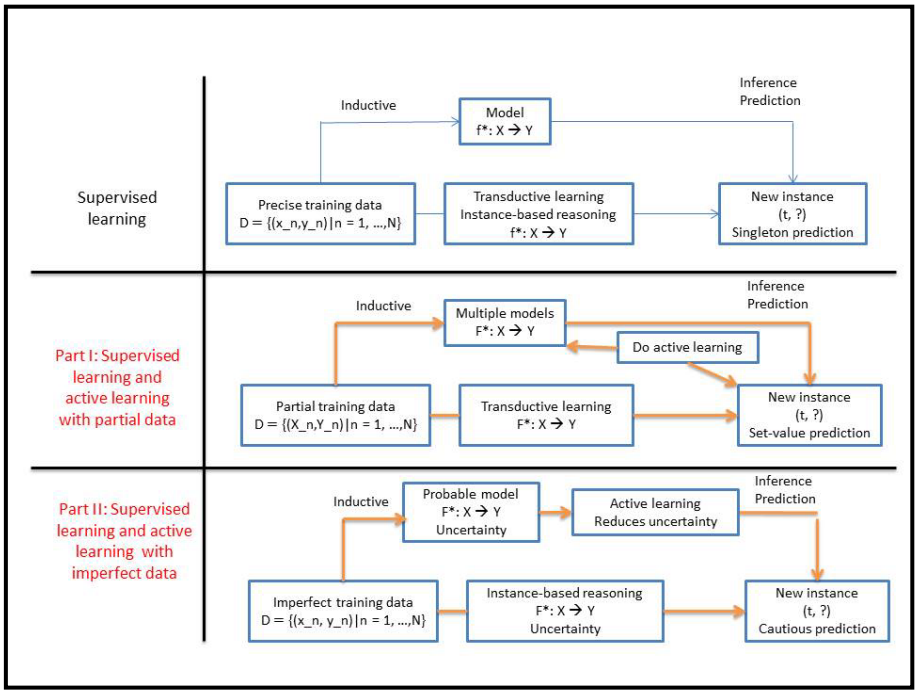
Imprecision in machine learning problems

Vu-Linh Nguyen

Par **Vu-Linh NGUYEN**

Imprecision in machine learning problems

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



Soutenue le 27 septembre 2018
Spécialité : Informatique : Unité de recherche Heudyasic (UMR-7253)

D2433

SORBONNE UNIVERSITES

UNIVERSITY OF TECHNOLOGY OF COMPIÈGNE

DOCTORAL THESIS

Imprecision in Machine Learning Problems

Spécialité : Informatique

Author:
Vu-Linh NGUYEN
(Nguyễn Vũ Linh)

Supervisors:
Dr. Sébastien DESTERCKE
(CNRS Researcher)
Assoc. Prof. Marie-Hélène MASSON

Jury:
Assoc. Prof. Cassio Polpo de CAMPOS
(Reviewer)
Utrecht University,
Utrecht, The Netherlands
Prof. Inés COUSO
(Reviewer)
University of Oviedo,
Oviedo, Spain
Prof. Thierry DENOEUX
(Examiner)
University of Technology of Compiègne, Compiègne, France
Prof. Eyke HÜLLERMEIER
(Examiner)
Paderborn University,
Paderborn, Germany

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

CID (Connaissances, Incertitudes, Données) Team
Heudiasyc Laboratory

September 27, 2018

Acknowledgements

This work has been funded by the University of Technology of Compiègne (UTC).

My first thanks go to my two supervisors, Dr. Sébastien DESTERCKE and Assoc. Prof. Marie-Hélène MASSON. Finding appropriate words to describe their tremendous supports appears to be extremely exhausted, so I would like to simply say thank you for helping me to see uncertainty less uncertain.

I would like to thank Assoc. Prof. Van-Nam HUYNH, Japan Advanced Institute of Science and Technology (JAIST), who introduced me to the UTC and has encouraged me since my first days at JAIST.

Thanks to the members of my jury for their comments and the various discussions about my works.

I also especially thank Prof. Eyke HÜLLERMEIER and members of the Intelligent Systems and Machine Learning group, Paderborn University, who have given me the opportunities to visit and collaborate with them on scientific interests that we have in common.

I would also like to thank members of the UTC, JAIST and others for sharing the office with me, and the enjoyable moments spent together, scientific conversations, sport, friendship and beers. So, in a non exhaustive list, they are: Dang-Phong Bach, Ngoc-Thang Bui, Carranza Alarcon Yonatan Carlos, Alia Chebly, Xuan-Nam Do, Clément Dubos, Dinh-Hiep Duong, Alberto García-Durán, Duc-Anh Hoang, Mohamed Ali Kandi, Minh-Ly Lieu, Duc-Hieu Nguyen, Tan-Nhu Nguyen, Cong-Cuong Pham, Trung-Nghia Phung, Shameem Puthiya Parambath, Khoat Than, Thi-Thuy-Hong Trinh, Gen Yang, ...

My final thanks go to my family, who have encouraged me, even if they do not understand what my job is really about. Especially, thank you mother, Thi-Tuoi VU, who gave me the first lectures on the perseverance, farming.

"*Somewhere, something incredible is waiting to be known.*"

– Carl Sagan

UNIVERSITY OF TECHNOLOGY OF COMPIÈGNE

Abstract

CID (Connaissances, Incertitudes, Données) Team
Heudiasyc Laboratory

Doctor

Imprecision in Machine Learning Problems

by Vu-Linh NGUYEN
(Nguyễn Vũ Linh)

We have focused on imprecision modeling in machine learning problems, where available data or knowledge suffers from important imperfections. In this work, imper-fect data refers to situations where either some features or the labels are imperfectly known, that is can be specified by sets of possible values rather than precise ones. Learning from partial data are commonly encountered in various fields, such as bio-statistics, agronomy, or economy. These data can be generated by coarse or censored measurements, or can be obtained from expert opinions. On the other hand, imperfect knowledge refers to the situations where data are precisely specified, however, there are classes, that cannot be distinguished due to a lack of knowledge (also known as epistemic uncertainty) or due to a high uncertainty (also known as aleatoric uncertainty).

Considering the problem of learning from partially specified data, we highlight the potential issues of dealing with multiple optimal classes and multiple optimal models in the inference and learning step, respectively. We have proposed active learning approaches to reduce the imprecision in these situations. Yet, the distinction epistemic/aleatoric uncertainty has been well-studied in the literature. To facilitate subsequent machine learning applications, we have developed practical procedures to estimate these degrees for popular classifiers. In particular, we have explored the use of this distinction in the contexts of active learning and cautious inferences.

Keywords imprecision, machine learning, active learning, racing algorithms, epistemic uncertainty, aleatoric uncertainty, multi-class classification

List of contributions

[1] Nguyen, V.-L., Destercke, S., Masson, M.-H. & Hüllermeier, E. *Reliable Multi-class classification based on pairwise epistemic and aleatoric uncertainty* in *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)* (2018), 5089-5095.

[2] Nguyen, V.-L., Destercke, S. & Masson, M.-H. Partial data querying through racing algorithms. *International Journal of Approximate Reasoning* **96**, 36-55 (2018).

[3] Nguyen, V.-L., Destercke, S. & Masson, M.-H. *K-nearest neighbour classification for interval-valued data* in *Proceedings of the 11th International Conference on Scalable Uncertainty Management (SUM)*(2017), 93-106.

[4] Nguyen, V.-L., Destercke, S. & Masson, M.-H. *Querying partially labelled data to improve a K-nn classifier* in *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI)*(2017), 2401 - 2407.

[5] Nguyen, V.-L., Destercke, S. & Masson, M.-H. *Partial data querying through racing algorithms* in *Proceedings of the 5th International Symposium on Integrated Uncertainty in Knowledge Modelling and Decision Making (IUKM)*(2016), 163-174.

[6] Nguyen, V.-L., Shaker, A., Hüllermeier, E., Destercke, S. & Masson, M.-H. Epistemic and aleatoric uncertainty in active learning. *submitted* (2018).

[7] Nguyen, V.-L., Destercke, S., Masson, M.-H. & Ghassani, R. Racing trees to query partial data. *submitted* (2017).

Contents

List of Figures

List of Tables

List of Notations

Symbols	Descriptions
	Data related notations
$\mathcal{X}$	input space, dimension P
$\mathcal{Y}$	output space, dimension M
$\boldsymbol{x}$	precise input
$\mathbf{X}$	imprecise input
X_n^p	p-th imprecise coordinate of instance $\mathbf{X_n}$
x_n^p	p-th precise coordinate of instance $\mathbf{X_n}$
y	precise output
Y	imprecise output
y_m	m-th class among the M possible ones
N	number of training instances
$\mathbf{D}, \mathbf{T}, \mathbf{U}$	training, test and pool data set
$\boldsymbol{t}$	input of instance of either pool or test set
$\mathcal{D}, \mathbf{d}$	set of replacements of $\mathbf{D}$ and its element
	Hypothesis-space, model related notations
θ	either a hypothesis or its corresponding parameters
Θ	set of models, dimension S
$\theta(\boldsymbol{x})$ $(\theta(\boldsymbol{t}))$	output of model θ for input $\boldsymbol{x}$ $(\boldsymbol{t})$
$\ell(y, \theta(\boldsymbol{x}))$	0-1 loss
$\underline{\ell}(Y, \theta(\mathbf{X}))$,$\overline{\ell}(Y, \theta(\mathbf{X}))$	lower and upper 0-1 losses
$R(\theta \mid \mathbf{D})$	empirical risk of model θ
$\underline{R}(\theta \mid \mathbf{D})$,$\overline{R}(\theta \mid \mathbf{D})$	lower and upper empirical risks of model θ
θ^*_{mm}, θ^*_{mM}	minimin and minimax optimal models
$\underline{R}(\theta_{k-l} \mid \mathbf{D})$	lower difference of empirical risks between θ_k and θ_l
$L(\theta \mid \mathbf{D})$	discriminative likelihood of model θ
θ^*	optimal model within Θ
	Uncertainty related notations
$[\underline{r}_n, \overline{r}_n]$	set-valued rank of $\boldsymbol{x}_n$
$s_{\boldsymbol{t}}^{max}(y)$, $s_{\boldsymbol{t}}^{min}(y)$, $s_{\boldsymbol{t}}^{small}(y)$	extremely voting scores of label y
$\mathbf{PL}_{\boldsymbol{t}}, \mathbf{NL}_{\boldsymbol{t}}$	possible and necessary label set of $\boldsymbol{t}$
$\mathbf{N}_{\boldsymbol{t}}$	nearest neighbour set of $\boldsymbol{t}$
$\mathbf{PN}_{\boldsymbol{t}}, \mathbf{NN}_{\boldsymbol{t}}$	possible and necessary neighbour of $\boldsymbol{t}$ set
Θ^*	set of undominated models
	Query related notations
q_n^p	a query: asking for the precise value of the X_n^p
q_n	a query: asking for the precise label of instance n
$E_{q_n^p}(\theta_l)$	single effect of query q_n^p
$J_{q_n^p}(\theta_k, \theta_l)$	pairwise effect of query q_n^p
	Decision tree related notations
H	number of terminal nodes of a tree

A_h	h-th terminal node of a tree
A_h^p	projection of A_h on the p-th axis
y_h	class associated to leaf A_h
	Experiment related notations
(ϵ, η)	contamination parameters in the experiments
	Probability related notations
$p_\theta(y \mid \boldsymbol{x})$	conditional probability given for label y by model θ
$\pi_\Theta(\theta)$	normalized likelihood of model θ
$\pi(y \mid \boldsymbol{x})$	degree of support for label y
$u_e(\boldsymbol{x}), u_a(\boldsymbol{x})$	degrees of epistemic and aleatoric uncertainty
$s_y(\boldsymbol{x})$	degree of strict preference
$s(\theta, \boldsymbol{x})$	degree of uncertainty
$\lvert\mathbf{A}\rvert$	cardinality of set $\mathbf{A}$
	Fuzzy relations related notations
$\succ$	strict preference
$\sim$	incomparability
$\perp$	indifference

To my parents

Chapter 1

Introduction

This work focuses on imprecision modeling in machine learning problems, where available data or knowledge suffers from important imperfections. By imperfect data, we refer to the situations where either some features or the labels are imperfectly known, that is can be specified by sets of possible values rather than precise ones. For example, when the label of some training instances is only known to belong to a set of labels, or when some features are imprecisely given in the form of intervals (or, more generally, sets). In the second scenario, imperfect knowledge refers to the situations where data are precisely specified, however, there are classes, that cannot be distinguished due to a lack of knowledge (also known as epistemic uncertainty) or due to a high uncertainty (also known as aleatoric uncertainty). In this introduction, we are going to formulate the problems we will consider, before providing a quick overview of our contributions. We are first going to summarize the basics of the learning problem, then highlight possible scenarios where the classical methods is likely to be insufficient and quickly introduce our proposals to tackle these situations.

1.1 Learning problems

Learning is, in general, the problem of teaching a learner (classifier) to generalize from experience [7, 47]. In the context of supervised learning, generalization refers to the ability of a learning machine to perform accurately on new examples after having experienced a training data set [89, 92]. Almost of the works on supervised learning literature can be categorized into either inductive techniques or transductive techniques. Roughly speaking, inductive techniques learn a model θ^* issued from the hypothesis space $\Theta \subseteq \mathcal{Y}^{\mathcal{X}}$ that best fits the training data set $\mathbf{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N} \subseteq \mathcal{X} \times \mathcal{Y}$ of N input/output samples, where $\mathcal{X} := \mathbb{R}^P$ and $\mathcal{Y} := \{y_1, \ldots, y_M\}$ are, respectively, the input and the output spaces, and uses θ^* to make predictions for new instances $(\boldsymbol{t}, ?)$. When using transductive techniques, training data are used directly to perform the inference step (on new instances $(\boldsymbol{t}, ?)$) without any induction step. This is illustrated by Figure 1.1.

Before going further, let us note that, we denote by Θ the underlying hypothesis space, i.e., the class of candidate models $\theta : \mathcal{X} \longrightarrow \mathcal{Y}$ the learner can choose from. Often, hypotheses are parametrized by a parameter vector $\theta \in \Theta$; in this case, we equate a hypothesis with the parameter θ, and the model space with the parameter space Θ.

Inductive learning

The goal of inductive learning (the upper path of Figure 1.1) is to extract a model $\theta^* : \mathcal{X} \to \mathcal{Y}$ within a model space Θ which best fits the training data set $\mathbf{D}$ [34, 40, 89–91]. This strategy has been widely studied and detailed for numerous applications,

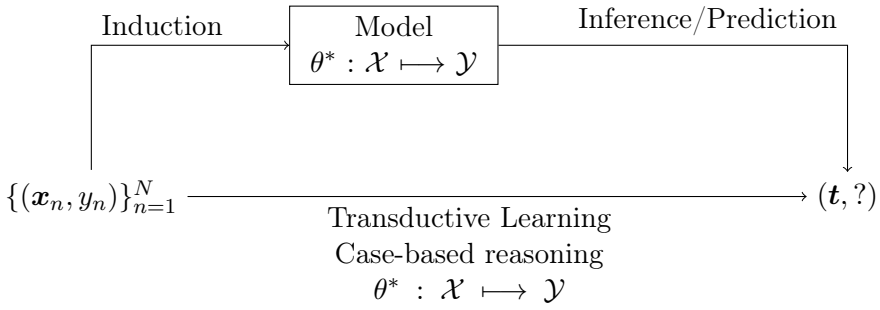


FIGURE 1.1: Inductive versus transductive learning

e.g, support vector machine (SVM) [10, 16], logistic regression [21, 93], Naive Bayes [77], etc. Two classical map to derive the optimal model θ^* are to use either the loss minimisation approach which seeks for the model (in the hypothesis space) that minimizes the loss on the training set, or, the likelihood maximization approach whose optimal candidate is the one that is most probable for the training data, usually assuming that there are i.i.d observations.

In the loss minimization approach, candidates of Θ are assessed by the mean of a risk scoring function $R : \Theta \to \mathbb{R}$ and seeks for the one minimizing this risk function function, i.e, the one minimizes the expected loss

$$R(\theta) = \int_{\mathcal{X}\times\mathcal{Y}} \ell(y, \theta(\boldsymbol{x}))d\mathbf{P}(\boldsymbol{x}, y), \tag{1.1}$$

where $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ is the loss function, and $\ell(y, \theta(\boldsymbol{x}))$ is the loss of predicting $\theta(\boldsymbol{x})$ when observing y. It is obvious that the probability measure $\mathbf{P}(\boldsymbol{x}, y)$, which specifies the data generating process, is unknown, thus the risk (1.1) cannot be computed directly. Therefore, in practice, it is usually estimated using the empirical risk $R(\theta \,|\, \mathbf{D})$, that is

$$R(\theta \,|\, \mathbf{D}) = \sum_{n=1}^{N} \ell(y_n, \theta(\boldsymbol{x}_n)). \tag{1.2}$$

The selected model is then the one minimizing (1.2). Thus, loss minimisation approach can be in principle applied as soon as a loss function is defined [43, 91].

Maximum likelihood estimation (MLE) [32, 65] requires a well-defined likelihood function and a probabilistic hypothesis space. MLE is based on the principle (originally developed by R.A. Fisher [32]) stating that the desired probability distribution is the one that makes the observed data most likely, which means the optimal model should be the one maximizing the likelihood function [65]. This can be done by either maximizing the conditional probability $p_\theta(y \,|\, \boldsymbol{x})$, for discriminative methods, or, the joint probability $p_\theta(\boldsymbol{x}, y)$, for generative learning methods [66].

