epsilon_g = \epsilon_0$. However, a striking difference between the

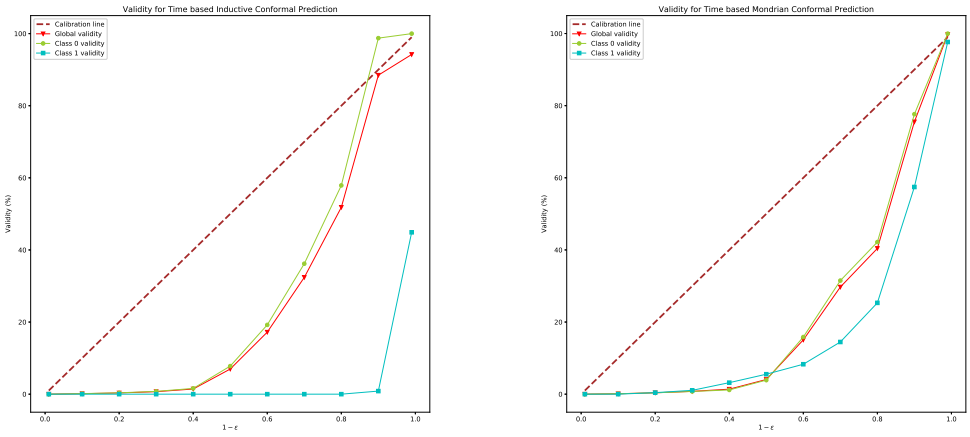
two is the number of precisely recognized persons that will be in debt or not in debt, i.e. the percentage of singletons. For in debt persons, in the matrices of Figure 5.10, this amounts to 19 persons for the random-based split (4.2% of 449 persons) and 25 persons for the person-based split (5.4% of 466 persons), while in the matrices of Figure 5.9, this amounts to 91 persons for the random-based split (41.4% of 222 persons) and 108 persons for the person-based split (43.2% of 249 persons). So indeed our accuracy on the first class has dropped drastically in our scheme, even with a small margin between ϵ_g and ϵ_0, but the clear upside is that we were able to detect much more (about four times more) problematic tenants, allowing for more prevention. For people who are not in debt, 2534 persons are precisely predicted in the case of classical MCP (Figure 5.10) for the random-based split (48.5% of 5224 persons) and 2854 persons for the person-based split (54.6% of 5227 persons), whereas with our class-wise approach (Figure 5.9), this amounts to 5052 persons for the random-based split (96.7% of 5224 persons) and 5056 persons for the person-based split (96.6% of 5233 persons), meaning that experts will have much less $\{0, 1\}$ cases to verify manually when using our method. This can be explained by the fact that, contrary to our approach, all ϵ values are equal in the classical MCP, thus $\epsilon_1 = 0.01$, which leads to more $\{0, 1\}$ sets predicted in order to ensure the 99% validity for the minority class. Consequently, this impacts the class 0, by decreasing the percentage of precisely predicted non in debt persons. Again, it is useful to recall that in this application, tenants that will be in debt and predicted as not in debt will anyway be detected and helped if possible. In Appendix C, Tables C.1 and C.2, we provide the full results obtained for various choices of ϵ_g and ϵ_0, in the case of random and person splits.

As a conclusion, the first feedback from Sopra Steria and an involved customer was enthusiastic and showed the advantages of using our class-wise approach, especially when it comes to the absolute number of well-verified tenants, a more important indicator for our data provider than the ratio. Indeed, using the person splitting strategy, it is possible to reserve an amount of tenants' data for training and calibration (for instance, tenants that are no longer customers of the social property owner), and only predict in the production phase on new never-seen-before tenants. Our empirical results also tend to indicate that the common assumption of conformal prediction (variable exchangeability, i.e., the fact that the drawn inferences are invariable under observation permutation) holds at least approximately in those cases.

They should also be compared to the confusion matrices of Figure 5.5, where the number of tenants falsely predicted as defaulting is as high as the number of tenants rightly predicted as defaulting, both for the random and person splits. This also shows that the conformal approaches can bring a significant edge when compared to standard methods.

5.4.4 *What did not work*

This section presents the results of our experiments for the time splitting strategy. For our first experiment, we verify the validity of ICP and MCP for the time splitting strategy by comparing the global accuracy and the accuracy of each class with the calibration line. Results are shown in Figure 5.11.

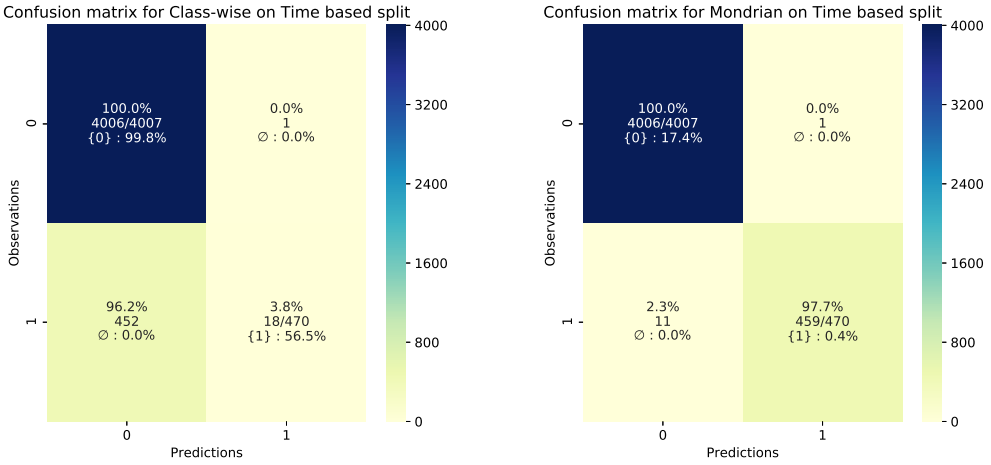


(a) ICP validity **(b)** MCP validity

Figure 5.11: Validity results for Time-based split.

In both cases, we notice that both methods are performing actually very badly, with the global and class accuracies below the calibration line. We also observe slightly better results for the MCP when it comes to the minority class 1, with its accuracy being closer to the global and majority class ones as compared to the ICP. We will discuss the possible reasons behind such a result in the next section.

For the second experiment, Figure 5.12 shows the confusion matrix of our class-wise approach for time split with $\epsilon_g = 0.05$ and $\epsilon_0 = 0.01$ and with $\epsilon_g = \epsilon_0 = 0.01$, corresponding to a Mondrian approach. These results go together with the results of our first experiment, showing that the chosen ϵ values are not respected for the class-wise approach with an actual global accuracy of 89.88% instead of 95% and an actual accuracy of 3.83% instead of the computed 45% for class 0 (c.f. Appendix C Table C.3 for full results). When closely examining the results, we also observe that for MCP (Figure 5.12b), the percentage of singletons is very small for all of the confusion matrix cells, meaning that the experts should manually verify nearly all examples. The only case where the amount of singleton is high (Figure 5.12a) is when most predictions belong to the class 0, in which case they are totally uninformative (hence useless).



(a) $\epsilon_g = 0.05$ and $\epsilon_0 = 0.01$ **(b)** $\epsilon_g = \epsilon_0 = 0.01$

Figure 5.12: Confusion matrix for class-wise confidence with a Time-based split.

As a conclusion, the time-based strategy, the splitting protocol that uses the temporal aspect of the data set and is the most consistent with our final applied intent, did not work. We discuss the possible fixes to that in the next section.

5.4.5 *What's next*

Overall, some of our experiments have shown that conformal approaches can be used to finely control class-wise defined errors, the need of which may easily arise in applications where some type of errors should be controlled finely, while other ones are deemed less important.

However, not everything worked, and even in those cases where it worked, there are probably still some room for improvement:

- **Time-based split:** in this case, the lack of an exact validity could be explained by the fact that the time splitting strategy violates the exchangeability assumption, possibly by a change of conditions not detected and not present in the data that changes the underlying distribution. Discussions with our end-user (Sopra Steria) did not make us able to identify a possible source for such a change, as the data set is a real, but outdated one. One possibility to solve this issue would be to explore, use and/or adapt conformal approaches able to deal with (temporal) distribution shift in the test set [Gibbs et al., 2021].

- **Better control of the error to satisfy the end-user:** what Mondrian conformal prediction can provide is a control of the class-conditional

error, that is, knowing that $y = c$, we have validity on the probability $P(y \in \hat{Y} \mid y = c)$ where $\hat{Y}$ is our conformal prediction. However, what the user is ultimately interested in is the control of $P(y = c \mid \hat{Y} = \{c\})$, of which our approach only provides a proxy (deemed satisfying by the user). Our next step would therefore be to search to control $P(y = c \mid \hat{Y} = \{c\})$, possibly by combining different conformal predictors (e.g., Class-wise with global ones), or by using Venn-Abers predictors [Vovk et al., 2014] that are able to provide calibrated probabilities as outputs, with the caveats that they provide multiple values for those, of which at least one is ensured to be calibrated.

5.5 CONCLUSION

In this work, we used the conformal prediction, and in particular the class-conditional Mondrian variant, to have more control on the error rate of a certain class, while preserving an overall confidence. We applied our method to the real estate domain in order to help social property owners limit the number of people that are falsely predicted as in debt when, in fact, they are not, as these misclassifications are costly. The results show the interest of this method on our data set when two types of splits are considered (random and person based), with less-satisfying results when using the time-based split.

Chapter 6

Conclusion and perspectives

In this thesis, the objective was to study distribution-free uncertainty quantification via conformal prediction and apply it on complex data with two main research directions.

The first study was focused on multi-target regression, a multi-task learning problem that was scarcely explored in the conformal prediction field. The idea was to provide a global confidence level on all targets at once instead of treating them individually. To do so, we first used a naive approach to get a hyper-rectangle conformal region with a Bonferroni-like correction that produces over-conservatively valid non-conformity measures. We then solved this issue by proposing a method that exploits the dependency structure between non-conformity scores using copulas, which improves the performance of non-conformity scores (for both validity and efficiency) compared to the independence assumption as supposed previously. Finally, we presented a more flexible approach that gives an ellipsoidal conformal region by using the local covariance matrix of each example. This last method gives tighter volumes while maintaining the required validity.