- The discriminative methods, e.g, logistic regression [21, 93], or, support vector machines (SVM) [10, 16], assume some functional form of $p_\theta(y \,|\, \boldsymbol{x})$, the conditional probability that the label $y \in \mathcal{Y}$ will be assigned to the instance $\boldsymbol{x}$, and seek for the model $\theta^* \in \Theta$ maximizing the discriminative likelihood function, i.e,

$$\theta^* = \arg\max_{\theta\in\Theta} L(\theta \,|\, \mathbf{D}) := \arg\max_{\theta\in\Theta} \prod_{n=1}^{N} p_\theta(y_n \,|\, \boldsymbol{x}_n). \tag{1.3}$$

The optimal model θ^* is then used to make inference/predictions on new instances $(\boldsymbol{t}, ?)$, typically using the expected loss minimisation, e.g, to assign for $\boldsymbol{t}$ the label $y^* \in \mathcal{Y}$ with the highest conditional probability $p_{\theta^*}(y \,|\, \boldsymbol{t})$.

- Generative learning methods, for instance, Naive Bayes [77], assume some functional form of $p_\theta(y)$ and $p_\theta(\boldsymbol{x} \,|\, y)$. With the assumption of conditional independence, which is typically made for generative models, the joint probability $p_\theta(\boldsymbol{x}, y)$ can be expressed in the factorized form as follows

$$p_\theta(\boldsymbol{x}, y) = p_\theta(y) p_\theta(\boldsymbol{x} \,|\, y) = p_\theta(y) \prod_{p=1}^{P} p_\theta(x^p \,|\, y). \tag{1.4}$$

Thus, the optimal model θ^*, which will be used to make inference, is the one maximizing the generative likelihood function, that is

$$\begin{aligned} \theta^* = \arg\max_{\theta \in \Theta} L(\theta \,|\, \mathbf{D}) : &= \arg\max_{\theta \in \Theta} \prod_{n=1}^{N} p_\theta(\boldsymbol{x}_n, y_n) \\ &= \arg\max_{\theta \in \Theta} \prod_{n=1}^{N} p_\theta(y_n) \prod_{p=1}^{P} p_\theta(x_n^p \,|\, y_n). \end{aligned} \tag{1.5}$$

Transductive learning

In a transductive learning approach, in contrast with inductive learning one, estimates for each new instance $(\boldsymbol{t}, ?)$ a potential model by using additional information related to this point [48, 50, 69, 92] (the lower path of the Figure 1.1). This means that, in a transductive approach, the training data $\mathbf{D}$ are always maintained and are used to make inference for the new instances $(\boldsymbol{t}, ?)$ (rather than using an optimal model θ^* as in inductive learning). For instance, in case of the K nearest neighbours (K-nn) method, a non-parametric classifier, for each new instance $(\boldsymbol{t}, ?)$, we extract directly from $\mathbf{D}$ a set of K nearest neighbours, denoted by $\mathbf{N}_{\boldsymbol{t}}$, and derive an optimal prediction y^* for $\boldsymbol{t}$ based on the voting scores given by training instances in $\mathbf{N}_{\boldsymbol{t}}$.

Let us remind that, in traditional learning problems, the input and output data are supposed to be precise, i.e, $(\boldsymbol{x}, y) \in \mathcal{X} \times \mathcal{Y}$. In this work, one of our interest is to investigate what could happen when either the input or the output becomes partially known, i.e, when having data of the form $(\mathbf{X}, Y) \subseteq \mathcal{X} \times \mathcal{Y}$.

1.2 Learning from partial data

The first question we look at is what happens when data becomes partial, i.e, given in form $(\mathbf{X}, Y) \subseteq \mathcal{X} \times \mathcal{Y}$, and we have to learn from them. Such situations are commonly encountered in various fields, such as biostatistics [41], agronomy [54], or economy [59]. These data can be generated by coarse or censored measurements (see e.g., [30]), anonymization techniques [29], or can be obtained from expert opinions. In particular, partially labelled data may come from easy-to-obtain high-level information. For instance, when characterizing names in subtitles to identify those characters present in an image/video [18], labeling characters with its location in word segmentation [100] or in signal segmentation [9, 56]. Another possible setting of partial data is when some features of some instances can only be partially specified, i.e, belong to intervals (or sets). This kind of data may come from imprecise measurement devices,

imperfect knowledge of an expert, or can also be the result of the summary of a huge data set.

To tackle the problem of learning from partial data, generic learning methods have to be adapted to cope with partial data, as the notion of optimal model is no longer well-defined. Two general trends in literature are:

- to adapt the criteria, for instance, likelihood [26, 27] or loss function [18, 43, 45] (e.g, the ones defined in (1.2)-(1.5)), so that the notions of optimal models are again well-defined,
- or, to consider sets of models corresponding to ways in which the data can be completed, e.g., by comparing interval-valued loss function, or by considering imprecise likelihoods [88].

Note that, in general, one may consider problems where only a part of the data is partially specified: either the labels or the features.

Partially labelled data

There are different approaches to learn from partially labelled data, i.e, data are given in the form $(\boldsymbol{x}, Y)$.

- T. Cour, et al. [18] assume that the precise value and observed partial data $\boldsymbol{x}, y, Y$ are distributed according to an (unknown) distribution $p_\theta(\boldsymbol{x}, y, Y) = p_\theta(\boldsymbol{x})p_\theta(y \,|\, \boldsymbol{x})p_\theta(Y \,|\, \boldsymbol{x}, y)$ and seek for the the distribution (model) with high conditional entropy for $p_\theta(Y \,|\, \boldsymbol{x}, y)$. The generic approach is formulated and investigated for the particular case of voting classifiers, which assign, for a given instance $\boldsymbol{x}$, a score $g_\theta(y \,|\, \boldsymbol{x})$ to each label y and select the highest scoring label $g_\theta(. \,|\, \boldsymbol{x})$. The optimal model θ^* is the candidate within Θ that minimizes the *Convex Loss for Partial Labels* (CLPL), a generalization of (1.2) with $\ell(y, \theta(\boldsymbol{x}))$ being the 0/1 loss.
- Adopting the *superset assumption*, which does not assume anything else than the observation Y being a superset of y, in [43–45], some authors propose to choose the optimal model by minimizing the *optimistic superset loss* (OSL).
- Another method to learn from partial data is *fuzzy EM* (FEM) [26, 27] which proposes to estimate the parameters of a probabilistic model based on maximizing the *observed-data likelihood* defined as the probability of the fuzzy data.

These methods differ by the choice of the likelihood/loss function and/or the prior assumptions about the incompleteness process generating partial data [19].

The first part of this work focuses on the *superset assumption* based approach [43, 45], this means we will process under a very generic assumption that whenever a feature or label is partially given, it is a superset of (i.e, covers) the true value. Yet, this approach has been detailed and justified, in both theoretical and experimental aspects, for the case of partially labelled data. Adapting the approach for the case of partially featured data is still challenging.

1.3 Active learning: missing and partial data

Missing data

In classical active learning, some observed data are complete and form the initial training data set $\mathbf{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$. The goal of active learning is to determine which new data is useful to improve the learned model.

A popular assumption in classical active learning is that it is possible to ask for the label of unlabelled data. In this work, we will concentrate on the solution where we have a set of precise data $\mathbf{D}$, and a pool $\mathbf{U}$ of unlabelled data. In this solution, several active learning approaches exist:

- The uncertainty sampling approach [55], measures how uncertain the current optimal model θ^* is about each instance within the given pool $\mathbf{U}$, using an utility score, e.g, conditional entropy, maximum conditional, margin of confidence, etc., and queries the instance with the highest uncertain score.

- The query-by-committee approach [83] assumes that a set of models $\Theta_{QBC} \subseteq \Theta$ is available and can be employed to assess the instances within the pool $\mathbf{U}$. For each unlabelled instance $(\boldsymbol{t}, ?) \in \mathbf{U}$, each member $\theta \in \Theta_{QBC}$ is then allowed to vote for its prediction $\theta(\boldsymbol{t})$. The most informative query is considered to be the instance about which they most disagree, e.g, to maximize the vote entropy or the Kullback-Leibler (KL) divergence.

- The expected model/error approach [82] has been developed upon the intuition that the learner seeks for instances that are likely to most influence the model, regardless of its true label. This approach has been highlighted to work well in empirical studies, however, can be computationally expensive if both the number of features and cardinality of the output space $\mathcal{Y}$ are very large.

These approaches assess the effect of querying each unlabelled instance, within a given pool $\mathbf{U}$, by mean of an utility score and query the instance with the highest score. Thus, they differ by the choice of utility score.

Yet, classical active learning approaches have shown advantages, including simple implementations and interpretable results, it have been debated for the lack of informing about the *reasons* for why an instance is considered uncertain, although this might be relevant for judging the usefulness of an instance. This demand comes from the fact that different sources of uncertainty could play quite different roles in specific applications [79, 85]. For instance, in active learning, Sharma and Bilgic [85] propose an evidence-based approach to active learning, in which *conflicting-evidence* uncertainty is distinguished from *insufficient-evidence* uncertainty. Experimentally, they support their conjecture that the former is more informative for an active learner than the latter, however, the uncertainty measures used by [85] are somewhat ad-hoc, and their approach is tailored for a specific learning algorithm (Naïve Bayes [77]).

Pursuing a similar purpose, in [79], authors proposed a distinction between the *epistemic*, caused by a lack of training data, and *aleatoric*, due to intrinsic randomness, uncertainty. Thus, it is reasonable to make the hypothesis that, when doing active learning, querying instances with high degrees of *epistemic* uncertainty could provide a significant improvement on the classifiers performance comparing with querying the ones of high *aleatoric* and other types of uncertainty. Furthermore, the formal model in [79] is generic as it can be, in principle, applied to any probabilistic classifier with a well-defined likelihood function. Thus, active learning methods (if can be) developed upon this building block can be applied in a broader context (compared to the ones of evidence-based approach).

Partially specified data

Classical active learning assumes either full or completely missing information, and mostly focuses on the output data. A much less studied setting is the case of partially known data, either in features or labels. Note that in this case, it is not clear that

whether we should (1) made a distinction between the (partial) training set $\mathbf{D} := \{(\mathbf{X}_n, Y_n)_{n=1}^N\}$ and pool $\mathbf{U}$, consisting of data to be queried, or (2) it is desirable to use the partial data in the learning step.

In this work, by assuming that the pool $\mathbf{U}$ is identical to the partial training set $\mathbf{D}$, we thus look at the following question: given the partial data $\mathbf{D} = \{(\mathbf{X}_n, Y_n)_{n=1}^N\}$ and some model Θ to learn, what partial data should we query (by query we mean choosing partially specified features or labels and asking its precise value to an oracle) in order to better learn the optimal model θ^*. This problem can be seen as a generalisation of the classical active learning [35, 55, 80, 81], where training instances are precisely specified while the instances in the pool $\mathbf{U}$ have precise features $\mathbf{X} := \boldsymbol{x} \in \mathcal{X}$ and completely missing labels, i,e, Y is either empty or identical to the output space $\mathcal{Y}$. The scenario where the feature values are either completely missing or precisely specified [60, 61] is also covered in our concern.

In our settings, we adopt the *superset assumption* [18, 43–45] and allow to use the partial data in the learning step. For instance, we will use the maximax approach [43–45] to make inferences. Thus, the presence of partial data can lead to the following indecision situations, where using active learning can be an efficient way to reduce the imprecision.

- When doing induction on a partial data set $\mathbf{D} = \{(\mathbf{X}_n, Y_n)\}_{n=1}^N$, it is reasonable to say that model θ is better than θ' if $L(\theta \,|\, \mathbf{d}) < L(\theta' \,|\, \mathbf{d})$, for any replacement $\mathbf{d}$ of $\mathbf{D}$. Thus there is a possibility of obtaining a set of models $\Theta^* \subseteq \Theta$ whose candidates are equivalently optimal, rather than a singleton. Yet, even if we can use either a minimin (optimistic) or a maximin (pessimistic) approach [87, 95] to learn an *optimal* model, this model is actually one candidate of Θ^* and the larger the size of Θ^*, the higher chance we pick up the wrong model. Thus if we are allowed to query some (partially specified) features or labels of some instances, we should query the data that can help to quickly reduce the set Θ^*.

- Another possible scenario is when a non-parametric model, e.g, a K nearest neighbours classifier (K-nn), is employed to make inference. In this case, we are following a transductive learning approach where the partial training data $\mathbf{D} = \{(\mathbf{X}_n, Y_n)\}_{n=1}^N$ are used directly to perform the inference step. Thus, it is quite possible that for some new instances $(\boldsymbol{t}, ?)$, we see multiple optimal predictions, i.e, a set $Y(\boldsymbol{t}) \subseteq \mathcal{Y}$ of labels that are equivalently optimal. Thus, if we can do active learning to reduce the risk of choosing wrong decision, we should query the data that can help the most to reduce $Y(\boldsymbol{t})$.

1.4 Cautious inferences

In classical supervised learning, typical (probabilistic and/or deterministic) models, once learned from the precise training data set $\mathbf{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, will provide, for each new instance $(\boldsymbol{t}, ?)$, an optimal inference or prediction in the form of a single class [7, 34, 89]. Yet, there are situations where it could be useful to make cautious inferences, in the form of set-valued, or credal, predictions when we are unsure about the optimal class to predict. This is especially true in safety-critical applications, such as medical diagnosis [28, 70] or drug discovery process [1, 31]. Cautious inference has been increasingly tackled in literature, for instances:

- A nondeterministic classifier [15] produces a set-valued prediction by invoking the principle of expected loss minimization, where the underlying cost measure combines the precision and correctness of the prediction.

- Methods based on imprecise probabilities, such as [15], augment probabilistic predictions into probability intervals or sets of probabilities, the size of which reflects the lack of information. Similar to this are methods based on confidence bands in calibration models, for instance [53, 99]. They usually control the amount of imprecision by adjusting some certainty parameters, e.g., a confidence value.

- Conformal prediction [4, 84] is another generic approach to reliable (set-valued) prediction that combines ideas from probability theory (specifically the principle of exchangeability), statistics (hypothesis testing, order statistics), and algorithmic complexity. Roughly speaking, for each new instance $(\boldsymbol{t}, ?)$, it assigns a *non-conformity* score to each candidate output. Then, considering each of these outcomes as a hypothesis, those outcomes for which the hypothesis can be rejected with high confidence are eliminated. The set-valued prediction is given by the set of outcomes that cannot be rejected.

These proposals can be seen as extensions of classification with a reject option whose prediction is either a singleton set or the entire $\mathcal{Y}$ [13]. Yet the predictive abilities of theses set-valued prediction classifiers have been studied both theoretically and experimentally. Giving the reasons for why a class should be included into or discarded from a set-value prediction seems to be challenging.

In a cautious inference approach, it is important to identify those instances for which the prediction is the most uncertain, or the less robust (i.e., for which a slight model change would change the prediction), and to find a good balance between informativeness (providing rather precise, but possibly wrong predictions) and cautiousness (predicting numerous classes probably containing the right one, but being poorly informative). It thus appears important, in this problem, to identify what make the prediction uncertain or poorly robust: is it ambiguous due to statistical variability and effects that are inherently random, or, to a lack of knowledge due to inadequate training data?

These two types of uncertainty, as mentioned in the active learning problem, are usually referred as *aleatoric* and *epistemic* [42, 79]. The distinction between *epistemic* and *aleatoric* can indeed provides insightful evidences for why we should be cautious while its complement, i.e, the strict preferences in favor of predicting one class over another/others are important when insisting on the informativeness.

Yet, the distinction *epistemic*/*aleatoric* is well-accepted in the literature on uncertainty [42] and has been very recently considered in machine learning [51, 79]. To practically determine/estimate such degrees of uncertainty may become rather complex and highly depends on the choice of the class of hypothesis and likelihood function. Thus, developing practical procedures to determine/estimate these degrees certainly benefits further applications based on this distinction. For instances, the problem of making cautious inference, considered here, or, the active learning problem highlighted in the previous Section.

1.5 Our contributions

In this work, we make the following contributions:

- In the case of transductive learning, and more precisely K nearest neighbours (K-nn) classifier, we propose to look at both the learning problem and the active learning problem from partial data, solving the first one by using a maximax

approach and the second one by proposing a querying scheme inspired by voting rules with incomplete information. This will be detailed in Chapter 2.

- Considering partial data and imprecisely-valued loss functions, we propose a generic racing approach to query partial data, in order to improve the subsequent learning step. Our proposal can also be seen as a contribution to formulate the active learning problem for partial data. This will be detailed in Chapter 3.

- In chapter 4, we differentiate two kinds of uncertainties: an aleatoric or irreducible one, that just comes from the fact that classes are mixed, and an epistemic or reducible one, that comes from the fact that we have few information about the instance. We provide estimation methods for the classical models that are logistic regression, local models and Naive Bayes. Finally, we explain how these estimates can be used to solve two problems: the one of active learning, and the one of performing cautious inferences.

Before detailing the proposals, let us note that there are two possible readings of this work: following a problem flow as sketched in the horizontal structure of Table 1.1, or going through the vertical structure for a method flow.

Problem	**Method**		
	Transductive	Inductive	
		Loss minimization	likelihood
Learning from partial data	Chapter 2		
Active learning for partial data	Chapter 2	Chapter 3	
Active learning for classical data			Chapter 4
Cautious inference			Chapter 4

TABLE 1.1: Summary of the work

Chapter 2

Transductive learning and partial data

This chapter first tackles the problem of making inference from the partially specified data and then presents active learning methods to reduce the imprecision in the inference step introduced by partially specified data.