The second direction was an applied research work to control the errors of a binary classification problem related to the tenants' debt prediction, with the guidance of real estate experts. This was achieved by using class-conditional Mondrian conformal prediction and showed the benefits of applying such a method, which is limiting the number of people that are falsely predicted as in debt. However, it is still lacking when we consider a time-based approach when splitting data.

For multi-target regression, perspectives include exploring further the flexibility of our methods, for instance by adapting them to the richer conformal predictive distributions [Vovk et al., 2017], by exploring the possibility of using vines [Joe et al., 2011] to model complex dependencies, or by proposing protocols allowing to obtain ϵ_g from different individual, user-defined confidence degrees.

Another future study would be to explore other ways to estimate the local covariance matrix in the ellipsoidal conformal approach, for example

by taking into consideration a density estimation by performing element-wise prediction. We would also like to adapt our method to other uncertainty ones such as jack-knife+ [Barber et al., 2021] and its leave-one-out approach that allows to use all data in the training, especially for small data sets.

We also would like to directly learn a cost function that takes into consideration validity and efficiency [Colombo et al., 2020] for a multi-target regression problem, possibly by using the hyper-rectangle volume as a parameter to define ϵ_t values that give us the smallest volume for the same validity.

Finally, while we mostly focused on multi-variate regression in this thesis, it would be interesting to try to extend the current approach to other multi-task settings, such as multi-label problems. A possibility could be to make such problems continuous, as proposed for instance by [Liu, 2019].

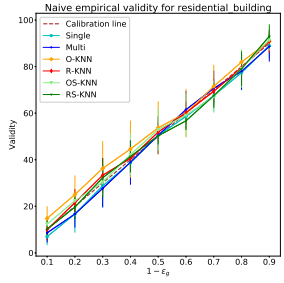
For our applied research on tenants' debt prediction, our contribution is rather modest from a methodological perspective, but it opens up questions about a problem we think is important: the one of identifying how various error rates should be elicited/chosen when considering conformal frameworks with such multiple error rates. Indeed, while choosing equal error rates is common in systematic studies, it can hardly be expected in real-life applications that all targets need the same accuracy.

From an application point of view, the perspectives include treating missing values to improve classification results, as for the moment we use a model that handles them naturally. Also, we would like to explore other non-conformity measures other than the standard one used in this work. Moreover, it would be interesting to work on the time-based split, by applying conformal prediction methods that treat temporal data such as time series [Chernozhukov et al., 2018], or by considering the combination of conformal approaches with robust approaches to the distribution shift [Tibshirani et al., 2019] in order to obtain validity even when the data generating distribution varies over time.

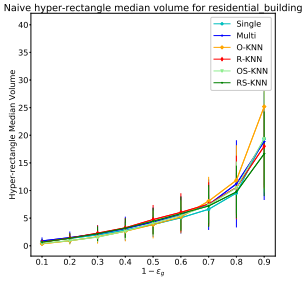
Appendix A

Complementary experimental results of naive conformal MTR

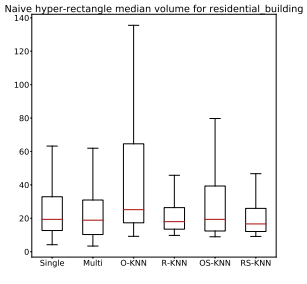
This appendix presents complementary validity and efficiency results of the remaining data sets. Note that for data sets with more than four targets, we use a logarithmic scale to plot the median volume.



(a) Empirical validity

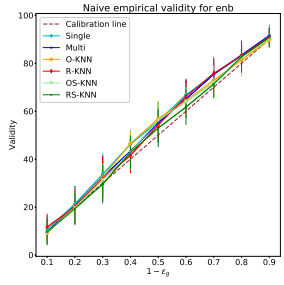


(b) Empirical efficiency

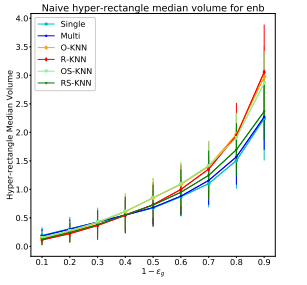


(c) Volume for $\epsilon_g = 0.1$

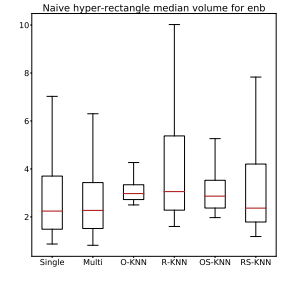
Figure A.1: Naive conformal results for "res building".



(a) Empirical validity

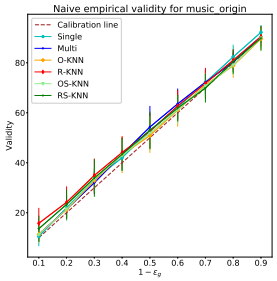


(b) Empirical efficiency

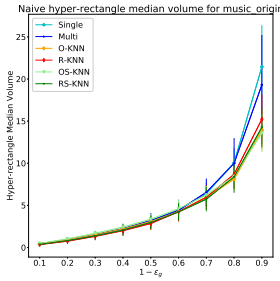


(c) Volume for $\epsilon_g = 0.1$

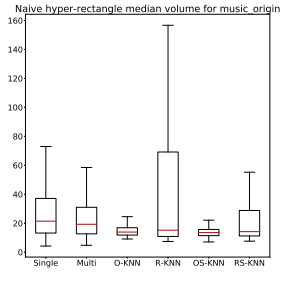
Figure A.2: Naive conformal results for "enb".



(a) Empirical validity

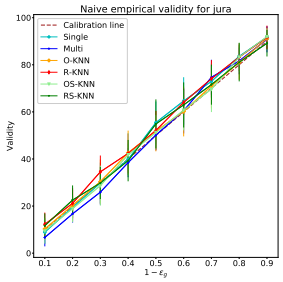


(b) Empirical efficiency

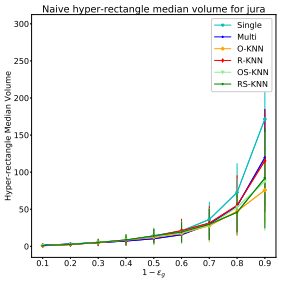


(c) Volume for $\epsilon_g = 0.1$

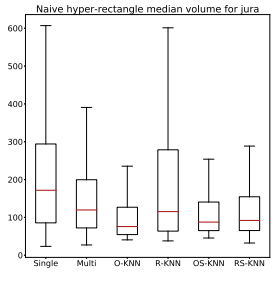
Figure A.3: Naive conformal results for "music origin".



(a) Empirical validity

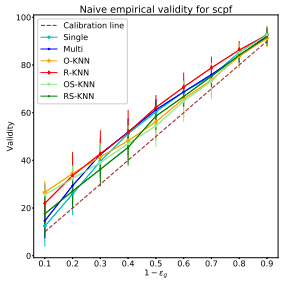


(b) Empirical efficiency

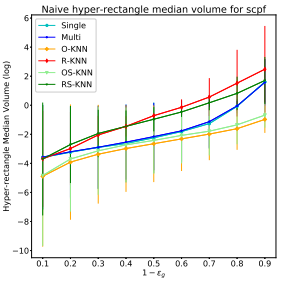


(c) Volume for $\epsilon_g = 0.1$

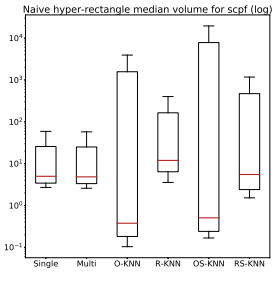
Figure A.4: Naive conformal results for "jura".



(a) Empirical validity

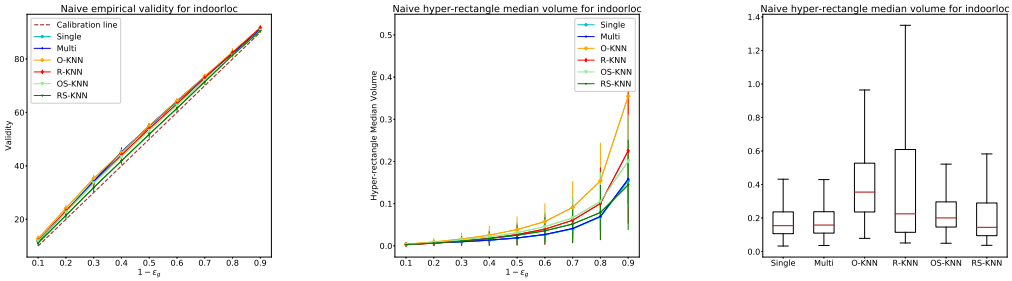


(b) Empirical efficiency



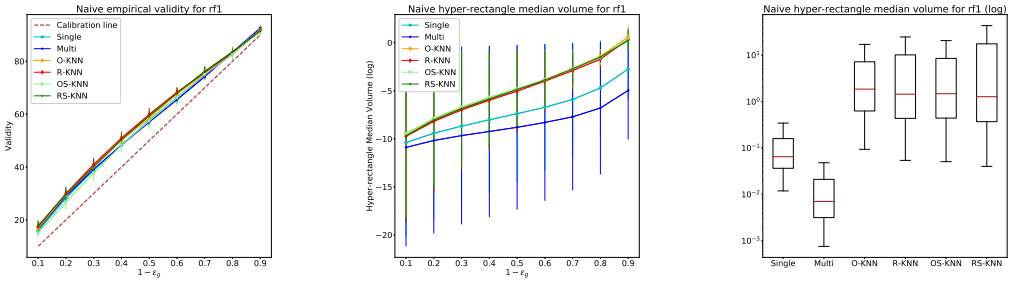
(c) Volume for $\epsilon_g = 0.1$

Figure A.5: Naive conformal results for "scpf".



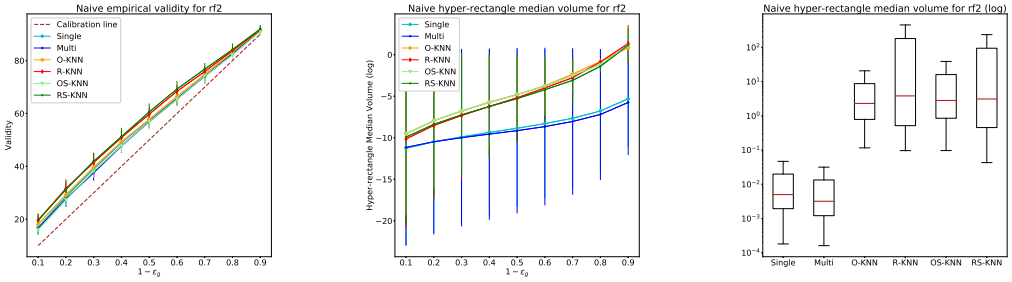
(a) Empirical validity **(b)** Empirical efficiency **(c)** Volume for $\epsilon_g = 0.1$

Figure A.6: Naive conformal results for "indoorloc".



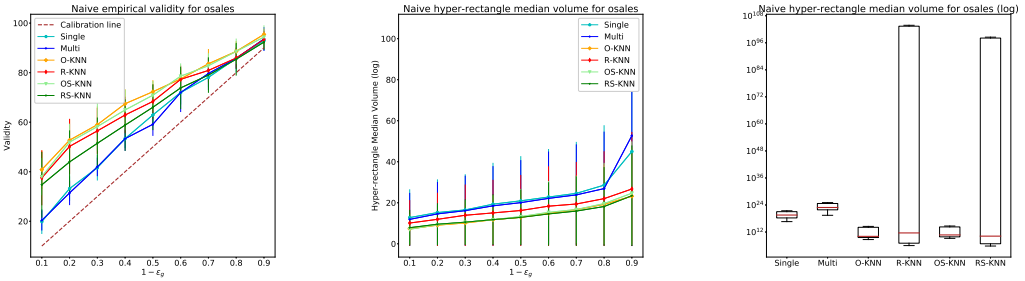
(a) Empirical validity **(b)** Empirical efficiency **(c)** Volume for $\epsilon_g = 0.1$

Figure A.7: Naive conformal results for "rf1".



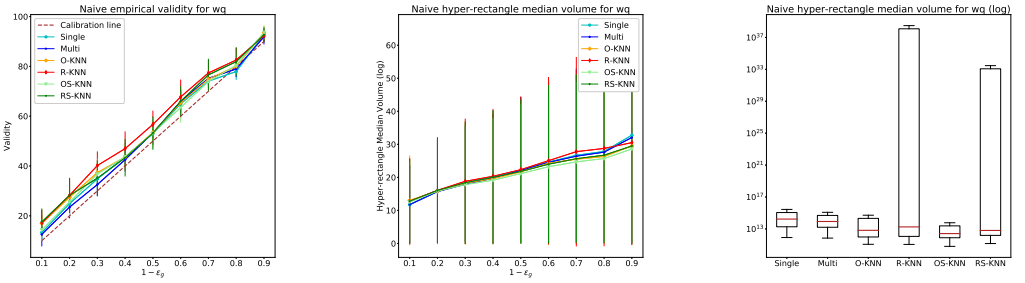
(a) Empirical validity **(b)** Empirical efficiency **(c)** Volume for $\epsilon_g = 0.1$

Figure A.8: Naive conformal results for "rf2".



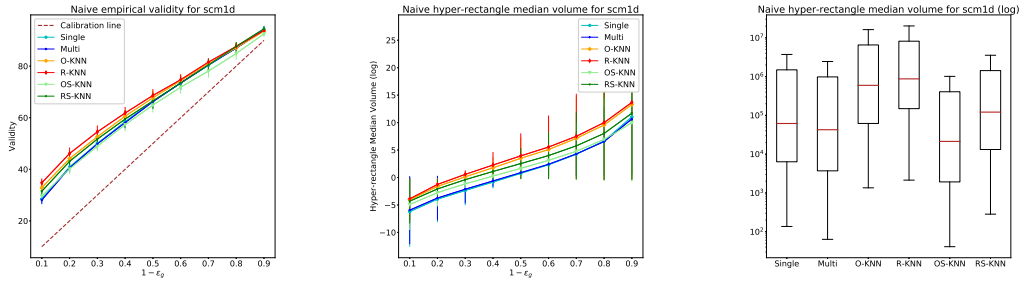
(a) Empirical validity **(b)** Empirical efficiency **(c)** Volume for $\epsilon_g = 0.1$

Figure A.9: Naive conformal results for "osales".



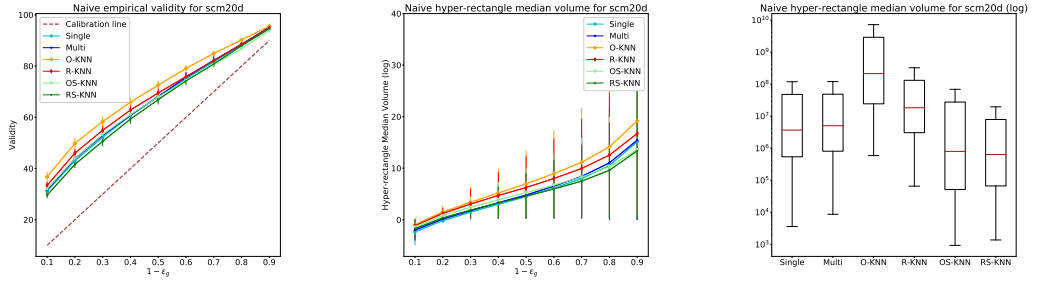
(a) Empirical validity **(b)** Empirical efficiency **(c)** Volume for $\epsilon_g = 0.1$

Figure A.10: Naive conformal results for "wq".



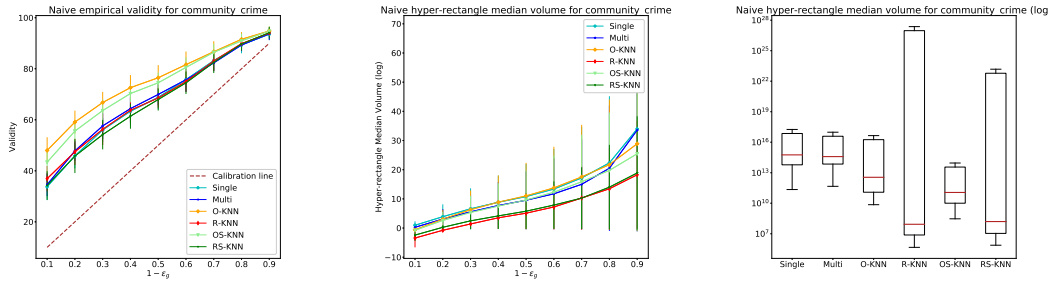
(a) Empirical validity **(b)** Empirical efficiency **(c)** Volume for $\epsilon_g = 0.1$

Figure A.11: Naive conformal results for "scm1d".



(a) Empirical validity **(b)** Empirical efficiency **(c)** Volume for $\epsilon_g = 0.1$

Figure A.12: Naive conformal results for "scm20d".



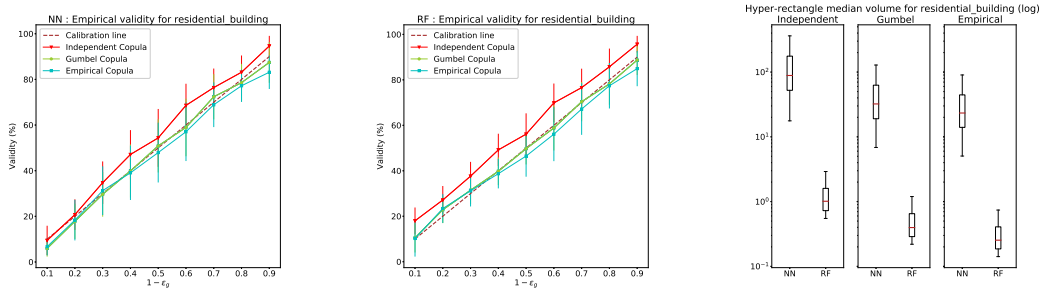
(a) Empirical validity **(b)** Empirical efficiency **(c)** Volume for $\epsilon_g = 0.1$

Figure A.13: Naive conformal results for "com crime".

Appendix B

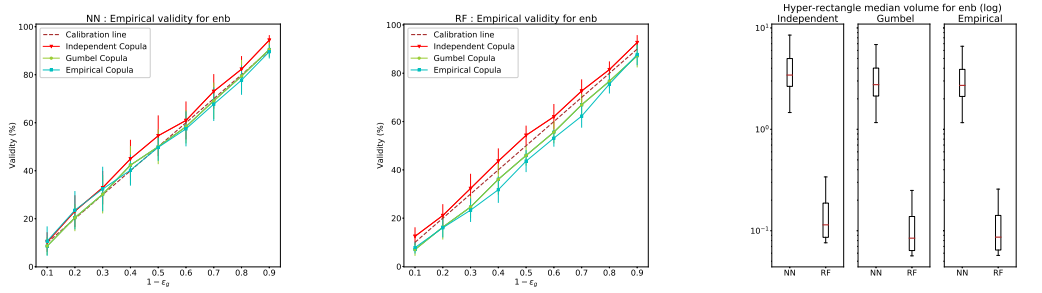
Complementary experimental results of copula conformal MTR

This appendix presents complementary validity and efficiency results of the remaining data sets.