2.1 Problem statements

The setting we consider here is when the training data set is partially specified, i.e, $\mathbf{D} = \{(\mathbf{X}_n, Y_n)\}_{n=1}^N$, and the maximax approach [43–45] is used to make inference for new (precise) instance $(\boldsymbol{t}, ?)$ (the general case when new instance can be partially specified is left as an open problem). We will process both learning and active learning problem under the *superset assumption* that is whenever a label or a feature is partially specified, the partial information covers the true value.

2.1.1 A Maximax approach for learning from partial data

Let us first recall that, in the classical loss minimization approach [43, 91], the candidates of hypothesis space Θ are assessed by the mean of a risk scoring function $R : \Theta \to \mathbb{R}$ and the chosen model is the one minimizing the expected loss

$$R(\theta) = \int_{\mathcal{X}\times\mathcal{Y}} \ell(y, \theta(\boldsymbol{x}))d\mathbf{P}(\boldsymbol{x}, y), \tag{2.1}$$

where $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ is the loss function, and $\ell(y, \theta(\boldsymbol{x}))$ is the loss of predicting $\theta(\boldsymbol{x})$ when observing y. As recalled in the introduction, in practice, it is usually estimated using the empirical risk $R(\theta \,|\, \mathbf{D})$ on the training data $\mathbf{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, that is

$$R(\theta \,|\, \mathbf{D}) = \sum_{n=1}^{N} \ell(y_n, \theta(\boldsymbol{x}_n)), \tag{2.2}$$

The selected model is then the one minimizing (2.2), that is

$$\theta^* = \arg\min_{\theta\in\Theta} R(\theta \,|\, \mathbf{D}). \tag{2.3}$$

Given the optimal model θ^*, we can simply assign for each new instance $\boldsymbol{t}$ the label candidate that minimizes the prediction loss $\ell(y, \theta^*(\boldsymbol{t}))$, i.e,

$$y^* = \arg\min_{y\in\mathcal{Y}} \ell(y, \theta^*(\boldsymbol{t})). \tag{2.4}$$

Assuming that the non-parametric K-nn classifiers [20, 97] are used to make predictions, i.e, following a transductive approach [48, 69, 92], for each new instance $\boldsymbol{t}$ its prediction is learned directly from the training data $\mathbf{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$. This mean that only the inference step (2.4) is concerned (without any learning step (2.2)-(2.3)). Denoting by $\mathbf{N}_{\boldsymbol{t}} = \{(\boldsymbol{x}_k, y_k)\}_{k=1}^K$ and $\boldsymbol{w} = (w_1, \ldots, w_k)$, the set of K nearest neighbours of $\boldsymbol{t}$ within $\mathbf{D}$ and the corresponding weight vectors, respectively, each label $y \in \mathcal{Y}$ will be given a voting score $s_{\boldsymbol{t}}(y)$ (of how likely it will be assigned for $\boldsymbol{t}$) s.t,

$$s_{\boldsymbol{t}}(y) = \sum_{k=1}^{K} w_k \mathbb{1}_{y=y_k}, \tag{2.5}$$

with $\mathbb{1}_A$ the indicator function of A ($\mathbb{1}_A = 1$ if A is true and 0 otherwise). The optimal prediction for $\boldsymbol{t}$ is thus the one maximizing (2.5), i.e,

$$y^* = \arg\max_{y \in \mathcal{Y}} s_{\boldsymbol{t}}(y). \tag{2.6}$$

The maximax approach [43–45] can be seen as a generalization of this K-nn classifier. Let us remind that in the case of general partial data, i,e, when having a training data $\mathbf{D} = \{(\mathbf{X}_n, Y_n)\}_{n=1}^N$ and a new precise instance $\boldsymbol{t}$, the *superset assumption* means that the observation X_n^p and Y_n are supersets of x_n^p and y_n, respectively. We define the set of possible replacements of $\mathbf{D}$ as follows:

$$\mathcal{D} = \big\{\mathbf{d} := \{(\boldsymbol{x}_n, y_n) \in (\mathbf{X}_n, Y_n)\}_{n=1}^N\big\}. \tag{2.7}$$

Thus, a replacement $\mathbf{d}$ of $\mathbf{D}$ is a precise data set where each partial information (either a feature or the label of an instance) in $\mathbf{D}$ is replaced by a possible precise value. For each replacement $\mathbf{d} \in \mathcal{D}$, denoting by

$$\mathbf{N}_{\boldsymbol{t}}^{\mathbf{d}} = \big\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_K, y_K)\big\}, \tag{2.8}$$

the set of K nearest (precise) neighbour instances of $\boldsymbol{t}$ in the replacement $\mathbf{d}$. Thus, the voting score that can be given for a label $y \in \mathcal{Y}$ is defined as follows

$$s_{\boldsymbol{t}}^{\mathbf{d}}(y) = \sum_{k=1}^{K} w_k \mathbb{1}_{y=y_k}, \tag{2.9}$$

Thus to minimize the *optimistic superset loss* (OSL) [43, 45] is equivalent to maximize the maximum voting score $s_{\boldsymbol{t}}^{\mathbf{d}}$ over the possible replacement $\mathbf{d} \in \mathcal{D}$, i.e, to look for

$$y^* = \arg\max_{y \in \mathcal{Y}} s_{\boldsymbol{t}}^{max}(y) := \arg\max_{y \in \mathcal{Y}} \big(\max_{\mathbf{d} \in \mathcal{D}} s_{\boldsymbol{t}}^{\mathbf{d}}(y)\big). \tag{2.10}$$

It is clear that determining the maximum scores

$$s_{\boldsymbol{t}}^{max}(y) = \max_{\mathbf{d} \in \mathcal{D}} s_{\boldsymbol{t}}^{\mathbf{d}}(y), \forall y \in \mathcal{Y}, \tag{2.11}$$

is the main task when adopting this maximax approach. This maximax approach is detailed for the scenario of set-valued labelled and precisely specified data, i.e, $\mathbf{D} = \{(\boldsymbol{x}_n, Y_n)\}_{n=1}^N$, in [44] and further justified in [18, 43, 45]. In this specific case,

the maximum score can be simply determined using counting operations, that is

$$s_{\boldsymbol{t}}^{max}(y) = \sum_{k=1}^{K} w_k \mathbb{1}_{y \in Y_k}. \tag{2.12}$$

As the nearest neighbour sets is determined based on a distance in $\mathcal{X}$, in case of partially labelled data, the possible replacements differ only by its choice of the replacement of partial labels. Determining the maximum score (2.11) is reduced to manipulate a set of set-valued labels, i,e., the set $\{Y_k\}_{k=1}^{K}$ where Y_k is the partial labels of the k-th nearest neighbour of $\boldsymbol{t}$. However, it is not necessarily the case where some features of some instances are partially specified. In this later case, the notion of nearest neighbour set $\mathbf{N}_{\boldsymbol{t}}$ is no-longer well defined. To tackle this issue, we adopt an optimistic approach, in Section 2.2, to replace the ill-known values, that requires to compute sets of possible and necessary neighbours of an instance.

2.1.2 Active learning for partial data

Let us first note that under the *superset assumption*, it is reasonable to consider that the optimal predictions learned from different replacements $\mathbf{d} \in \mathcal{D}$ (i.e, the labels maximizing the score (2.9) for at least one replacement $\mathbf{d}$) have equal possibility to be the true optimal one. Thus, in this sense, by minimizing the *optimistic superset loss* (OSL), the maximax approach assigns for each new instance $\boldsymbol{t}$ a possible optimal prediction. On the other hand, if a label $y \in \mathcal{Y}$ is the winner in all the possible replacements, it should remain to be the optimal one when having the complete precise training data, i.e, a necessary optimal label. In the specific case of partially labelled data [18, 43–45], the sets of such possible and necessary optimal labels are identical to the possible and necessary winner sets, respectively, studied in the voting procedures with incomplete preferences [5, 52, 64].

The notions of possible and necessary label sets, denoted by $\mathbf{PL}_{\boldsymbol{t}}$ and $\mathbf{NL}_{\boldsymbol{t}}$, respectively, can be easily extended for the general setting of partial data, i.e, $\mathbf{D} = \{(\mathbf{X}_n, Y_n)\}_{n=1}^{N}$. For a new instance $\boldsymbol{t}$, denoting by $y_{\boldsymbol{t}}^{\mathbf{d}}$ its corresponding optimal prediction learned from a replacement $\mathbf{d}$, we can define its possible and necessary label sets as follows:

$$\mathbf{PL}_{\boldsymbol{t}} = \{y \in \mathcal{Y} \,|\, \exists \mathbf{d} \in \mathcal{D} \text{ s.t } y = y_{\boldsymbol{t}}^{\mathbf{d}}\} \tag{2.13}$$

$$\mathbf{NL}_{\boldsymbol{t}} = \{y \in \mathcal{Y} \,|\, \forall \mathbf{d} \in \mathcal{D}, y = y_{\boldsymbol{t}}^{\mathbf{d}}\} \tag{2.14}$$

Thus, if we have to make a precise inference, we should assign for $\boldsymbol{t}$ a label $y^* \in \mathbf{PL}_{\boldsymbol{t}}$, e.g, by using the maximax approach. This means a larger size of $\mathbf{PL}_{\boldsymbol{t}}$ implies a higher chance of picking up a wrong decision, or, a higher degree of imprecision. We thus tackle the following problem: if we are allowed to query (ask for the true values of) some features or labels of some partial instances, which partial data should we query first to reduce the imprecision in the inference step? Let us remind that by querying partial data, we assume that the pool $\mathbf{U}$ is identical to the partial training set $\mathbf{D}$ in a active learning setting.

It is clear from (2.7) that the number of possible replacements $\mathbf{d} \in \mathcal{D}$ should be reduced along the querying process. Thus, the cardinality of the possible label set $\mathbf{PL}_{\boldsymbol{t}}$ (2.13) should decrease while the cardinality of the necessary label set $\mathbf{NL}_{\boldsymbol{t}}$ (2.14), in contrast, should increase when the querying process goes along. The changes of possible and necessary label sets will be considered as the potential effects in our active learning proposals.

- For the purpose of making (precise) inference, it is clear that we should look for partial data which, if they are queried (to know its precise value and update the training data set), can help to quickly reduce the cardinality of the possible label set $\mathbf{PL}_{\boldsymbol{t}}$.

- Also, at the beginning (of the querying process), it is reasonable to assume that the training data contains many partial data and there is a high chance of seeing empty necessary label sets. Furthermore, as soon as we see a non-empty empty necessary label set, i.e, $\mathbf{NL}_{\boldsymbol{t}} \neq \emptyset$, we can pick up any of them as an optimal prediction of $\boldsymbol{t}$ and making any further query is redundant. Thus seeking for a quick enlargement of the necessary label set $\mathbf{NL}_{\boldsymbol{t}}$ should be considered as another potential effect when querying partial data.

Our querying proposals developed upon these intuitions will be detailed for the case of partially labelled data in Section 2.3 and perspectives on developing similar proposals for partially featured data are given in Section 2.4.

2.2 Learning from partially featured data

We are going to detail the maximax approach for the case of interval-valued featured data (a simple yet reasonable assumption in the setting of partially featured data). Of course, the general assumption where both of training and test data can be partially featured are more reasonable and popular in practice. However, working on such a setting requires extensions in mathematical-based technique, e.g, the extreme distances between instances. We thus focus on a simpler setting that the training data can be partially featured while the test data are precisely given and leave the general case as a future work. More precisely, we consider here the setting consisting of a partially featured training data set $\mathbf{D} = \{(\mathbf{X}_n, y_n)\}_{n=1}^N$, where $\mathbf{X}_n = (X_n^1, \ldots, X_n^P)$ and $X_n^p = [a_n^p, b_n^p]$, $\forall p = 1, \ldots, P$, and precise test instances $\mathbf{T} = \{(\boldsymbol{t}_t, ?)\}_{t=1}^T$. In addition, we will only focus on the unweighted version of maximax approach while leaving the case of weighted maximax opened.

Let us remind that, the main concern when implementing the maximax approach, is to compute the maximum score $s_{\boldsymbol{t}}^{max}(y)$ (2.11) that can be assigned to each class candidate $\forall y \in \mathcal{Y}$. Our idea here is to first determine the set of possible and necessary neighbour sets, through computing interval ranks of distances, and, from that, $s_{\boldsymbol{t}}^{max}(y)$ can be derived easily using simple counting operations.

2.2.1 Determining interval ranks

Given a partial training data instance $\mathbf{X}_n \in \mathbf{D}$ and a precise instance $\boldsymbol{t}$, Groenen et al. [38] provide simple formulae to determine the imprecise distance $d(\mathbf{X}_n, \boldsymbol{t}) = \left[\underline{d}(\mathbf{X}_n, \boldsymbol{t}), \overline{d}(\mathbf{X}_n, \boldsymbol{t})\right]$ of $\mathbf{X}_n$ with respect to $\boldsymbol{t}$:

$$\overline{d}(\mathbf{X}_n, \boldsymbol{t}) = \left(\sum_{p=1}^{P} \left[|c_n^p - t^p| + r_n^p \right]^2 \right)^{1/2}, \tag{2.15}$$

$$\underline{d}(\mathbf{X}_n, \boldsymbol{t}) = \left(\sum_{p=1}^{P} \max\left[0, |c_n^p - t^p| - r_n^p\right]^2 \right)^{1/2}, \tag{2.16}$$

where $c_n^p = (b_n^p + a_n^p)/2$ and $r_n^p = (b_n^p - a_n^p)/2$, the center and width of the interval $X_n^p = [a_n^p, b_n^p]$, for $p = 1, \ldots, P$. Such interval of distances allow us to define a partial

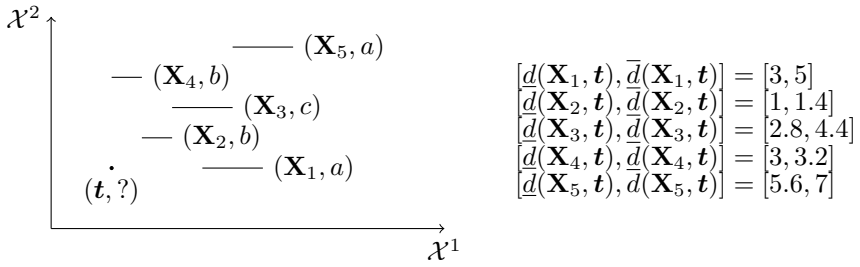


FIGURE 2.1: Example with $|\mathbf{D}| = 5$

	$\mathbf{X}_1$	$\mathbf{X}_2$	$\mathbf{X}_3$	$\mathbf{X}_4$	$\mathbf{X}_5$	$\sum_r$
$\mathbf{X}_1$	1	1	0	0	0	2
$\mathbf{X}_2$	0	1	0	0	0	1
$\mathbf{X}_3$	0	1	1	0	0	2
$\mathbf{X}_4$	0	1	0	1	0	2
$\mathbf{X}_5$	1	1	1	1	1	5
$\sum_c$	2	5	2	2	1	

TABLE 2.1: The corresponding ζ matrix for example in Figure 2.1

order on the set $\mathbf{D}$ of training instance as follows

$$\mathbf{X}_i \succeq \mathbf{X}_j \text{ if } \underline{d}(\mathbf{X}_i, \boldsymbol{t}) \geq \overline{d}(\mathbf{X}_j, \boldsymbol{t}) \tag{2.17}$$

where $\mathbf{X}_i \succeq \mathbf{X}_j$ means that $\mathbf{X}_i$ is farther than $\mathbf{X}_j$ from $\boldsymbol{t}$. As demonstrated by Patil and Taille [71, Sec. 4.1], this partial order then allows us to derive interval rank values as we have that

$$\mathbf{X}_i \succeq \mathbf{X}_j \Rightarrow r(\mathbf{X}_i) \geq r(\mathbf{X}_j),$$

where $r(\mathbf{X}_i)$ is the rank that can be assigned to $\mathbf{X}_i$.