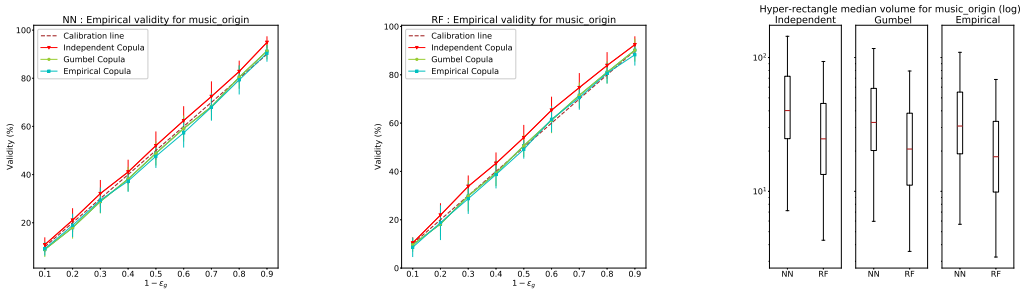
(a) Validity (NN) **(b)** Validity (RF) **(c)** Efficiency (both)

Figure B.1: Copula conformal results for "res building".



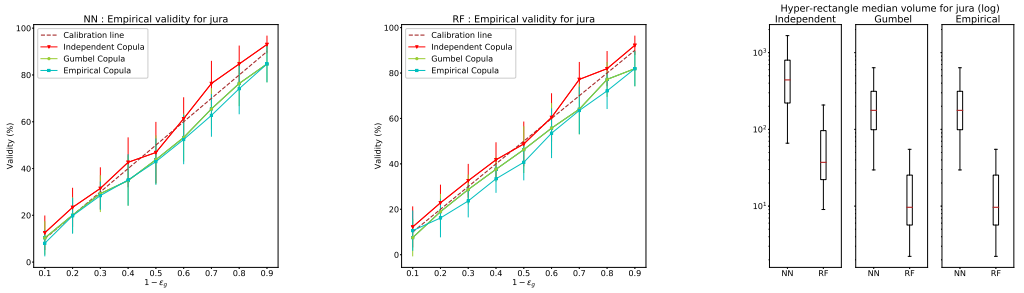
(a) Validity (NN) **(b)** Validity (RF) **(c)** Efficiency (both)

Figure B.2: Copula conformal results for "enb".



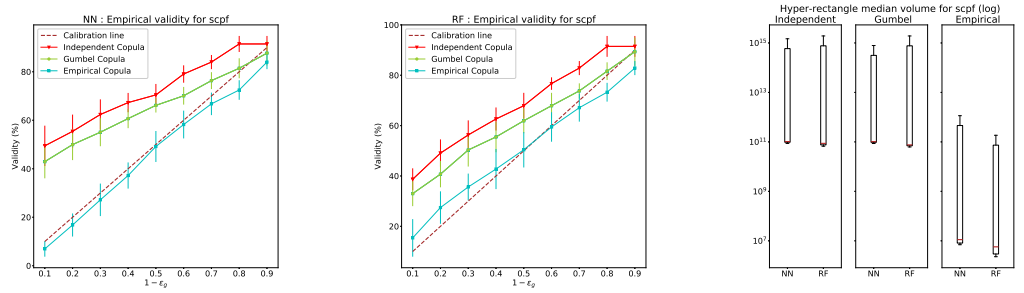
(a) Validity (NN) **(b)** Validity (RF) **(c)** Efficiency (both)

Figure B.3: Copula conformal results for "music origin".



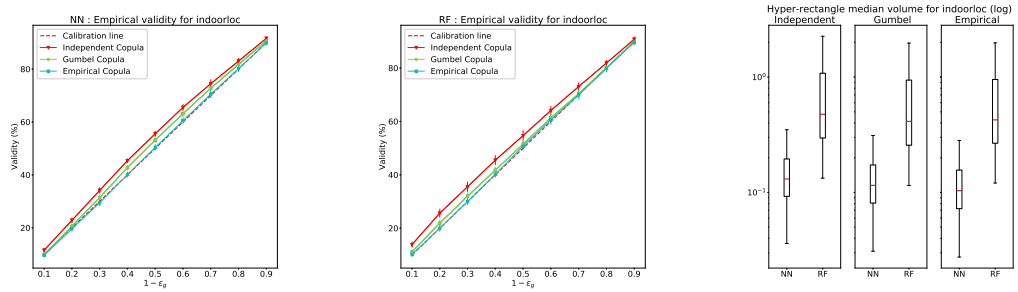
(a) Validity (NN) **(b)** Validity (RF) **(c)** Efficiency (both)

Figure B.4: Copula conformal results for "jura".



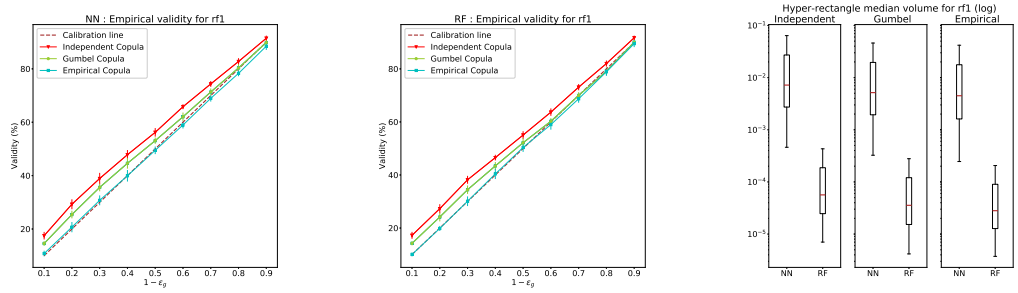
(a) Validity (NN) **(b)** Validity (RF) **(c)** Efficiency (both)

Figure B.5: Copula conformal results for "scpf".



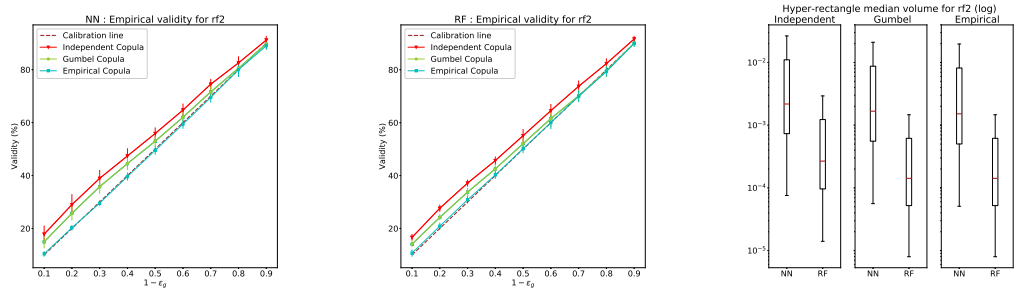
(a) Validity (NN) **(b)** Validity (RF) **(c)** Efficiency (both)

Figure B.6: Copula conformal results for "indoorloc".



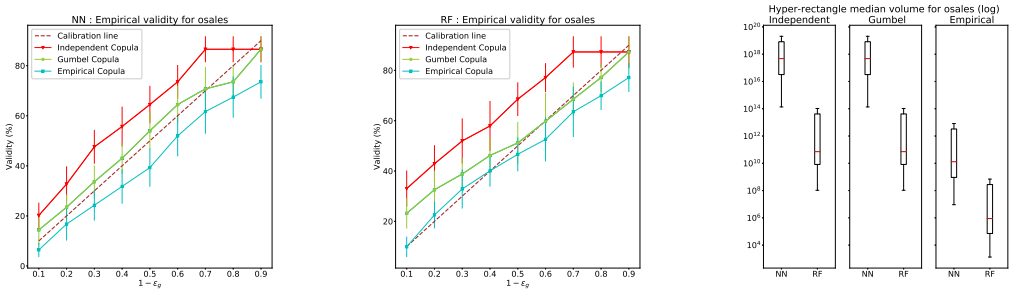
(a) Validity (NN) **(b)** Validity (RF) **(c)** Efficiency (both)

Figure B.7: Copula conformal results for "rf1".



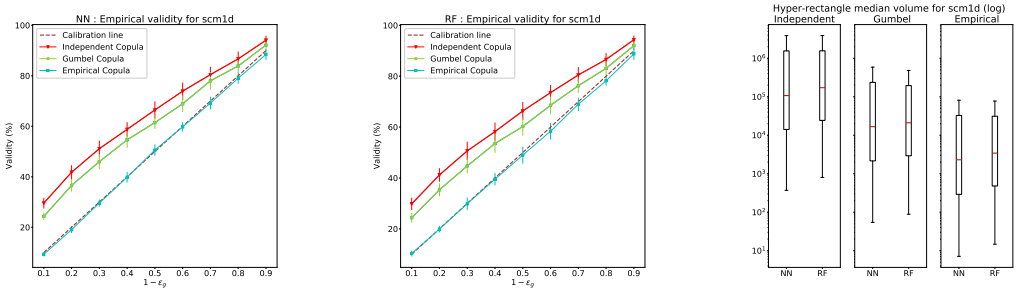
(a) Validity (NN) **(b)** Validity (RF) **(c)** Efficiency (both)

Figure B.8: Copula conformal results for "rf2".



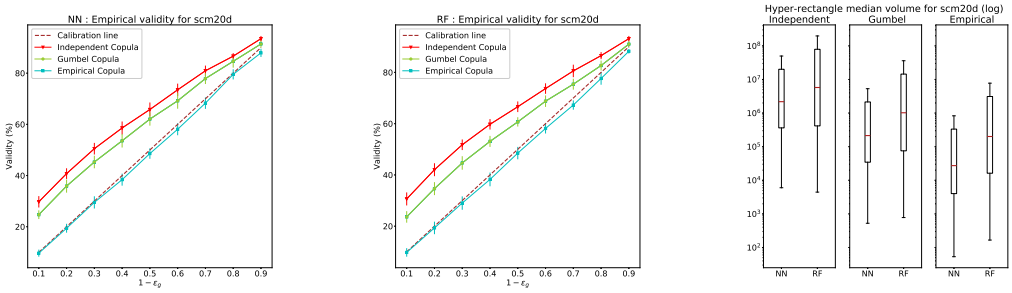
(a) Validity (NN) **(b)** Validity (RF) **(c)** Efficiency (both)

Figure B.9: Copula conformal results for "osales".



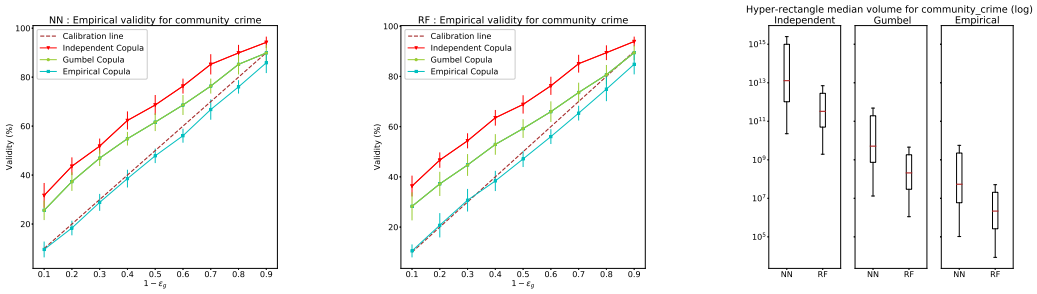
(a) Validity (NN) **(b)** Validity (RF) **(c)** Efficiency (both)

Figure B.10: Copula conformal results for "scm1d".



(a) Validity (NN) **(b)** Validity (RF) **(c)** Efficiency (both)

Figure B.11: Copula conformal results for "scm20d".



(a) Validity (NN) **(b)** Validity (RF) **(c)** Efficiency (both)

Figure B.12: Copula conformal results for “com crime”.

Appendix C

Additional results of MCP for tenants' debt prediction

Tables C.1, C.2 and C.3 present a complete overview of our results for different choices of ϵ_0 and ϵ_g for all splitting strategies.

Epsilons			Accuracy (%)			{0, 1} sets (%)		∅ sets (%)	
ϵ_g (chosen)	ϵ_0 (chosen)	ϵ_1 (computed)	Global	Class 0	Class 1	Class 0	Class 1	Class 0	Class 1
0.01	0.01	0.01	99.28	99.26	99.56	51.5	95.81	0.0	0.0
0.02	0.01	0.14	98.04	99.26	83.81	14.15	86.11	0.0	0.0
0.03	0.01	0.26	97.22	99.26	73.39	8.91	79.47	0.0	0.0
0.04	0.01	0.39	95.89	99.26	56.54	4.95	67.95	0.0	0.0
0.05	0.01	0.52	95.31	99.26	49.22	3.33	58.61	0.0	0.0
0.06	0.01	0.64	94.56	99.26	39.69	1.91	44.69	0.0	0.0
0.07	0.01	0.77	93.45	99.26	25.72	0.0	0.0	5.13	2.69
0.08	0.01	0.9	92.4	99.26	12.42	0.0	0.0	69.23	17.47
0.05	0.05	0.05	95.17	95.15	95.34	18.7	76.33	0.0	0.0
0.06	0.05	0.18	93.93	95.15	79.6	5.39	47.81	0.0	0.0
0.07	0.05	0.3	92.72	95.15	64.3	0.0	0.0	10.2	4.97
0.08	0.05	0.43	91.9	95.15	53.88	0.0	0.0	39.22	26.44
0.09	0.05	0.56	91.23	95.15	45.45	0.0	0.0	60.0	37.8
0.1	0.05	0.68	90.58	95.15	37.25	0.0	0.0	73.73	45.94
0.11	0.05	0.81	89.24	95.15	20.18	0.0	0.0	89.02	57.5
0.12	0.05	0.94	88.26	95.15	7.76	0.0	0.0	96.86	63.22
0.1	0.1	0.1	90.71	90.86	88.91	5.18	40.98	0.0	0.0
0.11	0.1	0.23	89.66	90.86	75.61	0.0	0.0	22.87	17.27
0.12	0.1	0.35	88.41	90.86	59.87	0.0	0.0	58.84	49.72
0.13	0.1	0.48	87.78	90.86	51.88	0.0	0.0	72.97	58.06
0.14	0.1	0.61	87.03	90.86	42.35	0.0	0.0	81.08	65.0
0.15	0.1	0.73	86.09	90.86	30.38	0.0	0.0	90.23	71.02
0.16	0.1	0.86	85.11	90.86	17.96	0.0	0.0	96.05	75.41
0.14	0.15	0.02	87.03	86.03	98.67	35.3	79.97	0.0	0.0
0.15	0.15	0.15	85.74	86.03	82.26	0.0	0.0	28.84	41.25
0.16	0.15	0.28	84.65	86.03	68.51	0.0	0.0	62.18	66.9
0.17	0.15	0.4	83.6	86.03	55.21	0.0	0.0	77.14	76.73
0.18	0.15	0.53	83.13	86.03	49.22	0.0	0.0	83.81	79.48
0.19	0.15	0.66	82.32	86.03	39.02	0.0	0.0	89.39	82.91
0.2	0.15	0.78	81.06	86.03	23.06	0.0	0.0	95.65	86.46
0.21	0.15	0.91	80.08	86.03	10.64	0.0	0.0	98.78	88.34
0.19	0.2	0.07	81.85	81.0	91.8	0.0	0.0	17.8	37.84
0.2	0.2	0.2	80.77	81.0	78.05	0.0	0.0	58.3	76.77
0.21	0.2	0.33	79.47	81.0	61.64	0.0	0.0	79.1	86.71
0.22	0.2	0.45	78.74	81.0	52.33	0.0	0.0	86.2	89.3
0.23	0.2	0.58	78.11	81.0	44.35	0.0	0.0	90.3	90.84
0.24	0.2	0.71	77.28	81.0	33.92	0.0	0.0	94.2	92.28
0.25	0.2	0.83	76.13	81.0	19.29	0.0	0.0	97.5	93.68
0.26	0.2	0.96	74.99	81.0	4.88	0.0	0.0	99.7	94.64
0.24	0.25	0.12	76.86	76.1	85.81	0.0	0.0	52.7	79.69
0.25	0.25	0.25	75.95	76.1	74.28	0.0	0.0	72.66	88.79
0.26	0.25	0.38	74.57	76.1	56.76	0.0	0.0	85.93	93.33
0.27	0.25	0.5	74.08	76.1	50.55	0.0	0.0	90.14	94.17
0.28	0.25	0.63	73.29	76.1	40.58	0.0	0.0	92.93	95.15
0.29	0.25	0.76	72.24	76.1	27.27	0.0	0.0	96.98	96.04
0.3	0.25	0.88	71.21	76.1	14.19	0.0	0.0	98.97	96.64

Table C.1: Summary results for class-wise confidence with different values of ϵ for Random split.

Epsilons			Accuracy (%)			{0, 1} sets (%)		∅ sets (%)	
ϵ_g (chosen)	ϵ_0 (chosen)	ϵ_1 (computed)	Global	Class 0	Class 1	Class 0	Class 1	Class 0	Class 1
0.01	0.01	0.01	99.3	99.33	98.94	45.44	94.63	0.0	0.0
0.02	0.01	0.13	98.37	99.06	90.5	17.38	88.92	0.0	0.0
0.03	0.01	0.26	97.29	99.06	77.11	9.98	82.01	0.0	0.0
0.04	0.01	0.38	96.09	99.06	62.2	5.93	72.79	0.0	0.0
0.05	0.01	0.54	95.0	99.09	50.92	3.37	56.78	0.0	0.0
0.06	0.01	0.63	94.21	99.06	38.88	1.74	43.4	0.0	0.0
0.07	0.01	0.75	93.0	99.06	23.76	0.0	0.0	16.0	2.83
0.08	0.01	0.88	92.25	99.06	14.47	0.0	0.0	64.0	13.38
0.05	0.05	0.05	94.32	94.43	93.04	16.67	72.1	0.0	0.0
0.06	0.05	0.18	93.32	94.43	80.65	4.13	38.76	0.0	0.0
0.07	0.05	0.31	92.29	94.43	67.83	0.0	0.0	12.63	10.14
0.08	0.05	0.44	90.91	94.43	50.65	0.0	0.0	46.08	41.41
0.09	0.05	0.57	89.79	94.43	36.74	0.0	0.0	68.6	54.3
0.1	0.05	0.7	89.07	94.43	27.83	0.0	0.0	82.25	59.94
0.11	0.05	0.83	88.37	94.43	19.13	0.0	0.0	90.78	64.25
0.12	0.05	0.95	87.24	94.43	5.0	0.0	0.0	97.95	69.57
0.1	0.1	0.1	89.93	90.0	89.02	2.7	25.81	0.0	0.0
0.11	0.1	0.23	88.64	90.0	71.96	0.0	0.0	44.95	50.0
0.12	0.1	0.36	87.78	90.0	60.51	0.0	0.0	62.1	64.5
0.13	0.1	0.49	86.64	90.0	45.33	0.0	0.0	79.05	74.36
0.14	0.1	0.62	85.9	90.0	35.51	0.0	0.0	87.62	78.26
0.15	0.1	0.75	85.02	90.0	23.83	0.0	0.0	95.24	81.6
0.16	0.1	0.88	84.0	90.0	10.28	0.0	0.0	97.52	84.38
0.14	0.15	0.02	86.18	85.08	98.31	22.78	70.41	0.0	0.0
0.15	0.15	0.15	85.03	85.08	84.39	0.0	0.0	28.79	43.24
0.16	0.15	0.28	84.29	85.08	75.53	0.0	0.0	60.38	63.79
0.17	0.15	0.41	82.9	85.08	58.65	0.0	0.0	77.45	78.57
0.18	0.15	0.54	81.85	85.08	45.99	0.0	0.0	85.61	83.59
0.19	0.15	0.68	80.77	85.08	32.91	0.0	0.0	93.12	86.79
0.2	0.15	0.81	79.68	85.08	19.62	0.0	0.0	97.71	88.98
0.21	0.15	0.94	78.56	85.08	6.12	0.0	0.0	99.49	90.56
0.19	0.2	0.08	80.39	79.47	91.15	0.0	0.0	20.35	32.5
0.2	0.2	0.2	79.69	79.47	82.3	0.0	0.0	58.66	66.25
0.21	0.2	0.32	78.89	79.47	72.12	0.0	0.0	72.65	78.57
0.22	0.2	0.44	78.19	79.47	63.27	0.0	0.0	81.22	83.73
0.23	0.2	0.55	77.22	79.47	50.88	0.0	0.0	88.58	87.84
0.24	0.2	0.67	76.47	79.47	41.37	0.0	0.0	92.91	89.81
0.25	0.2	0.79	75.49	79.47	28.98	0.0	0.0	95.86	91.59
0.26	0.2	0.91	74.17	79.47	12.17	0.0	0.0	98.8	93.2
0.24	0.25	0.12	76.37	75.47	87.0	0.0	0.0	54.07	77.59
0.25	0.25	0.25	75.34	75.47	73.77	0.0	0.0	76.73	88.89
0.26	0.25	0.38	74.53	75.47	63.45	0.0	0.0	84.33	92.02
0.27	0.25	0.51	73.43	75.47	49.33	0.0	0.0	90.15	94.25
0.28	0.25	0.64	72.34	75.47	35.43	0.0	0.0	94.57	95.49
0.29	0.25	0.77	71.18	75.47	20.63	0.0	0.0	97.52	96.33
0.3	0.25	0.89	70.29	75.47	9.19	0.0	0.0	98.91	96.79