Once the relation $\succeq$ is determined, $\mathbf{D}$ is a poset (partially ordered set) and the corresponding relation matrix, denoted by ζ, is a $N \times N$ matrix defined as

$$\zeta_{i,j} = \begin{cases} 1 \text{ if } \mathbf{X}_i \succeq \mathbf{X}_j \\ 0 \text{ otherwise.} \end{cases} \tag{2.18}$$

The results given by Theorems 1 and 2 in [71, Sec. 4.1] imply that each instance $\mathbf{X}_n \in \mathbf{D}$ can be associated to an imprecise rank $\mathbf{r}_n = [\underline{r}_n, \overline{r}_n]$, which measures how close it is to the target instance $\boldsymbol{t}$, where

$$\underline{r}_n = \sum_{j=1}^{N} \zeta_{n,j} \text{ and } \overline{r}_n = N + 1 - \sum_{j=1}^{N} \zeta_{j,n}. \tag{2.19}$$

Example 1. *Let us consider an example where* $|\mathbf{D}| = 5$ *and target instance* $\boldsymbol{t}$ *as illustrated in Figure 2.1. Using the relation (2.17), the corresponding* ζ *matrix is given in Table 2.1.*

By applying (2.19), we can easily compute the imprecise ranks of the training instances.

$$([\underline{r}_1, \overline{r}_1], [\underline{r}_2, \overline{r}_2], [\underline{r}_3, \overline{r}_3], [\underline{r}_4, \overline{r}_4], [\underline{r}_5, \overline{r}_5]) = ([2,4], [1,1], [2,4], [2,4], [5,5]). \tag{2.20}$$

2.2.2 Determining the extreme scores

Denoting by $\mathbf{R_t} = \{\mathbf{r}_n = [\underline{r}_n, \overline{r}_n] \,|\, n = 1, \ldots, N\}$ the imprecise ranks of the instances in $\mathbf{D}$, we can easily determine the sets of possible and necessary neighbours as

$$\mathbf{PN_t} = \{\mathbf{X}_n \,|\, \underline{r}_n \leq K\} \tag{2.21}$$

and

$$\mathbf{NN_t} = \{\mathbf{X}_n \,|\, \overline{r}_n \leq K\}. \tag{2.22}$$

We have that $\mathbf{X}_n \in \mathbf{NN_t}$ if it is in the set of nearest neighbours $\mathbf{X}_n \in \mathbf{N}_\mathbf{t}^\mathbf{d}$ for any replacement $\mathbf{d} \in \mathcal{D}$, while $\mathbf{X}_n \in \mathbf{PN_t}$ if $\mathbf{X}_n \in \mathbf{N}_\mathbf{t}^\mathbf{d}$ only for some replacement $\mathbf{d} \in \mathcal{D}$. For each label $y \in \mathcal{Y}$, we can then compute its minimum number of votes

$$s_\mathbf{t}^{small}(y) = \big|\{\mathbf{X}_n \in \mathbf{NN_t} \,|\, y_n = y\}\big|, \tag{2.23}$$

given by its necessary neighbours. From $s_\mathbf{t}^{small}(y)$ we can then be deduced the maximal and minimal number of votes y can receive from K nearest neighbours, according to the following formulae:

$$s_\mathbf{t}^{max}(y) = \min\left[\big|\{\mathbf{X}_n \,|\, \mathbf{X}_n \in \mathbf{PN_t}, y_n = y\}\big|, K - \sum_{y' \neq y} s_\mathbf{t}^{small}(y')\right], \tag{2.24}$$

and

$$s_\mathbf{t}^{min}(y) = \max\left[s_\mathbf{t}^{small}(y), K - \sum_{y' \neq y} s_\mathbf{t}^{max}(y')\right]. \tag{2.25}$$

These scores are simply derived from the fact that, among the K nearest neighbours, at least $s_\mathbf{t}^{small}(y)$ among them must give their votes to label y. This is proved in the next Lemma, where it is shown that $s_\mathbf{t}^{min}(y)$ and $s_\mathbf{t}^{max}(y)$ are the minimum and maximum number of votes that can be given to y over all replacements $\mathbf{d} \in \mathcal{D}$ (i.e, they are consistent with the one defined in the general setting (2.11)).

Lemma 1. *Given a number of nearest neighbours K, a target instance $\mathbf{t}$, the corresponding maximum and minimum score vectors*

$$\big(s_\mathbf{t}^{min}(y_1), \ldots, s_\mathbf{t}^{min}(y_M)\big) \text{ and } \big(s_\mathbf{t}^{max}(y_1), \ldots, s_\mathbf{t}^{max}(y_M)\big),$$

then, for any $y \in \mathcal{Y}$, we have that

$$s_\mathbf{t}^{min}(y) = \min_{\mathbf{d} \in \mathcal{D}} s_\mathbf{t}^\mathbf{d}(y) \text{ and } s_\mathbf{t}^{max}(y) = \max_{\mathbf{d} \in \mathcal{D}} s_\mathbf{t}^\mathbf{d}(y) \tag{2.26}$$

and consequently, we have that, $\forall \mathbf{d} \in \mathcal{D}$,

$$s_\mathbf{t}^{max}(y) \geq s_\mathbf{t}^\mathbf{d}(y) \geq s_\mathbf{t}^{min}(y), \forall y \in \mathcal{Y}. \tag{2.27}$$

Proof. The relation that $s_\mathbf{t}^{max}(y) = \max_{\mathbf{d} \in \mathcal{D}} s_\mathbf{t}^\mathbf{d}(y)$ can be simply proved by observing that $K - \sum_{y' \neq y} s_\mathbf{t}^{small}(y')$ bounds the number of instances that could be in the set of nearest neighbours and have y for label, while the value $|\{\mathbf{X}_n \,|\, \mathbf{X}_n \in \mathbf{PN_t}, y_n = y\}|$

simply gives the maximal number of such elements that are available within the set of possible neighbours, and that may be chosen freely to be/not be in the neighbour set, as long as they remain lower than the bound $K - \sum_{y' \neq y} s_{\boldsymbol{t}}^{small}(y')$. So, maximising this number of elements simply provides $s_{\boldsymbol{t}}^{max}(y)$.

Let us now prove that $s_{\boldsymbol{t}}^{min}(y) = \min_{\mathbf{d} \in \mathcal{D}} s_{\boldsymbol{t}}^{\mathbf{d}}(y)$, recalling that we just proved that $s_{\boldsymbol{t}}^{max}(y)$ is reachable for some replacement. We are going to focus on two cases:

Case 1: $s_{\boldsymbol{t}}^{small}(y) \geq K - \sum_{y' \neq y} s_{\boldsymbol{t}}^{max}(y')$ implies that $s_{\boldsymbol{t}}^{min}(y) = s_{\boldsymbol{t}}^{small}(y)$, hence for every replacement there is at least $s_{\boldsymbol{t}}^{small}(y)$ nearest neighbors of label y. Furthermore, $s_{\boldsymbol{t}}^{small}(y) \geq K - \sum_{y' \neq y} s_{\boldsymbol{t}}^{max}(y')$ implies that $\sum_{y' \neq y} s_{\boldsymbol{t}}^{max}(y') + s_{\boldsymbol{t}}^{small}(y) \geq K$, meaning that we can choose the remaining $K - s_{\boldsymbol{t}}^{small}(y)$ neighbours so that they vote for other labels. In other words, we can find a replacement $\mathbf{d}$ where $s_{\boldsymbol{t}}^{small}(y) = s_{\boldsymbol{t}}^{\mathbf{d}}(y)$, proving that $s^{min}(y) = \min_{\mathbf{d} \in \mathcal{D}} s_{\boldsymbol{t}}^{\mathbf{d}}(y)$ in the first case.

Case 2: $s_{\boldsymbol{t}}^{small}(y) < K - \sum_{y' \neq y} s_{\boldsymbol{t}}^{max}(y')$ implies that $s_{\boldsymbol{t}}^{min}(y) = K - \sum_{y' \neq y} s_{\boldsymbol{t}}^{max}(y')$. First note that for any replacement we cannot have $s_{\boldsymbol{t}}^{\mathbf{d}}(y) < K - \sum_{y' \neq y} s_{\boldsymbol{t}}^{max}(y')$, otherwise the set of nearest neighbour would be necessarily lower than K. $s_{\boldsymbol{t}}^{min}(y)$ then reaches this lower bound by simply taking the replacement $\mathbf{d}$ for which we have $s_{\boldsymbol{t}}^{\mathbf{d}}(y') = s_{\boldsymbol{t}}^{max}(y')$, proving that $s_{\boldsymbol{t}}^{min}(y) = \min_{\mathbf{d} \in \mathcal{D}} s_{\boldsymbol{t}}^{\mathbf{d}}(y)$ in the second case. □

2.2.3 Learning from interval-valued feature data

The maximax approach (2.10) can be then practically implemented for *interval-valued featured data* as follows:

$$\begin{aligned} \theta(\boldsymbol{t}) &= \arg\max_{y \in \mathcal{Y}} s_{\boldsymbol{t}}^{max}(y) \\ &= \arg\max_{y \in \mathcal{Y}} \left(\min \left[\left| \left\{ \mathbf{X}_n \,|\, \mathbf{X}_n \in \mathbf{PN}_{\boldsymbol{t}}, y_n = y \right\} \right|, K - \sum_{y' \neq y} s_{\boldsymbol{t}}^{small}(y') \right] \right). \end{aligned} \tag{2.28}$$

It may also happen that Equation (2.28) returns multiple labels that have the highest number of votes. We can then follow a different strategy, where we consider the result of the K-nn procedure for a peculiar replacement. Since every label receives its maximal number of votes by considering the lower distance $\underline{d}(\mathbf{X}_n, \boldsymbol{t})$, a quite simple idea is to consider the result obtained by the case of set-valued labelled data [44] when we consider the replacement $\mathbf{d}$ giving $d(\mathbf{X}_n, \boldsymbol{t}) = \underline{d}(\mathbf{X}_n, \boldsymbol{t})$ for every $\mathbf{X}_n$. The procedure to make predictions is summarized in Algorithm 1.

2.2.4 Experimental evaluation

Experiments

We run experiments on a contaminated version of 6 standard benchmark data sets described in Table 2.2 [1]. By contamination, we mean that we introduce artificially imprecision in these precise data sets. These data sets have various numbers of classes and features, but have a relatively small number of instances, for the reason that handling imprecise data is mainly problematic in such situations: when a lot of data are present, we can expect that enough precise data will exist to reach an accuracy level similar to the one of fully precise methods.

Our experimental setting is as follows: given a data set, we randomly chose a training set $\mathbf{D}$ consisting of 10% of instances and the rest (90%) as a test set $\mathbf{T}$,

[1] In this thesis, all experiments will be performed on data sets from the UCI repository http://archive.ics.uci.edu/ml/index.php

Algorithm 1: Maximax approach for interval-valued training data.

Input: **D**-imprecise training data, **T**-test set, K-number of nearest neighbours
Output: $\{p(\boldsymbol{t})|\boldsymbol{t} \in \mathbf{T}\}$-predictions

foreach $\boldsymbol{t} \in \mathbf{T}$ **do**
  compute its zeta matrix ζ through (2.15)-(2.18);
  foreach $\mathbf{X}_n \in \mathbf{D}$ **do**
    compute imprecise rank $[\underline{r}_n, \overline{r}_n]$ defined in (2.19);
  determine the $\mathbf{PN}_{\boldsymbol{t}}$ and $\mathbf{NN}_{\boldsymbol{t}}$ defined in (2.21)-(2.22);
  foreach $y \in \mathcal{Y}$ **do**
    compute $s_{\boldsymbol{t}}^{max}(y)$ through (2.23)-(2.24);
  determine $\theta(\boldsymbol{t})$ defined in (2.28);
  if $|\theta(\boldsymbol{t})| = 1$ **then**
    $p(\boldsymbol{t}) = \theta(\boldsymbol{t})$;
  else
    replace the imprecise distances by $\mathbf{d}_{\boldsymbol{t}} = \{\underline{d}(\mathbf{X}_n, \boldsymbol{t})|n = 1, \ldots, N\}$;
    determine $p(\boldsymbol{t})$ by performing classical K-nn on $\mathbf{d}_{\boldsymbol{t}}$;

Name	# instances	# features	# labels
iris	150	4	3
seeds	210	7	3
glass	214	9	6
ecoli	336	7	8
dermatology	385	34	6
vehicle	846	18	4

TABLE 2.2: Data sets used in the experiments

to limit the number of training samples. For each training instance $\boldsymbol{x}_n \in \mathbf{D}$ and each feature x_n^p, $p = 1, \ldots, P$ and $n = 1, \ldots, N$, a biased coin is flipped in order to decide whether or not the feature x_n^p will be contaminated; the probability of contamination is ϵ and we have tested different values of it ($\{0.2, 0.4, 0.6, 0.8\}$). In case x_n^p is contaminated, its precise value is transformed into an interval which can be asymmetric with respect to x_n^p.

To do that, a pair of widths $\{l_n^p, r_n^p\}$ will be generated from two Beta distributions, $Beta(\alpha_l, \beta)$ and $Beta(\alpha_r, \beta)$. To control the skewness of the generated data, we introduce a so called unbalance parameter η and assign $\{\alpha_l, \alpha_r\} = \{\beta * \eta, \beta/\eta\}$. Then the generated interval valued data is $X_n^p = [x_n^p + l_n^p(\underline{D}^p - x_n^p), x_n^p + r_n^p(\overline{D}^p - x_n^p)]$ where $\underline{D}^p = \min_n(x_n^p)$ and $\overline{D}^p = \max_n(x_n^p)$. As usual when working with Euclidean distance based K-nn, data is normalized. Then, the proposed method is used to make predictions on the test set and its accuracy is compared with the accuracy of two other cases: classical K-nn when fully precise data is given, and a basic imputation method consisting in replacing an interval-valued data X_n^p by its middle value, i.e, $x_n^p = (\underline{X}_n^p + \overline{X}_n^p)/2$. The disambiguated data is used to make predictions under the classical K-nn procedure.

Because the training set is randomly chosen and contaminated, the results maybe affected by random components. Then, for each data set, we repeat the above procedure 100 times and compute the average results. The experimental results on the data sets (described in Table 2.2) with several combinations of parameters $(K, \epsilon, \eta, \beta)$ are given in the Table 2.3, with the best results between imputation and the presented method put in bold (the precise case only serves as a reference value of the best accuracy achievable). These first results show that the difference between the two approaches is generally small. Surprisingly, this is true for all explored settings, even for skewed imprecision and high uncertainty ($\eta = 0.25$, $\epsilon = 0.8$). However, on the two data sets dermatology and vehicle, our approach really provides a significant, consistent increase of accuracy, and this even for low and balanced imprecision ($\eta = 1$, $\epsilon = 0.2$).

Conclusion

The very first experiments provided here suggest that a simple imputation method could often work as well as the presented approach, but for some data sets the maximax approach can bring a real advantage. In the future, we intend to do more experiments (varying K, increasing the number of data sets) and also try to understand the origin of the witnessed difference. However, the more interested point here is, by identifying possible and necessary neighbours, the maximax approach can also provides us with information about how uncertain our prediction is. This later advantage is instrumental in the next step we envision for this part of the work: determining which sample feature should be queried first to improve the overall algorithm accuracy, much like what we are going to investigate, in the next Section, for the case of partial labels.

2.3 Querying partially labelled data to improve the maximax approach

We are going to present our proposals for querying partially labelled data to improve the inference ability of the maximax method. We will first present a generic querying principle and a simple neighbour-based querying criteria which inspired by the high-density regions based technique in classical active learning [81], before detailing the idea highlighted in Section 2.1.2.

		iris	seeds	glass	ecoli	derma.	vehicle
$\epsilon = 0.2$, $\eta = 0.25$	Precise	91.55	84.88	49.70	75.21	82.26	53.55
	Imputation	88.93	83.79	47.30	74.40	80.20	49.45
	Maximax	**89.39**	**83.80**	**48.37**	**74.57**	**81.19**	**53.21**
$\epsilon = 0.2$, $\eta = 0.5$	Precise	91.57	85.15	50.46	74.98	81.76	53.65
	Imputation	89.07	**84.16**	47.41	**74.23**	77.41	50.35
	Maximax	**89.43**	83.92	**48.54**	74.13	**80.55**	**53.19**
$\epsilon = 0.2$, $\eta = 1$	Precise	91.35	85.39	50.49	75.11	82.13	53.65
	Imputation	88.80	**84.36**	47.48	**74.52**	75.12	50.76
	Maximax	**89.08**	84.31	**48.73**	74.35	**80.54**	**53.24**
$\epsilon = 0.4$, $\eta = 0.25$	Precise	91.44	85.31	50.34	75.33	82.26	53.54
	Imputation	87.70	83.83	46.70	**74.49**	75.87	49.88
	Maximax	**88.59**	**83.88**	**48.06**	74.02	**80.32**	**52.95**
$\epsilon = 0.4$, $\eta = 0.5$	Precise	91.14	85.26	50.20	75.47	82.04	53.50
	Imputation	87.00	**83.77**	46.31	**74.60**	75.14	49.70
	Maximax	**87.42**	83.61	**47.69**	73.87	**79.75**	**52.79**
$\epsilon = 0.4$, $\eta = 1$	Precise	91.11	85.33	50.18	75.36	82.24	53.52
	Imputation	**86.87**	**83.80**	46.17	**74.62**	73.10	49.77
	Maximax	86.59	83.52	**47.58**	73.57	**79.51**	**52.70**
$\epsilon = 0.6$, $\eta = 0.25$	Precise	92.53	84.59	50.82	74.54	81.10	53.25
	Imputation	80.46	**80.88**	43.56	**72.27**	75.38	43.41
	Maximax	**84.86**	80.85	**45.90**	69.48	**77.40**	**50.87**
$\epsilon = 0.6$, $\eta = 0.5$	Precise	92.00	85.39	50.97	74.86	81.98	53.38
	Imputation	80.06	**82.51**	44.04	**73.13**	73.28	45.10
	Maximax	**82.43**	82.06	**46.08**	70.24	**77.29**	**50.75**
$\epsilon = 0.6$, $\eta = 1$	Precise	91.66	85.57	51.01	74.83	81.97	53.46
	Imputation	80.22	**82.47**	44.37	**73.45**	68.41	46.48
	Maximax	**80.79**	82.16	**46.19**	70.47	**75.84**	**50.59**
$\epsilon = 0.8$, $\eta = 0.25$	Precise	91.62	85.46	50.74	74.97	81.91	53.40
	Imputation	79.13	**81.92**	44.34	**73.27**	69.42	44.52
	Maximax	**81.26**	81.86	**45.88**	70.19	**76.04**	**48.88**
$\epsilon = 0.8$, $\eta = 0.5$	Precise	91.27	85.29	50.85	74.92	82.08	53.44
	Imputation	78.53	81.95	44.33	**73.34**	69.00	44.18
	Maximax	**80.92**	**82.00**	**45.66**	70.17	**75.71**	**48.32**
$\epsilon = 0.8$, $\eta = 1$	Precise	91.16	85.35	50.71	75.00	82.18	53.45
	Imputation	78.58	82.04	44.25	**73.60**	66.67	44.71
	Maximax	**80.38**	**82.47**	**45.48**	70.46	**74.99**	**47.92**

Fixed parameters: $K = 3, \beta = 10$

TABLE 2.3: Experimental Results: Accuracy of classifiers (%)

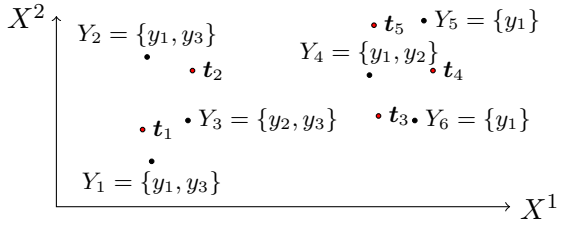


FIGURE 2.2: 3-nn classifiers

2.3.1 Generic querying scheme

General setting

In this proposal, we assume that we have a training data set $\mathbf{D} = \{(\boldsymbol{x}_n, Y_n)\}_{n=1}^N$ used to make predictions, with $\boldsymbol{x}_n \in \mathcal{X}$ the features and $Y_n \subseteq \mathcal{Y}$ are partially specified labels. Let us remind that we will adopt the *superset assumption* that is to assume that Y_n contains the true label y_n, as usual when working with partial labels [17, 18, 43–45, 100]. We also assume that we have an unlabelled target data set $\mathbf{T} = \{(\boldsymbol{t}_t, ?)\}_{t=1}^T$ that will be used to determine the partial labels to query and can be defined differently based on the usage purposes as pointed out latter in Section 2.3.3.