Table C.2: Summary results for class-wise confidence with different values of ϵ for Person split.

Epsilons			Accuracy (%)			$\{0,1\}$ sets (%)		$\emptyset$ sets (%)	
ϵ_g (chosen)	ϵ_0 (chosen)	ϵ_1 (computed)	Global	Class 0	Class 1	Class 0	Class 1	Class 0	Class 1
0.01	0.01	0.01	99.73	99.98	97.66	82.61	99.65	0.0	0.0
0.02	0.01	0.14	93.63	99.98	39.57	18.59	98.35	0.0	0.0
0.03	0.01	0.28	91.07	99.98	15.11	2.58	88.98	0.0	0.0
0.04	0.01	0.41	90.24	99.98	7.23	1.14	77.97	0.0	0.0
0.05	0.01	0.55	89.88	99.98	3.83	0.25	43.48	0.0	0.0
0.06	0.01	0.68	89.66	99.98	1.7	0.0	0.0	100.0	1.08
0.07	0.01	0.82	89.5	99.98	0.21	0.0	0.0	100.0	2.56
0.05	0.05	0.05	96.65	98.7	79.15	46.45	96.44	0.0	0.0
0.06	0.05	0.18	91.29	98.7	28.09	6.2	77.33	0.0	0.0
0.07	0.05	0.32	89.75	98.7	13.4	0.0	0.0	17.31	2.46
0.08	0.05	0.45	89.01	98.7	6.38	0.0	0.0	63.46	9.77
0.09	0.05	0.59	88.7	98.7	3.4	0.0	0.0	92.31	12.56
0.1	0.05	0.72	88.43	98.7	0.85	0.0	0.0	100.0	14.81
0.11	0.05	0.86	88.36	98.7	0.21	0.0	0.0	100.0	15.35
0.1	0.1	0.1	75.52	77.64	57.45	6.76	48.08	0.0	0.0
0.11	0.1	0.23	71.57	77.64	19.79	0.0	0.0	90.62	36.34
0.12	0.1	0.37	70.45	77.64	9.15	0.0	0.0	96.32	43.79
0.13	0.1	0.5	70.05	77.64	5.32	0.0	0.0	98.44	46.07
0.14	0.1	0.64	69.78	77.64	2.77	0.0	0.0	99.89	47.48
0.15	0.1	0.77	69.53	77.64	0.43	0.0	0.0	100.0	48.72
0.14	0.15	0.02	57.49	52.93	96.38	28.46	60.81	0.0	0.0
0.15	0.15	0.15	51.46	52.93	38.94	0.0	0.0	68.08	75.26
0.16	0.15	0.28	48.89	52.93	14.47	0.0	0.0	97.51	82.34
0.17	0.15	0.42	48.13	52.93	7.23	0.0	0.0	98.62	83.72
0.18	0.15	0.55	47.76	52.93	3.62	0.0	0.0	99.68	84.33
0.19	0.15	0.69	47.51	52.93	1.28	0.0	0.0	100.0	84.7
0.2	0.15	0.82	47.4	52.93	0.21	0.0	0.0	100.0	84.86
0.19	0.2	0.07	44.9	42.18	68.09	0.0	0.0	45.23	86.0
0.2	0.2	0.2	40.41	42.18	25.32	0.0	0.0	93.01	94.02
0.21	0.2	0.33	39.04	42.18	12.34	0.0	0.0	98.19	94.9
0.22	0.2	0.47	38.4	42.18	6.17	0.0	0.0	99.31	95.24
0.23	0.2	0.6	38.08	42.18	3.19	0.0	0.0	99.87	95.38
0.24	0.2	0.74	37.84	42.18	0.85	0.0	0.0	100.0	95.49
0.25	0.2	0.87	37.77	42.18	0.21	0.0	0.0	100.0	95.52
0.24	0.25	0.12	37.06	35.59	49.57	0.0	0.0	65.01	93.25
0.25	0.25	0.25	33.71	35.59	17.66	0.0	0.0	97.21	95.87
0.26	0.25	0.38	32.79	35.59	8.94	0.0	0.0	98.92	96.26
0.27	0.25	0.52	32.32	35.59	4.47	0.0	0.0	99.65	96.44
0.28	0.25	0.65	32.1	35.59	2.34	0.0	0.0	100.0	96.51
0.29	0.25	0.79	31.9	35.59	0.43	0.0	0.0	100.0	96.58

Table C.3: Summary results for class-wise confidence with different values of ϵ for Time split.

Bibliography

Abideen, Zain Ul, Mubeen Ghafoor, Kamran Munir, Madeeha Saqib, Ata Ullah, Tehseen Zia, Syed Ali Tariq, Ghufran Ahmed, and Asma Zahra (2020). "Uncertainty assisted robust tuberculosis identification with bayesian convolutional neural networks". In: *Ieee Access* 8, pp. 22812–22825 (cited on p. 10).

Aho, Timo, Bernard Ženko, and Sašo Džeroski (2009). "Rule ensembles for multi-target regression". In: *2009 Ninth IEEE International Conference on Data Mining*. IEEE, pp. 21–30 (cited on p. 37).

Alarcón, Yonatan Carlos Carranza, Soundouss Messoudi, and Sébastien Destercke (2020a). "Apprentissage de rangements prudent avec satisfaction de contraintes". In: *29èmes Rencontres francophones sur la Logique Floue et ses Applications* (cited on p. 4).

— (2020b). "Cautious label-wise ranking with constraint satisfaction". In: *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*. Springer, pp. 96–111 (cited on p. 4).

Angelopoulos, Anastasios N and Stephen Bates (2021). "A gentle introduction to conformal prediction and distribution-free uncertainty quantification". In: *arXiv preprint arXiv:2107.07511* (cited on p. 13).

Arzamasov, Vadim (2018). *UCI Electrical Grid Stability Simulated Data Data Set*. URL: https://archive.ics.uci.edu/ml/datasets/Electrical+Grid+Stability+Simulated+Data+ (cited on p. 27).

Balasubramanian, Vineeth, Shen-Shyang Ho, and Vladimir Vovk (2014). *Conformal prediction for reliable machine learning: theory, adaptations and applications*. Newnes (cited on p. 16).

Baldassarre, Luca, Lorenzo Rosasco, Annalisa Barla, and Alessandro Verri (2012). "Multi-output learning via spectral filtering". In: *Machine learning* 87.3, pp. 259–301 (cited on p. 37).

Barber, Rina Foygel, Emmanuel J Candes, Aaditya Ramdas, and Ryan J Tibshirani (2021). "Predictive inference with the jackknife+". In: *The Annals of Statistics* 49.1, pp. 486–507 (cited on p. 98).

Bayes, Thomas (1763). "LII. An essay towards solving a problem in the doctrine of chances. By the late Rev. Mr. Bayes, FRS communicated by Mr. Price, in a letter to John Canton, AMFR S". In: *Philosophical transactions of the Royal Society of London* 53, pp. 370–418 (cited on p. 10).

Borchani, Hanen, Gherardo Varando, Concha Bielza, and Pedro Larrañaga (2015). "A survey on multi-output regression". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 5.5, pp. 216–233 (cited on p. 36).

Boström, Henrik and Ulf Johansson (2020). "Mondrian conformal regressors". In: *Conformal and Probabilistic Prediction and Applications*. PMLR, pp. 114–133 (cited on p. 23).

Caruana, Rich (1993). "Multitask learning: A knowledge-based source of inductive bias". In: *Proceedings of the Tenth International Conference on Machine Learning*, pp. 41–48 (cited on p. 42).

— (1998). "A dozen tricks with multitask learning". In: *Neural networks: tricks of the trade*. Springer, pp. 165–191 (cited on p. 36).

Chernozhukov, Victor, Kaspar Wüthrich, and Zhu Yinchu (2018). "Exact and robust conformal inference methods for predictive machine learning with dependent data". In: *Conference On Learning Theory*. PMLR, pp. 732–749 (cited on p. 98).

Cho, Dongjin, Cheolhee Yoo, Jungho Im, and Dong-Hyun Cha (2020). "Comparative assessment of various machine learning-based bias correction methods for numerical weather prediction model forecasts of extreme air temperatures in urban areas". In: *Earth and Space Science* 7.4, e2019EA000740 (cited on p. 40).

Colombo, Nicolo and Vladimir Vovk (2020). "Training conformal predictors". In: *Conformal and Probabilistic Prediction and Applications*. PMLR, pp. 55–64 (cited on p. 98).