For a new instance $\boldsymbol{t}$ and a value K, its set of nearest neighbours in $\mathbf{D}$ is denoted by $\mathbf{N}_{\boldsymbol{t}} = \{\boldsymbol{x}_k^{\boldsymbol{t}} | k = 1, \ldots, K\}$ where $\boldsymbol{x}_k^{\boldsymbol{t}}$ is its k-th nearest neighbour. We will also say that $Y_n \in \mathbf{N}_{\boldsymbol{t}}$ if $\boldsymbol{x}_n$ is among the K nearest neighbours of a given instance $\boldsymbol{t}$. We also assume that we have a vector $\boldsymbol{w}_{\boldsymbol{t}} = (w_1^{\boldsymbol{t}}, \ldots, w_K^{\boldsymbol{t}})$ weighting each neighbour in $\mathbf{N}_{\boldsymbol{t}}$ according to its distance to the target. Similarly, for a training instance $\boldsymbol{x}_n \in \mathbf{D}$, we denote by $\mathbf{G}_{\boldsymbol{x}_n} = \{\boldsymbol{t} \,|\, \boldsymbol{x} \in \mathbf{N}_{\boldsymbol{t}}\}$ the set of target instances of which $\boldsymbol{x}_n$ is a nearest neighbour.

In the remainder of this proposal, we will use the maximax approach [43–45] to make decision, i.e, assign for each new instance $\boldsymbol{t}$, the prediction such that:

$$\theta(\boldsymbol{t}) = \arg\max_{y \in \mathcal{Y}} \sum_{\boldsymbol{x}_k^{\boldsymbol{t}} \in \mathbf{N}_{\boldsymbol{t}}} w_k^{\boldsymbol{t}} \mathbb{1}_{y \in Y_k^{\boldsymbol{t}}}. \tag{2.29}$$

The idea of the above method is to count one (weighted) vote for y whenever it is in the partial label $Y_k^{\boldsymbol{t}}$.

Example 2. *Let us consider the case illustrated in Figure 2.2, where the training data set contains* 6 *instances, the target set has* 5 *instances and the output space* $\mathcal{Y} = \{y_1, y_2, y_3\}$.

Assuming that we work with $K = 3$*, the nearest neighbours of each target instance and their associated (illustrative) weights are given in Table 2.4. And we have also:*

$$\mathbf{G}_{\boldsymbol{x}_1} = \mathbf{G}_{\boldsymbol{x}_2} = \mathbf{G}_{\boldsymbol{x}_3} = \{\boldsymbol{t}_1, \boldsymbol{t}_2\}$$
$$\mathbf{G}_{\boldsymbol{x}_4} = \mathbf{G}_{\boldsymbol{x}_5} = \mathbf{G}_{\boldsymbol{x}_6} = \{\boldsymbol{t}_3, \boldsymbol{t}_4, \boldsymbol{t}_5\}.$$

Instances $\boldsymbol{x}_5$ *and* $\boldsymbol{x}_6$ *cannot be queried because they are precise, but which label among* $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3$ *and* $\boldsymbol{x}_4$ *should be queried is not obvious. Indeed,* $\boldsymbol{x}_4$ *is involved in more decisions than the three other partial labels (as* $|\mathbf{G}_{\boldsymbol{x}_4}|$ *is greater than all other sets), but getting more information about* $\boldsymbol{x}_4$ *will not change these decisions, as the result of Equation (2.29) will not change whatever the true label of* $\boldsymbol{x}_4$*. In contrast, knowing the true label of* $\boldsymbol{x}_1$*,* $\boldsymbol{x}_2$ *or* $\boldsymbol{x}_3$ *may change our decision about* $\boldsymbol{x}_1$ *and* $\boldsymbol{x}_2$*, hence, from a decision viewpoint, querying these partial labels seems more interesting.*

$\boldsymbol{t}$	$\mathbf{N}_{\boldsymbol{t}}$	$\boldsymbol{w}_{\boldsymbol{t}}$
$\boldsymbol{t}_1$	$\{\boldsymbol{x}_1, \boldsymbol{x}_3, \boldsymbol{x}_2\}$	(0.9,0.8,0.7)
$\boldsymbol{t}_2$	$\{\boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_1\}$	(0.8,0.8,0.4)
$\boldsymbol{t}_3$	$\{\boldsymbol{x}_6, \boldsymbol{x}_4, \boldsymbol{x}_5\}$	(0.8,0.8,0.4)
$\boldsymbol{t}_4$	$\{\boldsymbol{x}_6, \boldsymbol{x}_4, \boldsymbol{x}_5\}$	(0.7,0.7,0.7)
$\boldsymbol{t}_5$	$\{\boldsymbol{x}_4, \boldsymbol{x}_5, \boldsymbol{x}_6\}$	(0.8,0.8,0.4)

TABLE 2.4: Weights and neighbours of Example 2

We are going to explore querying patterns following both intuitions (neighbour-based and ambiguity-based), but we first introduce a general querying scheme and a simple neighbour-based criteria.

A generic scheme

Our generic querying scheme follows a simple rule: for each partially labelled instance $\boldsymbol{x}_n$ and each target instance $\boldsymbol{t}$, we will define a function

$$f_{\boldsymbol{x}_n}(\boldsymbol{t}) \tag{2.30}$$

called local effect score, whose exact definition will vary for different criteria. The role of this function is to evaluate whether querying the training instance $\boldsymbol{x}_n$ can impact the result of the maximax method (2.29) for the target instance $\boldsymbol{t}$. Since we want to improve the algorithm over the whole target set, this will be done by simply summing the effect of $\boldsymbol{x}_n$ over all data in the target set $\mathbf{T}$, that is by computing

$$f_{\boldsymbol{x}_n}(\mathbf{T}) = \sum_{\boldsymbol{t} \in \mathbf{T}} f_{\boldsymbol{x}_n}(\boldsymbol{t}), \tag{2.31}$$

that we will call global score function. The chosen instance to be queried, denoted by $\boldsymbol{x}_{n^*}$, will then simply be the one with the highest effect score, or in other words

$$\boldsymbol{x}_{n^*} = \arg\max_{\boldsymbol{x}_n \in \mathbf{D}} f_{\boldsymbol{x}_n}(\mathbf{T}).$$

We will now propose different ways to define $f_{\boldsymbol{x}_n}(\boldsymbol{t})$, that will be tested in the experimental evaluation section. Since the computation of the global effect score from the local ones is straightforward, we will focus in the next sections on computing $f_{\boldsymbol{x}_n}(\boldsymbol{t})$ for a single instance. Also, we will denote by q_n the query consisting in asking the true label of $\boldsymbol{x}_n$.

Neighbour-based querying criteria

Our first idea is quite simple and consists in evaluating whether a partially labelled instance $\boldsymbol{x}_n$ is among the neighbours of the target instance $\boldsymbol{t}$, hence if $\boldsymbol{x}_n$ will participate to its classification, and how strongly $\boldsymbol{x}_n$ does so. This can be estimated by the simple function $f^{MW}_{\boldsymbol{x}_n}(\boldsymbol{t})$ as follows

$$f^{MW}_{\boldsymbol{x}_n}(\boldsymbol{t}) = \frac{w_n}{\sum_{k=1}^{K} w_k^{\boldsymbol{t}}} \tag{2.32}$$

where w_n is $w_k^{\boldsymbol{t}}$ if $\boldsymbol{x}_n$ is the k-th neighbour of $\boldsymbol{t}$, and zero otherwise. The global effect score of $\boldsymbol{x}_n$ can then be computed using Equation (2.31). In the unweighted case, this score is the number of target instances of which $\boldsymbol{x}_n$ is a neighbour. This

$f^{MW}_{\boldsymbol{x}_n}$	$\boldsymbol{t}_1$	$\boldsymbol{t}_2$	$\boldsymbol{t}_3$	$\boldsymbol{t}_4$	$\boldsymbol{t}_5$	**T**
$\boldsymbol{x}_1$	0.4	0.2	0	0	0	0.6
$\boldsymbol{x}_2$	0.3	0.4	0	0	0	0.7
$\boldsymbol{x}_3$	0.3	0.4	0	0	0	0.7
$\boldsymbol{x}_4$	0	0	0.4	0.3	0.4	**1.1**

TABLE 2.5: Effect scores obtained by using f^{MW} in Example 2

strategy is similar to the one of querying data in high-density regions in active learning techniques [81].

Table 2.5 summarizes the global effect scores for Example 2. As expected, $\boldsymbol{x}_4$ is the one that should be queried according to $f^{MW}_{\boldsymbol{x}_n}$, since it is the one participating to most decisions.

2.3.2 Indecision-based querying criteria

This Section presents other effect scores based on whether a partially labelled instance $\boldsymbol{x}_n$ introduces some ambiguity in the decision about an instance $\boldsymbol{t}$. We first define what we mean by ambiguity.

Ambiguous instance: definition

In the maximax approach [43–45], each neighbour can be seen as a (weighted) voter in favor of her preferred class. Partial labels can then be assimilated to voters providing incomplete preferences. For this reason, we will define ambiguity by using ideas issued from plurality voting with incomplete preferences [5, 52, 64]. More precisely, we will use the notions of necessary and possible winners of such a voting scheme to determine when a decision is ambiguous.

For an instance $\boldsymbol{t}$, as its set of neighbours $\mathbf{N}_{\boldsymbol{t}} = \{\boldsymbol{x}^{\boldsymbol{t}}_1, \ldots, \boldsymbol{x}^{\boldsymbol{t}}_K\}$ can be derived easily, manipulating the set of possible replacement $\mathcal{D}$ is thus reduced to handling the following set

$$\mathcal{D}_{\boldsymbol{t}} = \{\mathbf{d}_{\boldsymbol{t}} =: (y^{\boldsymbol{t}}_1, \ldots, y^{\boldsymbol{t}}_K) \,|\, y^{\boldsymbol{t}}_k \in Y^{\boldsymbol{t}}_k\},$$

i,e., the set of possible replacements of $\mathbf{N}_{\boldsymbol{t}}$ with cardinality $|\mathcal{D}_{\boldsymbol{t}}| = \prod_{k=1}^{K} |Y^{\boldsymbol{t}}_k|$. For a given replacement $\mathbf{d}_{\boldsymbol{t}}$, the corresponding winner(s) of the voting procedure is (are)

$$y^{\mathbf{d}}_{\boldsymbol{t}} = \arg\max_{y \in \mathcal{Y}} \sum_{k=1}^{K} w^{\boldsymbol{t}}_k \mathbb{1}_{y^{\boldsymbol{t}}_k = y}$$

with $w^{\boldsymbol{t}}_k$ the weight corresponding to the k-th neighbor. Let us note that the arg max can return multiple labels.

The possible and necessary label sets of $\boldsymbol{t}$ (i.e., $\mathbf{PL}_{\boldsymbol{t}}$ and $\mathbf{NL}_{\boldsymbol{t}}$ defined in (2.13)-(2.14)) can be determined as follows:

$$\mathbf{PL}_{\boldsymbol{t}} = \{y \in \mathcal{Y} \,|\, \exists \mathbf{d}_{\boldsymbol{t}} \in \mathcal{D}_{\boldsymbol{t}} \text{ s.t } y \in y^{\mathbf{d}}_{\boldsymbol{t}}\} \tag{2.33}$$

and

$$\mathbf{NL}_{\boldsymbol{t}} = \{y \in \mathcal{Y} \,|\, \forall \mathbf{d}_{\boldsymbol{t}} \in \mathcal{D}_{\boldsymbol{t}}, y \in y^{\mathbf{d}}_{\boldsymbol{t}}\}, \tag{2.34}$$

which are nothing else but the set of possible and necessary winners in social choice theory [5, 52, 64]. By definition, we have $\mathbf{NL}_{\boldsymbol{t}} \subseteq \mathbf{PL}_{\boldsymbol{t}}$. Given a target instance $\boldsymbol{t}$, we adopt the following definition of ambiguity.

y	s^{cores}	$\boldsymbol{x}_1$	$\boldsymbol{x}_2$	$\boldsymbol{x}_3$	$\boldsymbol{x}_4$	$\boldsymbol{x}_5$
y_1	s^{min}	0	0	1.2	1.4	1.2
	s^{max}	0.7	0.8	2	2.1	2
y_2	s^{min}	0	0	0	0	0
	s^{max}	1.7	1.2	0.8	0.7	0.8
y_3	s^{min}	0	0	0	0	0
	s^{smax}	2.4	2	0	0	0

TABLE 2.6: Minimal and maximal scores for Example 2

Definition 1. *A target instance* $\boldsymbol{t}$ *is called ambiguous if* $\mathbf{NL}_{\boldsymbol{t}} \neq \mathbf{PL}_{\boldsymbol{t}}$.

Let us remind that querying partial labels is equivalent to reducing the number of possible replacements $\mathbf{d}_t$. Thus, we can reduce the ambiguity of $\boldsymbol{t}$ by either reducing $\mathbf{PL}_{\boldsymbol{t}}$ or increasing $\mathbf{NL}_{\boldsymbol{t}}$, eventually getting $\mathbf{NL}_{\boldsymbol{t}} = \mathbf{PL}_{\boldsymbol{t}}$. We are going to investigate those effects and then present our querying proposals.

Ambiguous instance: computation

A first issue is how to actually compute $\mathbf{NL}_{\boldsymbol{t}}$ and $\mathbf{PL}_{\boldsymbol{t}}$. The problem of determining $\mathbf{NL}_{\boldsymbol{t}}$ is very easy [52]. However, determining $\mathbf{PL}_{\boldsymbol{t}}$ is in practice much more difficult. In the unweighted case, known results [5, 98] indicate that $\mathbf{PL}_{\boldsymbol{t}}$ can be determined in cubic (hence polynomial) time with respect to M, by solving a maximum flow problem and using the fact that when votes are (made) unitary, the solution of this flow problem is integer-valued (due to the submodularity of the constraint matrix).

However, when votes are non-unitary (when weights are different), this result does not hold anymore, and the problem appears to be NP-complete, as it can be reduced to a 3-dimensional matching problem. A refined analysis of the complexity in terms of fixed parameters (M or K) could however help to identify those cases that are harder to solve from those that remain easy (polynomial). In addition to that, in our setting we can have to evaluate the set of possible labels $\mathbf{PL}_{\boldsymbol{t}}$ a high number of times (in contrast with what happens in social choice, where $\mathbf{NL}_{\boldsymbol{t}}$ and $\mathbf{PL}_{\boldsymbol{t}}$ have to be evaluated at most a few times), hence even a cubic algorithm may have a prohibitive computational time. This is why we will provide an easy-to-compute approximation of it, denoted by $\mathbf{APL}_{\boldsymbol{t}}$. Let us first provide some definitions.

Given the set of nearest neighbours $\mathbf{N}_{\boldsymbol{t}}$, we denote by $\mathcal{Y}_{\boldsymbol{t}} = \cup_{k=1}^{K} Y_k^{\boldsymbol{t}} \subseteq \mathcal{Y}$ all labels included in the neighbours of $\boldsymbol{t}$. For each label $y \in \mathcal{Y}_{\boldsymbol{t}}$, its minimum and maximum scores are

$$s_{\boldsymbol{t}}^{min}(y) = \sum_{k=1}^{K} w_k^{\boldsymbol{t}} \mathbb{1}_{y=Y_k^{\boldsymbol{t}}} \text{ and } s_{\boldsymbol{t}}^{max}(y) = \sum_{k=1}^{K} w_k^{\boldsymbol{t}} \mathbb{1}_{y \in Y_k^{\boldsymbol{t}}},$$

respectively. For a given replacement $\mathbf{d}_{\boldsymbol{t}}$, we also denote by $s_{\boldsymbol{t}}^{\mathbf{d}}(y) = \sum_{k=1}^{K} w_k^{\boldsymbol{t}} \mathbb{1}_{y=y_k^{\boldsymbol{t}}}$ the score received by y. For any $\mathbf{d}^{\boldsymbol{t}}$, we can see that

$$s_{\boldsymbol{t}}^{min}(y) \leq s_{\boldsymbol{t}}^{\mathbf{d}}(y) \leq s_{\boldsymbol{t}}^{max}(y). \tag{2.35}$$

$s_{\boldsymbol{t}}^{min}(y)$ and $s_{\boldsymbol{t}}^{max}(y)$ are therefore the minimal and maximal scores that the candidate y can receive (thus are consistent with the generic notion defined in (2.11)).

Table 2.6 provides the score bounds obtained for the different $\boldsymbol{x}_n$ of Example 2.

From the minimal and maximal scores, we can easily get $\mathbf{NL}_{\boldsymbol{t}}$ and an approximation ($\mathbf{APL}_{\boldsymbol{t}}$) of $\mathbf{PL}_{\boldsymbol{t}}$, as indicated in the next proposition and definition.

Proposition 1. *Given target instance* $\boldsymbol{t}$*, weight* $\boldsymbol{w_t}$ *and nearest neighbour set* $\mathbf{N_t}$*, a label* $y \in \mathbf{NL_t}$ *iff*

$$s_{\boldsymbol{t}}^{min}(y) \geq s_{\boldsymbol{t}}^{max}(y^{'}), \forall y^{'} \neq y, y^{'} \in \mathcal{Y}_{\boldsymbol{t}}. \tag{2.36}$$

Proof. ($\Rightarrow$): For any pair $y, y^{'} \in \mathcal{Y}_{\boldsymbol{t}}$, $\exists \mathbf{d_t}$ s.t

$$s_{\boldsymbol{t}}^{\mathbf{d}}(y) = s_{\boldsymbol{t}}^{min}(y) \text{ and } s_{\boldsymbol{t}}^{\mathbf{d}}(y^{'}) = s_{\boldsymbol{t}}^{max}(y^{'}). \tag{2.37}$$

Just consider $\mathbf{d_t}$ s.t $y_k^{\boldsymbol{t}} = y^{'}$ if $y^{'} \in Y_k^{\boldsymbol{t}}$, and $y_k^{\boldsymbol{t}} = y$ if $Y_k^{\boldsymbol{t}} = y$. Then, if $y \in \mathbf{NL_t}$, $s_{\boldsymbol{t}}^{\mathbf{d}}(y) \geq s_{\boldsymbol{t}}^{\mathbf{d}}(y^{'})$ $\forall \mathbf{d_t}$, and in particular the one reaching $s_{\boldsymbol{t}}^{min}(y)$, $s_{\boldsymbol{t}}^{max}(y^{'})$. Combined with relation (2.37), we have

$$s_{\boldsymbol{t}}^{min}(y) \geq s_{\boldsymbol{t}}^{max}(y^{'}), \forall y^{'} \neq y, y^{'} \in \mathcal{Y}_{\boldsymbol{t}}.$$

($\Leftarrow$): Suppose $y \in \mathcal{Y}_{\boldsymbol{t}}$ satisfies condition (2.36),then

$$s_{\boldsymbol{t}}^{\mathbf{d}}(y) \geq s_{\boldsymbol{t}}^{min}(y) \geq s_{\boldsymbol{t}}^{max}(y^{'}) \geq s_{\boldsymbol{t}}^{\mathbf{d}}(y^{'}), \forall \mathbf{d_t}$$

thus $s_{\boldsymbol{t}}^{\mathbf{d}}(y) \geq s_{\boldsymbol{t}}^{\mathbf{d}}(y^{'})$ $\forall \mathbf{d_t}$. Hence $y \in \mathbf{NL_t}$. □

Definition 2. *Given target instance* $\boldsymbol{t}$*, weight* $\boldsymbol{w_t}$ *and nearest neighbour set* $\mathbf{N_t}$*, a label* $y \in \mathbf{APL_t}$ *iff*

$$s_{\boldsymbol{t}}^{max}(y) \geq \max_{y^{'} \in \mathcal{Y}_{\boldsymbol{t}}} s_{\boldsymbol{t}}^{min}(y^{'}), \forall y^{'} \neq y, y^{'} \in \mathcal{Y}_{\boldsymbol{t}}. \tag{2.38}$$

Example 3. *According to Table 2.6, the sets obtained for Example 2 with* $K = 3$ *are*

$$\mathbf{NL}_{\boldsymbol{t}_3} = \mathbf{NL}_{\boldsymbol{t}_4} = \mathbf{NL}_{\boldsymbol{t}_5} = \mathbf{APL}_{\boldsymbol{t}_3} = \mathbf{APL}_{\boldsymbol{t}_4} = \mathbf{APL}_{\boldsymbol{t}_5} = \{y_1\}$$
$$\textit{and}$$
$$\mathbf{NL}_{\boldsymbol{t}_1} = \mathbf{NL}_{\boldsymbol{t}_2} = \emptyset \quad \mathbf{APL}_{\boldsymbol{t}_1} = \mathbf{APL}_{\boldsymbol{t}_2} = \{y_1, y_2, y_3\},$$

showing, as expected, that only $\boldsymbol{t}_1, \boldsymbol{t}_2$ *are ambiguous.*

The next proposition states that $\mathbf{APL_t}$ is an outer approximation of $\mathbf{PL_t}$ (therefore not missing any possible answer) and that both coincide whenever $\mathbf{NL_t}$ is non-empty (therefore guaranteeing that using $\mathbf{APL_t}$ will not make some instance artificially ambiguous). The following Lemma which determines a condition for a label y to not be in $\mathbf{PL_t}$, and an illustrative example are necessarily to carry out the proof of the Proposition.

Lemma 2. *Given* $\boldsymbol{t}$ *,* $\boldsymbol{w_t}$ *and* $\mathbf{N_t}$*,* $y \notin \mathbf{PL_t}$ *if* $\exists$ $y^{'} \neq y$ *s.t* $s^{min}(y^{'}) > s^{max}(y)$.

Proof. If $\exists$ $y^{'} \neq y$ s.t $s_{\boldsymbol{t}}^{min}(y^{'}) > s_{\boldsymbol{t}}^{max}(y)$, then for $\forall \mathbf{d_t}$, we have

$$s_{\boldsymbol{t}}^{\mathbf{d}}(y^{'}) \geq s^{min}(y^{'}) > s_{\boldsymbol{t}}^{max}(y) \geq s_{\boldsymbol{t}}^{\mathbf{d}}(y),$$

or $s_{\boldsymbol{t}}^{\mathbf{d}}(y^{'}) > s_{\boldsymbol{t}}^{\mathbf{d}}(y)$, then y is not a possible label of $\boldsymbol{t}$. □

Thus $\mathbf{APL_t}$ is an outer approximation of $\mathbf{PL_t}$. The next example shows that $\mathbf{APL_t}$ can indeed be a strict superset of $\mathbf{PL_t}$.

Example 4. *Consider the simple unweighted case where* $K = 4$*,* $Y_1 = \{y_1, y_2\}$ *and* $Y_2 = Y_3 = Y_4 = \{y_2, y_3\}$*. As we have* $s_{\boldsymbol{t}}^{min}(y) = 0$ *and* $s_{\boldsymbol{t}}^{max}(y) > 0$ *for all labels*

$y \in \mathcal{Y}$, then $\mathbf{APL_t} = \{y_1, y_2, y_3\}$, *but* $\mathbf{PL_t} = \{y_2, y_3\}$ *(indeed, y_1 can get only one vote, while the two others will receive at least two votes).*

Proposition 2. *Given target instance $\boldsymbol{t}$, weight $\boldsymbol{w_t}$ and nearest neighbour set $\mathbf{N_t}$, the following properties hold*

A1 $\mathbf{APL_t} \supseteq \mathbf{PL_t}$

A2 *if* $\mathbf{NL_t} \neq \emptyset$, *then* $\mathbf{APL_t} = \mathbf{PL_t}$.

Proof. **(A1):** By definition of $\mathbf{PL_t}$ and $\mathbf{NL_t}$, we have that $\mathbf{PL_t} \supseteq \mathbf{NL_t}$. Lemma 2 together with the definition of $\mathbf{APL_t}$ tells us that all labels not in $\mathbf{APL_t}$ are also not in $\mathbf{PL_t}$, hence

$$\mathbf{APL_t} \supseteq \mathbf{PL_t} \supseteq \mathbf{NL_t}.$$

with Example 4 showing that $\mathbf{APL_t}$ can be a strict outer-approximation of $\mathbf{PL_t}$.

(A2): We are going to show that if $\mathbf{NL_t} \neq \emptyset$, then $\mathbf{APL_t} = \mathbf{PL_t}$. Since (**A1**) ensures $\mathbf{APL_t} \supseteq \mathbf{PL_t}$, then (**A2**) will be proved by showing that if $\mathbf{NL_t} \neq \emptyset$, then $\mathbf{APL_t} \subseteq \mathbf{PL_t}$.

Since $\mathbf{NL_t} \neq \emptyset$, then $\exists y^{'} \in \mathbf{NL_t}$ s.t $s^{min} s_{\boldsymbol{t}}^{\mathbf{d}}(y^{'}) \geq s^{max} s_{\boldsymbol{t}}^{\mathbf{d}}(y)$ for $y \neq y^{'}$. From that, we can infer that if $y \in \mathbf{APL_t}$, then we have $s^{min} s_{\boldsymbol{t}}^{\mathbf{d}}(y^{'}) = s^{max} s_{\boldsymbol{t}}^{\mathbf{d}}(y)$ for any $y^{'} \in \mathbf{NL_t}$. This means that $\exists \mathbf{d}^{\boldsymbol{t}}$ s.t $s_{\boldsymbol{t}}^{\mathbf{d}}(y) = s_{\boldsymbol{t}}^{max}(y) = s_{\boldsymbol{t}}^{min}(y^{'}) \geq s_{\boldsymbol{t}}^{\mathbf{d}}(y^{''})$ for $y^{''} \in \mathcal{Y}_{\boldsymbol{t}} \setminus \{y, y^{'}\}$. Hence y is also in $\mathbf{PL_t}$, or in other words $\mathbf{APL_t} \subseteq \mathbf{PL_t}$. □

Effect of a query on ambiguous instances

Now that we have defined how to identify an ambiguous instance, the question arises as to how we can identify queries that will help to reduce this ambiguity. This Section provides some answers by using the notions of necessary and (approximated) possible labels to define a local effect score (2.30). More precisely, the local effect score $f_{\boldsymbol{x}_n}(\boldsymbol{t})$ will take value one if a query can modify either the sets $\mathbf{PL_t}$ or $\mathbf{APL_t}$, or the set $\mathbf{NL_t}$. Additionally, as this local effect score aims at detecting whether a query can affect the final decision, it will also take value one if it can change the decision $\theta(\boldsymbol{t})$ taken by Equation (2.29). In some sense, such a strategy is close to active learning techniques aiming to identify the instances for which the decision is the most uncertain (uncertainty sampling [55], query-by-committee [83]).

To define this score, we need to know when a query q_n can potentially change the values of the possible label set $\mathbf{PL_t}$, the approximated possible label set $\mathbf{APL_t}$, the prediction set $\theta(\boldsymbol{t})$ or the necessary label set $\mathbf{NL_t}$. A first remark is that if an instance $\boldsymbol{x}_n \notin \mathbf{N_t}$ is not among the neighbours of $\boldsymbol{t}$, then a query q_n cannot change any of these values. Let us now investigate the conditions under which q_n can change the sets when $\boldsymbol{x}_n \in \mathbf{N_t}$. We first introduce some useful relations between the sets $\mathbf{PL_t}$, $\mathbf{APL_t}$, or $\mathbf{NL_t}$. We will denote by $\mathbf{PL_t^{q_n}}$, $\mathbf{APL_t^{q_n}}$, and $\mathbf{NL_t^{q_n}}$ the sets potentially obtained once $\boldsymbol{x}_n$ is queried.

In this proposal, a query will be considered interesting (i.e., having a local effect score of one) if at least one value $y \in Y_n$ can change $\mathbf{NL_t}$, $\mathbf{PL_t}$, $\mathbf{APL_t}$ or $\theta(\boldsymbol{t})$. Indeed, requiring all possible values $y \in Y_n$ to change the sets of necessary labels $\mathbf{NL_t}$, possible labels $\mathbf{PL_t}$, approximation $\mathbf{APL_t}$ or prediction set $\theta(\boldsymbol{x})$ is much too demanding, and is unlikely to happen in practice.

We will go from the cases that are the most likely to happen in practice, that is changes in $\mathbf{PL_t}$ or $\mathbf{APL_t}$, to the most unlikely cases, that is changes in $\mathbf{NL_t}$. The next proposition investigates conditions under which $\mathbf{APL_t}$ will not change.

Proposition 3. *Given target instance* $\boldsymbol{t}$*, nearest neighbour set* $\mathbf{N_t}$*, approximated possible label set* $\mathbf{APL_t}$ *of* $\boldsymbol{t}$ *and weight* $\boldsymbol{w_t}$*, query* q_n *cannot change* $\mathbf{APL_t}$ *if the two following conditions hold*

B1 *for any* $y \in \mathbf{APL_t} \setminus Y_n$*, we have*

$$s_{\boldsymbol{t}}^{max}(y) \geq \max_{y' \in Y_n} s_{\boldsymbol{t}}^{min}(y') + w_n.$$

B2 *and for any* $y \in \mathbf{APL_t} \cap Y_n$*, we have*

$$s_{\boldsymbol{t}}^{max}(y) - w_n \geq \max\left(\max_{y' \in Y_n \setminus \{y\}} s_{\boldsymbol{t}}^{min}(y') + w_n, \max_{y' \in \mathcal{Y}_{\boldsymbol{t}} \setminus Y_n} (s_{\boldsymbol{t}}^{min}(y'))\right).$$

Proof. It is clear from the definition (2.33) that $\mathbf{APL}_{\boldsymbol{t}}^{q_n} \subseteq \mathbf{APL_t}$. To show that $\mathbf{APL}_{\boldsymbol{t}}^{q_n} = \mathbf{APL_t}$ under conditions **(B1)** and **(B2)**, we will show that **(B1)** and **(B2)** imply $\mathbf{APL}_{\boldsymbol{t}}^{q_n} \supseteq \mathbf{APL_t}$. To do so, we will show that if $y \in \mathbf{APL_t}$, and y satisfies **(B1)** and **(B2)**, then $y \in \mathbf{APL}_{\boldsymbol{t}}^{q_n}$. As **(B1)** and **(B2)** partition $\mathbf{APL_t}$ in two disjoint sets (we have either $y \in \mathbf{APL_t} \setminus Y_n$ or $y \in \mathbf{APL_t} \cap Y_n$), we can treat them separately.

Also recall that if $y \in \mathbf{APL_t}$, then $s_{\boldsymbol{t}}^{max}(y) \geq \max_{y' \neq y, y' \in \mathcal{Y}_{\boldsymbol{t}}} s_{\boldsymbol{t}}^{min}(y')$.

(B1) Case $y \in \mathbf{APL_t} \setminus Y_n$: once a query q_n is done for $\boldsymbol{x}_n$, it can only increase the minimal score of one label (the true unknown one) by w_n, hence the highest increase of a minimal score is

$$\max_{y' \in Y_n} s_{\boldsymbol{t}}^{min,q_n}(y') = \max_{y' \in Y_n} s_{\boldsymbol{t}}^{min}(y') + w_n,$$

meaning that if condition **(B1)** holds, we have $s_{\boldsymbol{t}}^{max,q_n}(y) \geq \max_{y' \neq y} s_{\boldsymbol{t}}^{min,q_n}(y')$ regardless of the result of q_n, implying that $y \in \mathbf{APL}_{\boldsymbol{t}}^{q_n}$.

(B2) Case $y \in \mathbf{APL_t} \cap Y_n$: once a query q_n is done, it can decrease the maximal score of a label within Y_n of at most w_n, meaning that at worst we have $s^{max,q_n}(y) = s^{max}(y) - w_n$, while we still have

$$\max_{y' \in Y_n \setminus \{y\}} s_{\boldsymbol{t}}^{min,q_n}(y') = \max_{y' \in Y_n \setminus \{y\}} s_{\boldsymbol{t}}^{min}(y') + w_n.$$

Condition **(B2)** holding implies that $s^{max,q_n}(y) \geq \max_{y' \neq y} s_{\boldsymbol{t}}^{min,q_n}(y')$, regardless of the result of q_n, hence $y \in \mathbf{APL}_{\boldsymbol{t}}^{q_n}$.

□

According to Equation (2.38), a label $y \notin \mathbf{APL_t}$ if there is a label y' whose minimal score $s_{\boldsymbol{t}}^{min}(y')$ is higher than $s_{\boldsymbol{t}}^{max}(y)$. Proposition 3 identifies, for a label $y \in \mathbf{APL_t}$, those conditions under which an increase of the minimal score $s_{\boldsymbol{t}}^{min}(y')$ for other labels is not sufficient to become higher than $s_{\boldsymbol{t}}^{max}(y)$. Otherwise, y could get out of $\mathbf{APL_t}$.

The case of $\mathbf{PL_t}$ is more complex, and since estimating it requires to enumerate selections, the same goes for evaluating whether a query can change it. In particular, we could not find any simple-to-evaluate conditions (as those of Proposition 3) to check whether a query can change $\mathbf{PL_t}$, and we are reduced to provide the following definition. This means that evaluating whether a query can change the set $\mathbf{PL_t}$ will only be doable when K or the cardinality of partial labels neighbours will be small.