De Brébisson, Alexandre, Étienne Simon, Alex Auvolat, Pascal Vincent, and Yoshua Bengio (2015). "Artificial neural networks applied to taxi destination prediction". In: *arXiv preprint arXiv:1508.00021* (cited on p. 43).

De'Ath, Glenn (2002). "Multivariate regression trees: a new technique for modeling species–environment relationships". In: *Ecology* 83.4, pp. 1105–1117 (cited on p. 37).

Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei (2009). "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255 (cited on p. 29).

Der Kiureghian, Armen and Ove Ditlevsen (2009). "Aleatory or epistemic? Does it matter?" In: *Structural safety* 31.2, pp. 105–112 (cited on p. 8).

Domingos, Pedro (1999). "Metacost: A general method for making classifiers cost-sensitive". In: *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 155–164 (cited on p. 81).

Embrechts, Paul, Alexander McNeil, and Daniel Straumann (2002). "Correlation and dependence in risk management: properties and pitfalls". In: *Risk management: value at risk and beyond* 1, pp. 176–223 (cited on p. 52).

Favre, Anne-Catherine, Salaheddine El Adlouni, Luc Perreault, Nathalie Thiémonge, and Bernard Bobée (2004). "Multivariate hydrological frequency analysis using copulas". In: *Water resources research* 40.1 (cited on p. 52).

Frank, Maurice J (1979). "On the simultaneous associativity of F(x,y) and x+y-F(x,y)". In: *Aequationes mathematicae* 19.1, pp. 194–226 (cited on p. 55).

Fréchet, Maurice (1951). "Sur les tableaux de corrélation dont les marges sont données". In: *Ann. Univ. Lyon, 3ê serie, Sciences, Sect. A* 14, pp. 53–77 (cited on p. 52).

Freni, Gabriele and Giorgio Mannina (2010). "Bayesian approach for uncertainty quantification in water quality modelling: The influence of prior distribution". In: *Journal of Hydrology* 392.1-2, pp. 31–39 (cited on p. 10).

Gammerman, Alex, Volodya Vovk, and Vladimir Vapnik (1998). "Learning by transduction". In: *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, 148–155 (cited on p. 1).

Genest, Christian and Louis-Paul Rivest (1993). "Statistical inference procedures for bivariate Archimedean copulas". In: *Journal of the American statistical Association* 88.423, pp. 1034–1043 (cited on p. 55).

Gibbs, Isaac and Emmanuel Candès (2021). "Adaptive Conformal Inference Under Distribution Shift". In: *arXiv preprint arXiv:2106.00170* (cited on p. 95).

Gneiting, Tilmann and Adrian E Raftery (2007). "Strictly proper scoring rules, prediction, and estimation". In: *Journal of the American statistical Association* 102.477, pp. 359–378 (cited on p. 11).

Gopalan, Giri, Andrew Zammit-Mangion, and Felicity McCormack (2021). "A Review of Bayesian Modelling in Glaciology". In: *arXiv preprint arXiv:2112.13663* (cited on p. 10).

Gumbel, Emil Julius (1960). "Distributions des valeurs extremes en plusiers dimensions". In: *Publ. Inst. Statist. Univ. Paris* 9, pp. 171–173 (cited on p. 55).

Guo, Cheng and Felix Berkhahn (2016). "Entity embeddings of categorical variables". In: *arXiv preprint arXiv:1604.06737* (cited on p. 43).

Guo, Chuan, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger (2017). "On calibration of modern neural networks". In: *International Conference on Machine Learning*. PMLR, pp. 1321–1330 (cited on p. 12).

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). "Deep residual learning for image recognition". In: *CVPR*, pp. 770–778 (cited on p. 29).

Hechtlinger, Yotam, Barnabás Póczos, and Larry Wasserman (2018). "Cautious deep learning". In: *arXiv preprint arXiv:1805.09460* (cited on pp. 17, 24).

Hendrycks, Dan, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song (2021). "Natural adversarial examples". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15262–15271 (cited on p. 7).

Hofert, Marius, Ivan Kojadinovic, Martin Mächler, and Jun Yan (2019). *Elements of copula modeling with R*. Springer (cited on pp. 55, 58, 59, 62).

Höffding, Wassilij (1940). "Masstabinvariante korrelationstheorie". In: *Schriften des Mathematischen Instituts und Instituts fur Angewandte Mathematik der Universitat Berlin* 5, pp. 181–233 (cited on p. 52).

Höffding, Wassily (1941). "Masstabinvariante korrelationsmasse für diskontinuierliche Verteilungen". In: *Archiv für mathematische Wirtschafts-und Sozialforschung* 7, pp. 49–70 (cited on p. 52).

— (1994). "Scale—invariant correlation theory". In: *The collected works of Wassily Höffding*. Springer, pp. 57–107 (cited on p. 52).

Hüllermeier, Eyke and Willem Waegeman (2021). "Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods". In: *Machine Learning* 110.3, pp. 457–506 (cited on pp. 8, 14).

Joe, Harry and Dorota Kurowicka (2011). *Dependence modeling: vine copula handbook*. World Scientific (cited on p. 97).

Johnstone, Chancellor and Bruce Cox (2021). "Conformal uncertainty sets for robust optimization". In: *Conformal and Probabilistic Prediction and Applications*. PMLR, pp. 72–90 (cited on pp. 69, 72).

Kendall, Alex and Yarin Gal (2017). "What uncertainties do we need in bayesian deep learning for computer vision?" In: *Advances in neural information processing systems* 30 (cited on p. 11).

Klambauer, Günter, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter (2017). "Self-normalizing neural networks". In: *Advances in neural information processing systems*, pp. 971–980 (cited on pp. 46, 60).

Kuleshov, Alexander, Alexander Bernstein, and Evgeny Burnaev (2018). "Conformal prediction in manifold learning". In: *Conformal and Probabilistic Prediction and Applications*, pp. 234–253 (cited on p. 36).

Kumar, Ananya, Percy S Liang, and Tengyu Ma (2019). "Verified uncertainty calibration". In: *Advances in Neural Information Processing Systems* 32 (cited on p. 12).

Lann, E (1959). *Testing statistical hypotheses*. Wiley, New York (cited on p. 81).

Li, Tao and Mitsunori Ogihara (2003). "Detecting emotion in music". In: Johns Hopkins University (cited on p. 37).

Liu, Weiwei (2019). "Copula Multi-label Learning". In: *Advances in Neural Information Processing Systems*, pp. 6337–6346 (cited on p. 98).

Liu, Ziwei, Ping Luo, Xiaogang Wang, and Xiaoou Tang (2015). "Deep Learning Face Attributes in the Wild". In: *Proceedings of International Conference on Computer Vision (ICCV)* (cited on p. 27).

Maas, Andrew L, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts (2011). "Learning word vectors for sentiment analysis". In: *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-vol. 1*, pp. 142–150 (cited on p. 27).

Manokhin, Valery (2022). *Awesome Conformal Prediction*. DOI: 10.5281/ZENODO.6467205. URL: https://zenodo.org/record/6467205 (cited on p. 1).

Manville, Michael, Paavo Monkkonen, Michael Lens, and Richard Green (2020). "COVID-19 and Renter Distress: Evidence from Los Angeles". In: (cited on p. 80).

McNeil, Alexander J, Rüdiger Frey, and Paul Embrechts (2015). *Quantitative risk management: concepts, techniques and tools-revised edition*. Princeton university press (cited on pp. 53, 54).

Messoudi, Soundouss, Sébastien Destercke, and Sylvain Rousseau (2020a). "Conformal multi-target regression using neural networks". In: *Conformal and Probabilistic Prediction and Applications*. PMLR, pp. 65–83 (cited on pp. 3, 36).

Messoudi, Soundouss, Sylvain Rousseau, and Sébastien Destercke (2020b). "Deep Conformal Prediction for Robust Models". In: *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*. Springer, pp. 528–540 (cited on pp. 3, 7, 15, 24).

— (2020c). "Prédiction conformelle profonde pour des modèles robustes". In: *Extraction et Gestion des Connaissances (EGC 2020)* (cited on p. 4).

Messoudi, Soundouss, Sébastien Destercke, and Sylvain Rousseau (2021a). "Class-wise confidence for debt prediction in real estate management: discussion and lessons learned from an application". In: *Conformal and Probabilistic Prediction and Applications*. PMLR, pp. 211–228 (cited on pp. 4, 79).

— (2021b). "Confiance de classe pour la prédiction de dette en gestion immobilière". In: *30èmes Rencontres francophones sur la Logique Floue et ses Applications* (cited on p. 4).

— (2021c). "Copula-based conformal prediction for multi-target regression". In: *Pattern Recognition* 120, p. 108101 (cited on pp. 3, 36).

— (2022a). "Ellipsoidal conformal inference for Multi-Target Regression". In: *Conformal and Probabilistic Prediction with Applications*. PMLR, pp. 294–306 (cited on pp. 3, 36).

— (2022b). "Prédiction conformelle basée sur les copules pour la régression multi-cibles". In: *Extraction et Gestion des Connaissances (EGC 2022)* (cited on p. 4).

Michelmore, Rhiannon, Matthew Wicker, Luca Laurenti, Luca Cardelli, Yarin Gal, and Marta Kwiatkowska (2020). "Uncertainty quantification with statistical guarantees in end-to-end autonomous driving control". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 7344–7350 (cited on p. 10).

Monarch, Robert Munro (2021). *Human-in-the-Loop Machine Learning: Active learning and annotation for human-centered AI*. Simon and Schuster (cited on p. 9).

Naeini, Mahdi Pakdaman, Gregory Cooper, and Milos Hauskrecht (2015). "Obtaining well calibrated probabilities using bayesian binning". In: *Twenty-Ninth AAAI Conference on Artificial Intelligence* (cited on p. 12).

Neeven, Jelmer and Evgueni Smirnov (2018). "Conformal stacked weather forecasting". In: *Conformal and Probabilistic Prediction and Applications*. PMLR, pp. 220–233 (cited on p. 36).