Definition 3. *Given partial label Y_n, nearest neighbours $\mathbf{N_t}$, possible label set $\mathbf{PL_t}$, set $\mathcal{Y}_t$ and weight $\boldsymbol{w_t}$, a query q_n on $\boldsymbol{x}_n \in \mathbf{N_t}$ is said to not affect $\mathbf{PL_t}$ if, for every possible answer $y \in Y_n$ of the query, we have $\mathbf{PL_t^{q_n=y}} = \mathbf{PL_t}$, where $\mathbf{PL_t^{q_n=y}}$ denotes the set $\mathbf{PL}^{q_n}$ when $Y_n = y$.*

The next proposition investigates whether or not a query can change the decision given by Equation (2.29) that we use to make predictions from partially labelled neighbours.

Proposition 4. *Given target instance $\boldsymbol{t}$, nearest neighbour set $\mathbf{N_t}$, prediction set $\theta(\boldsymbol{t})$, label set $\mathcal{Y}_t$ and weight $\boldsymbol{w_t}$, query q_n does not affect $\theta(\boldsymbol{t})$ if at least one of following conditions hold*

C1 $\theta(\boldsymbol{t}) \cap Y_n = \emptyset$.

C2 $\forall y \in \theta(\boldsymbol{t}) \cap Y_n$,

$$s_{\boldsymbol{t}}^{max}(y) - w_n > \max_{y' \in \mathcal{Y}_t \setminus \{y\}} s_{\boldsymbol{t}}^{max}(y') \tag{2.39}$$

Proof. **(C1)** Note that Equation (2.29) is equivalent to

$$\theta(\boldsymbol{t}) = \left\{ y \,|\, y = \arg\max_{y \in \mathcal{Y}_t} s_{\boldsymbol{t}}^{max}(y) \right\}.$$

It is clear that for a query q_n on $\boldsymbol{x}_n$ such that $\theta(\boldsymbol{t}) \cap Y_n = \emptyset$, then $s_{\boldsymbol{t}}^{max,q_n}(y) = s_{\boldsymbol{t}}^{max}(y)$ for all $y \in \theta(\boldsymbol{t})$, while the maximal scores for $y \notin \theta(\boldsymbol{t})$ can only decrease. Hence $\theta^{q_n}(\boldsymbol{t}) = \theta(\boldsymbol{t})$.

(C2) Since $y \in \theta(\boldsymbol{t}) \cap Y_n$, its maximal score either become $s_{\boldsymbol{t}}^{max,q_n}(y) = s_{\boldsymbol{t}}^{max}(y) - w_n$ in the worst case or is unchanged. Then Equation (2.39) guarantees that $y \in \theta^{q_n}(\boldsymbol{t})$, regardless of the true label of Y_n. □

Since classifier θ takes decisions based on the maximal number of votes a label can receive, this proposition simply identifies the cases where the reduced score of $s_{\boldsymbol{t}}^{max}(y)$ with $y \in \theta(\boldsymbol{x})$ (or non-reduction in case **C1**) cannot become smaller than another $s_{\boldsymbol{t}}^{max}(y')$. Finally, we give some conditions under which $\mathbf{NL_t}$ will not change, which may happen in practice.

Proposition 5. *Given target instance $\boldsymbol{t}$, nearest neighbour set $\mathbf{N_t}$, necessary label set $\mathbf{NL_t}$ and weight $\boldsymbol{w_t}$, then query q_n cannot change $\mathbf{NL_t}$ if the two following conditions hold*

D1 *for any $y \notin \mathbf{NL_t}$ and $y \notin Y_n$,*

$$s_{\boldsymbol{t}}^{min}(y) < \max\left(\max_{y' \neq y, y' \in \mathcal{Y}_t \setminus Y_n} s_{\boldsymbol{t}}^{max}(y'), \max_{y' \in Y_n} s_{\boldsymbol{t}}^{max}(y') - w_n, \min_{y' \in Y_n} s_{\boldsymbol{t}}^{max}(y') \right),$$

D2 *for any $y \notin \mathbf{NL_t}$ and $y \in Y_n$*

$$s_{\boldsymbol{t}}^{min}(y) + w_n < \max\left(\max_{y' \notin Y_n} s_{\boldsymbol{t}}^{max}(y'), \max_{y' \in Y_n \setminus \{y\}} s_{\boldsymbol{t}}^{max}(y') - w_n \right).$$

Proof. Note that showing that $\mathbf{NL_t^{q_n}} = \mathbf{NL_t}$ is equivalent to show that $\overline{\mathbf{NL_t}} = \overline{\mathbf{NL_t^{q_n}}}$, where $\overline{\mathbf{NL_t}}$ ($\overline{\mathbf{NL_t^{q_n}}}$) denotes the complement of $\mathbf{NL_t}$ ($\mathbf{NL_t^{q_n}}$).

It is implied by the definition of $\mathbf{NL_t}$ (2.34) that $\overline{\mathbf{NL}_t^{q_n}} \subseteq \overline{\mathbf{NL_t}}$, hence showing that under Conditions **(D1)** and **(D2)**, $\overline{\mathbf{NL}_t^{q_n}} \supseteq \overline{\mathbf{NL_t}}$ is sufficient to show the desired equality.

We will proceed as for Proposition 3, by showing that if $y \in \overline{\mathbf{NL_t}}$ and satisfies **(D1)** and **(D2)**, then $y \in \overline{\mathbf{NL}_t^{q_n}}$, which is equivalent to show that at least one label has a maximal score higher than y, i.e.,

$$s_{\boldsymbol{t}}^{min,q_n}(y) < \max_{y^{'} \neq y} s_{\boldsymbol{t}}^{max,q_n}(y^{'}). \tag{2.40}$$

Again, note that **(D1)** and **(D2)** form a partition of $\overline{\mathbf{NL_t}}$, hence, the two cases can be treated separately.

(D1) Case $y \notin \mathbf{NL_t}$ and $y \notin Y_n$: once query q_n is performed, the minimal score of y is unchanged because $y \notin Y_n$, $s_{\boldsymbol{t}}^{min,q_n}(y) = s_{\boldsymbol{t}}^{min}(y)$. The maximal scores of labels in Y_n is

$$\max_{y^{'} \in Y_n} s_{\boldsymbol{t}}^{max,q_n}(y^{'}) = \max\left(\max_{y^{'} \in Y_n} s_{\boldsymbol{t}}^{max}(y^{'}) - w_n, \min_{y^{'} \in Y_n} s_{\boldsymbol{t}}^{max}(y^{'})\right),$$

because all labels within Y_n see their maximal scores decrease, except one. The maximal scores outside $\mathbf{y}_n$ remain unchanged:

$$\max_{y^{'} \neq y, y^{'} \notin Y_n} s_{\boldsymbol{t}}^{max,q_n}(y^{'}) = \max_{y^{'} \neq y, y^{'} \notin Y_n} s_{\boldsymbol{t}}^{max}(y^{'}).$$

Then satisfying Equation (2.40) in case **(C1)** is equivalent to

$$s_{\boldsymbol{t}}^{min}(y) < \max\left(\max_{y^{'} \neq y, y^{'} \notin Y_n} s_{\boldsymbol{t}}^{max}(y^{'}), \max_{y^{'} \in Y_n} s_{\boldsymbol{t}}^{max}(y^{'}) - w_n, \min_{y^{'} \in Y_n} s_{\boldsymbol{t}}^{max}(y^{'})\right).$$

(D2) Case $y \notin \mathbf{NL_t}$ and $y \in Y_n$: after performing query q_n, the minimal score of y can increase to $s_{\boldsymbol{t}}^{min,q_n}(y) = s_{\boldsymbol{t}}^{min}(y) + w_n$. Such an increase also implies that for all other labels $y^{'} \in Y_n$ and $y^{'} \neq y$, we have $s_{\boldsymbol{t}}^{max,q_n}(y^{'}) = s_{\boldsymbol{t}}^{max}(y^{'}) - w_n$, while the maximal scores of labels outside Y_n remain unchanged. Therefore, satisfying Equation (2.40) in case **(D2)** is equivalent to

$$s_{\boldsymbol{t}}^{min}(y) + w_n < \max\left(\max_{y^{'} \notin Y_n} s_{\boldsymbol{t}}^{max}(y^{'}), \max_{y^{'} \in Y_n \setminus \{y\}} s_{\boldsymbol{t}}^{max}(y^{'}) - w_n\right).$$

□

According to Equation (2.36), a label $y \in \mathbf{NL_t}$ if its minimal score $s_{\boldsymbol{t}}^{min}(y)$ is higher than the maximal scores of all the other labels $y^{'}$. Proposition 5 identifies, for a given label $y \in \mathcal{Y}_{\boldsymbol{t}}$, the conditions under which a decrease of the maximal score $s_{\boldsymbol{t}}^{max}(y^{'})$ of the other labels is not sufficient to become lower than $s_{\boldsymbol{t}}^{min}(y)$ (otherwise, y could be included in $\mathbf{NL_t}$ after the query). Condition **D1** covers the cases where y is certainly not the true label, while condition **D2** covers the cases where it may be the true label.

		$\mathbf{APL_t}$ Prop. 3	$\theta(\boldsymbol{t})$ Prop. 4	$\mathbf{NL_t}$ Prop. 5	$\mathbf{PL_t}$ Def. 3
$\boldsymbol{t}_1$	$\boldsymbol{x}_1$	No (y_3)	No (y_2)	No (y_3)	No (y_3)
	$\boldsymbol{x}_2$	No (y_3)	Yes	Yes	Yes
	$\boldsymbol{x}_3$	No (y_2)	No (y_2)	Yes	Yes
$\boldsymbol{t}_2$	$\boldsymbol{x}_1$	Yes	No (y_2)	Yes	Yes
	$\boldsymbol{x}_2$	Yes	No (y_1)	Yes	No (y_3)
	$\boldsymbol{x}_3$	No (y_3)	No (y_3)	No (y_3)	No (y_3)

TABLE 2.7: Check for propositions for Example 2

	$f^{PL}_{\boldsymbol{x}_n}(\boldsymbol{t}_1)$	$f^{APL}_{\boldsymbol{x}_n}(\boldsymbol{t}_1)$	$f^{PL}_{\boldsymbol{x}_n}(\boldsymbol{t}_2)$	$f^{APL}_{\boldsymbol{x}_n}(\boldsymbol{t}_2)$
$\boldsymbol{x}_1$	0.4	0.4	0.2	0.2
$\boldsymbol{x}_2$	0	0.3	0.4	0.4
$\boldsymbol{x}_3$	0.3	0.3	0.4	0.4

TABLE 2.8: Ambiguity effect for Example 2

We can now use those propositions and definitions to define the two local effect scores measuring whether querying $\boldsymbol{x}_n$ can impact our decision on $\boldsymbol{t}$:

$$f^{PL}_{\boldsymbol{x}_n}(\boldsymbol{t}) = \begin{cases} 0 \text{ if Def. 3, Prop. 4, Prop. 5 hold} \\ \frac{w_n}{\sum_{k=1}^{K} w_k^{\boldsymbol{t}}} \text{ otherwise.} \end{cases} \tag{2.41}$$

and

$$f^{APL}_{\boldsymbol{x}_n}(\boldsymbol{t}) = \begin{cases} 0 \text{ if Prop. 3, Prop. 4, Prop. 5 hold} \\ \frac{w_n}{\sum_{k=1}^{K} w_k^{\boldsymbol{t}}} \text{ otherwise.} \end{cases} \tag{2.42}$$

In the next Sections, query schemes corresponding to $f^{PL}_{\boldsymbol{x}_n}(\boldsymbol{t})$ and $f^{APL}_{\boldsymbol{x}_n}(\boldsymbol{t})$ are denoted shortly by PL and APL, respectively. Since $f^{PL}_{\boldsymbol{x}_n}(\boldsymbol{t})$ uses exact information to identify the ambiguous instances, we can expect the model accuracy to improve faster by using it, yet getting $f^{PL}_{\boldsymbol{x}_n}(\boldsymbol{t})$ is computationally demanding. In practice, $f^{APL}_{\boldsymbol{x}_n}(\boldsymbol{t})$ offers a cheap approximation that can still provide good results (this will be confirmed by our experiments).

Tables 2.7 and 2.8 provide an overview of the computations associated to Example 2. Each time a proposition does not hold, we provide between parenthesis the specific answer for which it does not hold.

From Table 2.8, we can see that $f^{PL}_{\boldsymbol{x}_3}(\mathbf{T}) = f^{APL}_{\boldsymbol{x}_3}(\mathbf{T}) = 0.7$, but that $f^{PL}_{\boldsymbol{x}_2}(\mathbf{T}) = 0.4$ and $f^{APL}_{\boldsymbol{x}_2}(\mathbf{T}) = 0.7$, meaning that the two effect scores given by Equations (2.42) and (2.41) would provide different results. Finally, note that since $f^{PL}_{\boldsymbol{x}_n}(\boldsymbol{t})$ and $f^{APL}_{\boldsymbol{x}_n}(\boldsymbol{t})$ will be positive as soon as only one proposition or definition does not hold, we do not need to evaluate all of them if we know that one does not hold.

We are going to finish this section with comments on the relation between the approximation approach f^{APL} and the exact approach f^{PL}. Let us first note that there are queries that can change $\mathbf{APL_t}$, however it can not change $\mathbf{PL_t}$. In particular, such an example can be derived by focusing on the elements of $\mathbf{APL_t} \setminus \mathbf{PL_t}$. For example, if we query Y_1 in the example 4 and its true value is y_2, thus $\mathbf{PL_t}$ is remain, however we can reduce $\mathbf{APL}_{\boldsymbol{t}}^{q_1} = \{y_2, y_3\}$.

Let us remind that if a query q_n can discard a label y from the $\mathbf{PL_t}$, hence, there is no replacement $\mathbf{d} \in \mathcal{D}$ whose winners contain y (after performing q_n). Thus, y cannot

belong to $\mathbf{APL}_{\boldsymbol{t}}^{q_n}$ since the definition of $s_{\boldsymbol{t}}^{max}(y)$ (2.11) implies that y is the winner of the replacement (among the possible replacements) which gives the maximum voting score for y. Or in other word, if a query can change $\mathbf{PL}_{\boldsymbol{t}}$, it can also change $\mathbf{APL}_{\boldsymbol{t}}$.

In short, there two approaches would provide different results when the querying process goes along. However, as pointed out the next section, the improvements provided by two approaches appear to be experimentally close. Furthermore, if we have sufficient precise data, we should have $\mathbf{NL}_{\boldsymbol{t}} \neq \emptyset$ where $\mathbf{APL}_{\boldsymbol{t}} = \mathbf{PL}_{\boldsymbol{t}}$ as discussed in **A2** of the proposition 2.

2.3.3 Experimental evaluation

This Section presents the experimental setup and the results obtained with benchmark data sets which are used to illustrate the behaviour of the proposed schemes.

Experimental setup

We do experiments on "contaminated" versions of standard, precise benchmark data sets. To contaminate a given data set, we used two methods [44]:

Random Model: Each training instance is contaminated randomly with probability ϵ. In case an example $\boldsymbol{x}_n$ is contaminated, the set Y_n of candidate labels is initialized with the original label y_n, and all other labels $y^{'} \in \mathcal{Y} \setminus \{y_n\}$ are added with probability η, independently of each other.

Bayes Model: In order to take the dependencies between labels (more likely to happen in practice) into account, a second approach is used. First, a Naive Bayes classifier θ is trained using the original data (precise labels) so that each label is associated to a posterior probability $p_\theta(y \,|\, \boldsymbol{x}_n)$. As before, each training instance will be contaminated randomly with probability ϵ. In case of contamination, the true label is retained, the other labels are re-arranged according to their probabilities and the k-th label is included in the set of labels with probability $\frac{2k\eta}{|\mathcal{Y}|}$.

Note that in Bayes model, the probability $2kq/|\mathcal{Y}|$ can exceed 1 when parameter ϵ is greater than 0.5. However, this value of $2k\eta/|\mathcal{Y}|$ ensures that the expected cardinality of the partial labels, in case of contamination, is $1 + (M-1)\eta$ for both contamination models, making them comparable [44]. In practice, we lowered $2k\eta/|\mathcal{Y}|$ to 1 once it goes over it.

Results have been obtained for 15 UCI data sets described in Table 2.9. Three different values for K (3, 6 and 9) have been used for all experiments. The weight $w_k^{\boldsymbol{t}}$ for an instance $\boldsymbol{t}$ is $w_k^{\boldsymbol{t}} = 1 - (d_k^{\boldsymbol{t}})/(\sum_{j=1}^{K} d_j^{\boldsymbol{t}})$ with $d_j^{\boldsymbol{t}}$ the Euclidean distance between $\boldsymbol{x}_j^{\boldsymbol{t}}$ and $\boldsymbol{t}$. As usual when working with Euclidean distance based K-nn, data is normalized.

We use a three-folds cross-validation procedure: each data set is randomly split into 3 folds. Each fold is in turn considered as the test set, the other folds are used for the training set. The training set is contaminated according to one of the models with two combinations of (ϵ, η) parameters: $(\epsilon = 0.7,\ \eta = 0.5)$ and $(\epsilon = 0.9,\ \eta = 0.9)$, which correspond to low and high levels of partiality. The error rate is computed as the average error obtained from the 3 test sets. This process is repeated 10 times and results are also averaged. For each data set, the number of queries I has been fixed to 10% of the number of training data.