Nelsen, Roger B (1999). "An Introduction to Copulas, volume 139 of". In: *Lecture Notes in Statistics* (cited on pp. 52, 54).

Nikoloulopoulos, Aristidis K and Dimitris Karlis (2008). "Multivariate logit copula model with an application to dental data". In: *Statistics in Medicine* 27.30, pp. 6393–6406 (cited on p. 52).

Nugteren, Cedric and Valeriu Codreanu (2015). "CLTune: A generic auto-tuner for OpenCL kernels". In: *2015 IEEE 9th International Symposium on Embedded Multicore/Many-core Systems-on-Chip*. IEEE, pp. 195–202 (cited on p. 40).

Papadopoulos, Harris (2008). "Inductive conformal prediction: Theory and application to neural networks". In: *Tools in artificial intelligence*. InteChOpen (cited on p. 20).

Papadopoulos, Harris, Vladimir Vovk, and Alexander Gammerman (2011a). "Regression conformal prediction with nearest neighbours". In: *Journal of Artificial Intelligence Research* 40, pp. 815–840 (cited on pp. 23, 36).

Papadopoulos, Harris and Haris Haralambous (2011b). "Reliable prediction intervals with regression neural networks". In: *Neural Networks* 24.8, pp. 842–851 (cited on pp. 23, 59).

Platt, John et al. (1999). "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods". In: *Advances in large margin classifiers* 10.3, pp. 61–74 (cited on p. 11).

Rafiei, Mohammad Hossein and Hojjat Adeli (2016). "A novel machine learning model for estimation of sale prices of real estate units". In: *Journal of Construction Engineering and Management* 142.2, p. 04015066 (cited on p. 40).

Read, Jesse, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank (2011). "Classifier chains for multi-label classification". In: *Machine learning* 85.3, p. 333 (cited on p. 37).

Redmond, M (2011). "Communities and crime unnormalized data set". In: *UCI Machine Learning Repository. In website: http://www. ics. uci. edu/mlearn/MLRepository. html* (cited on p. 40).

Ruder, Sebastian (2017). "An overview of multi-task learning in deep neural networks". In: *arXiv preprint arXiv:1706.05098* (cited on p. 42).

Ruschendorf, Ludger (1976). "Asymptotic distributions of multivariate rank order statistics". In: *The Annals of Statistics*, pp. 912–923 (cited on pp. 55, 59).

Ruymgaart, Frederik Hendrik (1978). "Asymptotic theory of rank tests for independence". In: *MC Tracts* (cited on pp. 55, 59).

Sánchez-Fernández, Matilde, Mario de Prado-Cumplido, Jerónimo Arenas-García, and Fernando Pérez-Cruz (2004). "SVM multiregression for nonlinear channel estimation in multiple-input multiple-output systems". In: *IEEE transactions on signal processing* 52.8, pp. 2298–2307 (cited on p. 37).

Sarker, Iqbal H (2021). "Machine learning: Algorithms, real-world applications and research directions". In: *SN Computer Science* 2.3, pp. 1–21 (cited on p. 1).

Schmidt, Thorsten (2007). "Coping with copulas". In: *Copulas-From theory to application in finance*, pp. 3–34 (cited on p. 53).

Shafer, Glenn and Vladimir Vovk (2008). "A Tutorial on Conformal Prediction." In: *Journal of Machine Learning Research* 9.3 (cited on p. 16).

Sheng, Victor S and Charles X Ling (2006). "Thresholding for making classifiers cost-sensitive". In: *Aaai*. Vol. 6, pp. 476–81 (cited on p. 81).

Sklar, M (1959). "Fonctions de repartition an dimensions et leurs marges". In: *Publ. inst. statist. univ. Paris* 8, pp. 229–231 (cited on pp. 52, 53).

Spyromitros-Xioufis, Eleftherios, Grigorios Tsoumakas, William Groves, and Ioannis Vlahavas (2012). "Multi-label classification methods for multi-target regression". In: *arXiv preprint arXiv:1211.6581*, pp. 1159–1168 (cited on p. 37).

— (2016). "Multi-target regression via input space expansion: treating targets as inputs". In: *Machine Learning* 104.1, pp. 55–98 (cited on p. 38).

Tapkan, Pınar, Lale Özbakır, Sinem Kulluk, and Adil Baykasoğlu (2016). "A cost-sensitive classification algorithm: BEE-Miner". In: *Knowledge-Based Systems* 95, pp. 99–113 (cited on p. 81).

Tibshirani, Ryan J, Rina Foygel Barber, Emmanuel Candes, and Aaditya Ramdas (2019). "Conformal prediction under covariate shift". In: *Advances in neural information processing systems* 32 (cited on p. 98).

Tong, Xin, Yang Feng, and Anqi Zhao (2016). "A survey on Neyman-Pearson classification and suggestions for future research". In: *Wiley Interdisciplinary Reviews: Computational Statistics* 8.2, pp. 64–81 (cited on p. 81).

Torres-Sospedra, Joaquín, Raúl Montoliu, Adolfo Martínez-Usó, Joan P Avariento, Tomás J Arnau, Mauri Benedito-Bordonau, and Joaquín Huerta (2014). "UJIIndoorLoc: A new multi-building and multi-floor database for WLAN fingerprint-based indoor localization problems". In: *2014 international conference on indoor positioning and indoor navigation (IPIN)*. IEEE, pp. 261–270 (cited on p. 40).

Tsoumakas, Grigorios, Eleftherios Spyromitros-Xioufis, Jozef Vilcek, and Ioannis Vlahavas (2011). "MULAN: A Java Library for Multi-Label Learning". In: *Journal of Machine Learning Research* 12.71, pp. 2411–2414. URL: http://jmlr.org/papers/v12/tsoumakas11a.html (cited on p. 40).

Turney, Peter D (1994). "Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm". In: *Journal of artificial intelligence research* 2, pp. 369–409 (cited on p. 81).

Vembu, Shankar and Thomas Gärtner (2010). "Label ranking algorithms: A survey". In: *Preference learning*. Springer, pp. 45–64 (cited on p. 36).

Vovk, Vladimir, David Lindsay, Ilia Nouretdinov, and Alex Gammerman (2003). "Mondrian confidence machine". In: *Technical Report* (cited on p. 23).

Vovk, Vladimir, Alex Gammerman, and Glenn Shafer (2005). *Algorithmic learning in a random world*. Springer Science & Business Media (cited on pp. 6, 13, 15, 25, 26).

Vovk, Vladimir and Ivan Petej (2014). "Venn-Abers predictors". In: *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, pp. 829–838 (cited on p. 96).

Vovk, Vladimir, Jieli Shen, Valery Manokhin, and Min-ge Xie (2017). "Nonparametric predictive distributions based on conformal prediction". In: *Conformal and Probabilistic Prediction and Applications*. PMLR, pp. 82–102 (cited on p. 97).

Walley, Peter (1991). "Statistical reasoning with imprecise probabilities". In: (cited on p. 11).

Wang, Huazhen, Xin Liu, Ilia Nouretdinov, and Zhiyuan Luo (2015). "A comparison of three implementations of multi-label conformal prediction". In: *International Symposium on Statistical Learning and Data Sciences*. Springer, pp. 241–250 (cited on p. 36).

Wang, Ran, Sam Kwong, Xu Wang, and Yuheng Jia (2020). "Active k-Labelsets Ensemble for Multi-label Classification". In: *Pattern Recognition*, p. 107583 (cited on p. 36).

Weisstein, Eric W (2004). "Bonferroni correction". In: *https://mathworld. wolfram. com/* (cited on p. 42).

Zadrozny, Bianca and Charles Elkan (2001). "Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers". In: *Icml*. Vol. 1. Citeseer, pp. 609–616 (cited on p. 11).

— (2002). "Transforming classifier scores into accurate multiclass probability estimates". In: *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 694–699 (cited on p. 12).

Zadrozny, Bianca, John Langford, and Naoki Abe (2003). "Cost-sensitive learning by cost-proportionate example weighting". In: *Third IEEE international conference on data mining*. IEEE, pp. 435–442 (cited on p. 81).

Zaffalon, Marco (2002). "The naive credal classifier". In: *Journal of statistical planning and inference* 105.1, pp. 5–21 (cited on p. 11).

Zhang, Min-Ling and Zhi-Hua Zhou (2013). "A review on multi-label learning algorithms". In: *IEEE transactions on knowledge and data engineering* 26.8, pp. 1819–1837 (cited on p. 36).

Zhou, Fang, Q Claire, and Ross D King (2014). "Predicting the geographical origin of music". In: *2014 IEEE International Conference on Data Mining*. IEEE, pp. 1115–1120 (cited on p. 40).

List of Tables

List of Figures

List of Abbreviations

AI	**Artificial Intelligence**
ML	**Machine Learning**
DL	**Deep Learning**
NN	**Neural Networks**
CNN	**Convolutional Neural Networks**
GRU	**Gated Recurrent Unit**
MLP	**Multi Layer Perceptron**
RF	**Random Forest**
MTR	**Multi Target Regression**
ICP	**Inductive Conformal Prediction**
MCP	**Mondrian Conformal Prediction**
NCM	**Non Conformity Measure**
k**NN**	k **Nearest Neighbors**
O-KNN	**Original** k**NN**
OS-KNN	**Original** k**NN with Standard Deviation**
R-KNN	**Representation** k**NN**
OS-KNN	**Representation** k**NN with Standard Deviation**
SEC	**Standard Empirical Copula**
NEC	**Normalized Empirical Copula**
SGE	**Standard Global Ellipsoid**
NLE	**Normalized Local Ellipsoid**
TP	**True Positive**
FP	**False Positive**
TN	**True Negative**
FN	**False Negative**

FPR	**False Positive Rate**
FNR	**False Negative Rate**
i.i.d.	**i**ndependent and **i**dentically **d**istributed
c.d.f.	**c**umulative **d**istribution **f**unction