Similarly to what is done in active learning, the pool of instances to be queries $\mathbf{U}$ is identical the set of partially labelled instances, i.e, $\mathbf{U} = \{(\boldsymbol{x}_n, Y_n) \,|\, (\boldsymbol{x}_n, Y_n) \in \mathbf{D}, |Y_n| > 1\}$. The target set $\mathbf{T}$ used to assess the querying effects is defined as the data space with imperfect information, i.e, $\mathbf{T} = \{(\boldsymbol{x}_n, ?) \,|\, (\boldsymbol{x}_n, Y_n) \in \mathbf{U}\}$. Thus, we

Name	# instances	# features	# labels
iris	150	4	3
wine	178	13	3
forest	198	27	4
seeds	210	7	3
glass	214	9	6
ecoli	336	7	8
libras	360	91	15
dermatology	385	34	6
vehicle	846	18	4
vowel	990	10	11
yeast	1484	8	12
winequality	1599	11	6
optdigits	1797	64	10
segment	2300	19	7
wall-following	5456	24	4

TABLE 2.9: Data set used in the experiments

RD	MP/ACT	MW	APL	PL
$\mathcal{O}(1)$	$\mathcal{O}(T)$	$\mathcal{O}(TK)$	$\mathcal{O}(TM(M+K))$	$\mathcal{O}(TM^K)$

TABLE 2.10: Complexities of query schemes

apply the querying process using only information from training data set $\mathbf{D}$, instead of requiring a separated validation/target set.

To evaluate the efficiency of the proposed query schemes (MW, PL and APL), we compare our results with 3 baseline schemes:

- RD: a query is picked up at random from the pool;
- MP: the one with the largest partial label is picked up;
- ACT: partially labelled instances are considered as unlabeled ones and ModFF, a classical active learning scheme [49], is used to query instances. ModFF selects the queries in such a way that all target data have labelled samples at a bounded maximum distance.

The complexity of each scheme for a single query is given in Table 2.10. Note that the more computationally demanding PL scheme was only tested for the case $K = 3$.

Results

For each scheme, the error rate after querying 10% of the number of training data has been computed and the schemes have been ranked according to this error rate. The average error rates and the average ranks of the schemes over the 15 data sets are given in Table 2.11.

A Friedman test done over the ranks indicates that, in all settings, there are significant evidence that not all algorithms are equivalent (except for the random setting with low partiality that gave a p-value of 0.002, all other are below 10^{-5}). Nemenyi post-hoc test performed to identify the differences between the schemes indicate that our proposed schemes (MW, PL, APL) work almost systematically better than any baseline, with APL having a significant positive difference in pairwise tests.

K	Scheme	Random $\epsilon = 0.7$ $\eta = 0.5$	Bayes $\epsilon = 0.7$ $\eta = 0.5$	Random $\epsilon = 0.9$ $\eta = 0.9$	Bayes $\epsilon = 0.9$ $\eta = 0.9$
3	no query	36.4	42.6	77.8	78.8
	RD	30.8(4.60)	34.6(4.40)	61.6(3.73)	62.4(3.33)
	MP	29.9(3.60)	34.3(4.20)	62.4(4.13)	63.1(3.93)
	ACT	32.6(5.53)	37.5(5.73)	66.2(5.33)	66.5(5.07)
	MW	27.6(2.33)	29.9(2.73)	54.0(2.20)	54.2(1.53)
	APL	27.3(1.67)	29.4(1.67)	53.5(1.53)	54.1(1.60)
	PL	27.2(1.27)	29.3(1.33)	53.5(1.33)	54.1(1.60)
6	no query	25.7	30.4	63.3	65.6
	RD	24.0(3.40)	26.4(3.53)	44.9(3.27)	45.6(3.27)
	MP	23.7(2.00)	26.0(3.00)	45.6(3.80)	46.6(3.60)
	ACT	24.4(3.87)	27.8(4.87)	51.2(4.93)	52.7(4.93)
	MW	23.6(2.40)	25.0(2.07)	37.8(1.87)	38.9(1.73)
	APL	23.4(1.53)	24.6(1.07)	36.0(1.13)	37.5(1.20)
9	no query	25.4	27.9	53.7	57.5
	RD	24.4(2.47)	25.5(2.67)	37.0(2.93)	38.2(2.80)
	MP	24.1(1.53)	25.6(2.93)	38.3(3.73)	39.8(3.67)
	ACT	24.6(3.07)	26.5(4.40)	43.4(4.73)	45.8(4.87)
	MW	24.5(3.07)	25.8(2.47)	33.7(2.33)	34.8(2.33)
	APL	24.3(2.40)	25.6(1.73)	31.7(1.13)	33.3(1.33)

TABLE 2.11: Average error rates % (average ranks) over the 15 data sets

A noticeable exception is when the partiality is low and $K = 9$. However in this case it can be seen from Table 2.11 that all querying techniques only improve results in a very marginal way (with an accuracy gain around 1% for all methods).

A second look at Table 2.11 confirms that the proposed methods really provide an edge (in terms of average accuracy gain) in the situations where ambiguous situations are the most present, that is when:

- K is low, in which case even a few partial labels among the neighbours may lead to ambiguous situations, a fact that is much less likely when K gets higher.

- There is a large amount of partial labels, in which case increasing the value of K will have a very limited effect on the number of ambiguous cases.

Both cases are of practical interest, as even if picking a higher value of K is desirable when having low partiality, it may be computationally unaffordable.

Finally, we can notice that the Bayes contamination induces slightly more ambiguity in the data sets, as more likely classes (hence similar labels in a given region of the input space) have more chances to appear in the contaminated labels. Bayes contamination also seem somehow more realistic, as experts or labellers will have a tendency to provide sets of likely labels as partial information.

2.4 Perspectives on querying partially featured data

Yet, in the case of partially featured data and precisely featured test data, by using the specific properties of the partially ordered sets and the monotonicity of the extreme distances, we can perform the querying procedure with a manageable complexity

(polynomial time). However, no significant improvement has been observed from the experiments we did on the case of partially featured data. Let us note that in Section 2.2, we restrict ourselves to the unweighted version of the maximax. Thus, the unpromising experimental results in this very specific case is somehow insufficient to envision any conclusion about the performance the maximax in the more generic settings, e.g, to investigate the performance of the weighted version or to explore the general setting of partially featured data where both training and test data can be partially featured. On the other hand, the computations of the possible and necessary label sets, summarizing in this section, might suggest extensions/adaptations for other generic settings which are still left opened.

2.4.1 Determining the possible label set

Let us note that we will only consider the setting consists of a partially featured training data set $\mathbf{D} = \{(\mathbf{X}_n, y_n)\}_{n=1}^N$, where $\mathbf{X}_n = (X_n^1, \ldots, X_n^P)$ and $X_n^p = [a_n^p, b_n^p]$, $\forall p = 1, \ldots, P$, and precise target instances $\mathbf{T} = \{(\boldsymbol{t}_t, ?)\}_{t=1}^T$.

For a label $y_m \in \mathcal{Y}$, the relations among scores (2.27) and the definition of the possible label set (2.13) imply that y_m is a possible label ($y_m \in \mathbf{PL}_{\boldsymbol{t}}$) if and only if there is a replacement $\mathbf{d} \in \mathcal{D}$ with a score vector $\left(s_{\boldsymbol{t}}^{\mathbf{d}}(y_1), \ldots, s_{\boldsymbol{t}}^{\mathbf{d}}(y_M)\right)$ such that

$$\sum_{i=1}^{M} s_{\boldsymbol{t}}^{\mathbf{d}}(y_i) = K, \tag{2.43}$$

and

$$\min\left(s_{\boldsymbol{t}}^{\mathbf{d}}(y_m), s_{\boldsymbol{t}}^{max}(y_i)\right) \geq s_{\boldsymbol{t}}^{\mathbf{d}}(y_i) \geq s_{\boldsymbol{t}}^{min}(y_i), i = 1, \ldots, M. \tag{2.44}$$

The condition $\sum_{i=1}^M s_{\boldsymbol{t}}^{\mathbf{d}}(y_i) = K$ simply ensures that $\mathbf{d}$ is a legal replacement. The constraint (2.44) then ensures that all other labels have a score lower than $s_{\boldsymbol{t}}^{\mathbf{d}}(y_m)$ for the replacement $\mathbf{d}$ (note that $\min(s_{\boldsymbol{t}}^{\mathbf{d}}(y_m), s_{\boldsymbol{t}}^{max}(y_m)) = s_{\boldsymbol{t}}^{\mathbf{d}}(y_m)$), and that their scores are bounded by Eq. (2.27).

The question is now to know whether we can instantiate such a vector making a winner of y_m. To achieve this task, we will first maximise its score, such that $s_{\boldsymbol{t}}^{\mathbf{d}}(y_m) = s_{\boldsymbol{t}}^{max}(y_m)$. The scores of all other labels y_i is also lower-bounded by $s_{\boldsymbol{t}}^{min}(y_i)$, meaning that among the K neighbours we choose in $\mathbf{d}$, only $K - s_{\boldsymbol{t}}^{max}(y_m) - \sum_{i=1, i\neq m}^M s_{\boldsymbol{t}}^{min}(y_i)$ remain to be fixed in order to specify the score vector. Then we can focus on the relative difference between $s^{min}(y_i)$ and the additional number of chosen neighbours voting for y_i. Solving the problem defined by Eqs. (2.43), (2.44) is equivalent to determine a score vector $(w(y_1), \ldots, w(y_{m-1}), w(y_{m+1}), \ldots, w(y_M))$ with $w(y_i) = s_{\boldsymbol{t}}^{\mathbf{d}}(y_i) - s_{\boldsymbol{t}}^{min}(y_i)$, $\forall i \neq m$, s.t.

$$\sum_{i=1, i\neq m}^{M} w(y_i) = K - s_{\boldsymbol{t}}^{max}(y_m) - \sum_{i=1, i\neq m}^{M} s_{\boldsymbol{t}}^{min}(y_i), \tag{2.45}$$

$$\min\left(s_{\boldsymbol{t}}^{max}(y_m), s_{\boldsymbol{t}}^{max}(y_i)\right) - s_{\boldsymbol{t}}^{min}(y_i) \geq w(y_i) \geq 0, \forall i \neq m. \tag{2.46}$$

Eq. (2.45) again ensures that the replacement is a legal one (the number of neighbours sums up to K), and Eq. (2.46) ensures that y_m is a winning label. Also note that if $\exists y_i \in \mathcal{Y} \setminus \{y_m\}$ s.t $s_{\boldsymbol{t}}^{max}(y_m) < s_{\boldsymbol{t}}^{min}(y_i)$, then there no chance for y_m to be a possible label.

We will now give a proposition allowing to determine in an easy way if a label belongs to the set of possible labels.

Proposition 6. *Given the number of nearest neighbours K, a target instance $\boldsymbol{t}$, its corresponding maximum and minimum score vectors $\left(s_{\boldsymbol{t}}^{min}(y_1),\ldots,s_{\boldsymbol{t}}^{min}(y_M)\right)$ and $\left(s_{\boldsymbol{t}}^{max}(y_1),\ldots,s_{\boldsymbol{t}}^{max}(y_M)\right)$. Assuming that $s_{\boldsymbol{t}}^{max}(y_m) \geq s_{\boldsymbol{t}}^{min}(y_i)$, for $\forall y_i \in \mathcal{Y} \setminus \{y_m\}$, then y_m is a possible label if and only if*

$$K \leq s_{\boldsymbol{t}}^{max}(y_m) + \sum_{i=1,i\neq m}^{M} \min\left(s_{\boldsymbol{t}}^{max}(y_m), s_{\boldsymbol{t}}^{max}(y_i)\right). \tag{2.47}$$

Proof. ($\Rightarrow$) Let us prove that y_m being a possible label implies (2.47). First, if $y_m \in \mathbf{PL}_{\boldsymbol{t}}$ and $\mathbf{d}$ is a legitimate replacement, we have that

$$w(y_i) \leq \min\left(s_{\boldsymbol{t}}^{max}(y_m), s_{\boldsymbol{t}}^{max}(y_i)\right) - s_{\boldsymbol{t}}^{min}(y_i),\ \forall i \neq m \tag{2.48}$$

otherwise y_m would not be a winner, or we would give a higher score to y_i than it actually can get (we would have $s_{\boldsymbol{t}}^{\mathbf{d}}(y_i) > s_{\boldsymbol{t}}^{max}(y_i)$). Since for any replacement we have that Eq. (2.45) must be satisfied, we have necessarily

$$K - s_{\boldsymbol{t}}^{max}(y_m) - \sum_{i=1,i\neq m}^{M} s_{\boldsymbol{t}}^{min}(y_i) = \sum_{i=1,i\neq m}^{M} w(y_i).$$

If we replace $w(y_i)$ by its upper bound (2.48), we get the following inequality

$$\begin{aligned} K - s_{\boldsymbol{t}}^{max}(y_m) - \sum_{i=1,i\neq m}^{M} s_{\boldsymbol{t}}^{min}(y_i) \leq & \sum_{i=1,i\neq m}^{M} \min\left(s_{\boldsymbol{t}}^{max}(y_m), s_{\boldsymbol{t}}^{max}(y_i)\right) \\ & - \sum_{i=1,i\neq m}^{M} s_{\boldsymbol{t}}^{min}(y_i), \end{aligned}$$

that is equivalent to the relation

$$K \leq s_{\boldsymbol{t}}^{max}(y_m) + \sum_{i=1,i\neq m}^{M} \min\left(s_{\boldsymbol{t}}^{max}(y_m), s_{\boldsymbol{t}}^{max}(y_i)\right).$$

($\Leftarrow$) Let us now show that if the conditions given by Eqs. (2.45)-(2.46) are satisfied, then $y_m \in \mathbf{PL}_{\boldsymbol{t}}$. First remark that, once we have assigned the maximal score to y_m and the minimal ones to the other labels, there remain

$$K - s_{\boldsymbol{t}}^{max}(y_m) - \sum_{i=1,i\neq m}^{M} s_{\boldsymbol{t}}^{min}(y_i)$$

neighbours to choose from. We also know from (2.46) that at most

$$\sum_{i=1,i\neq m}^{M} \left[\min\left(s_{\boldsymbol{t}}^{max}(y_m), s_{\boldsymbol{t}}^{max}(y_i)\right) - s_{\boldsymbol{t}}^{min}(y_i)\right]$$

neighbours can still be affected to other labels than y_m without making it a loser. Clearly, if

$$K - s_{\boldsymbol{t}}^{max}(y_m) - \sum_{i=1, i\neq m}^{M} s_{\boldsymbol{t}}^{min}(y_i) \leq \sum_{i=1, i\neq m}^{M} \left[\min\left(s_{\boldsymbol{t}}^{max}(y_m), s_{\boldsymbol{t}}^{max}(y_i)\right) - s_{\boldsymbol{t}}^{min}(y_i)\right],$$

we can reach the number of K neighbours without making y_m a loser, or inversely letting y_m be a winner for the chosen replacement, meaning that $y_m \in \mathbf{PL}_{\boldsymbol{t}}$. □

Example 5. *Let us continue with the data set in Example 1 with value $K = 3$. From Table 2.1 and the interval ranks (2.20), we can see that*

$$\mathbf{PN}_{\boldsymbol{t}} = \{(\mathbf{X}_1, a), (\mathbf{X}_2, b), (\mathbf{X}_3, c), (\mathbf{X}_4, b)\}, \mathbf{NN}_{\boldsymbol{t}} = \{(\mathbf{X}_2, b)\}.$$

Then the maximum and minimum scores for all the labels are

$$\begin{aligned}(s_{\boldsymbol{t}}^{min}(a), s_{\boldsymbol{t}}^{min}(b), s_{\boldsymbol{t}}^{min}(c)) &= (0, 1, 0)\\ (s_{\boldsymbol{t}}^{max}(a), s_{\boldsymbol{t}}^{max}(b), s_{\boldsymbol{t}}^{max}(c)) &= (1, 2, 1).\end{aligned}$$

We will now determine whether a given label in $\mathcal{Y} = \{a, b, c\}$ is a possible label. For label a, we have that

$$s_{\boldsymbol{t}}^{max}(a) + \min\left(s_{\boldsymbol{t}}^{max}(a), s_{\boldsymbol{t}}^{max}(b)\right) + \min\left(s_{\boldsymbol{t}}^{max}(a), s_{\boldsymbol{t}}^{max}(c)\right) = 1 + 1 + 1 = 3 \geq K,$$

hence $a \in \mathbf{PL}_{\boldsymbol{t}}$. The same procedure applied to b and c gives the result $\mathbf{PL}_{\boldsymbol{t}} = \{a, b, c\}$.

2.4.2 Determining the necessary label set

Let us now focus on characterizing the set $\mathbf{NL}_{\boldsymbol{t}}$ defined in (2.14). The following propositions gives a very easy way to determine it, by simply comparing the minimum score of a given label y_m to the maximal scores of the others.

Proposition 7. *Given the maximum and minimum scores $\left(s_{\boldsymbol{t}}^{min}(y_1), \ldots, s_{\boldsymbol{t}}^{min}(y_M)\right)$ and $\left(s_{\boldsymbol{t}}^{max}(y_1), \ldots, s_{\boldsymbol{t}}^{max}(y_M)\right)$, then a given label y_m is a necessary label if and only if*

$$s_{\boldsymbol{t}}^{min}(y_m) \geq s_{\boldsymbol{t}}^{max}(y_i), \forall i \neq m. \tag{2.49}$$

Proof. ($\Rightarrow$) We proceed by contradiction. Assuming that $\exists\ y_m \in \mathbf{NL}_{\boldsymbol{t}}$ and $\exists\ y_i \in \mathcal{Y}$ where $s_{\boldsymbol{t}}^{min}(y_m) < s_{\boldsymbol{t}}^{max}(y_i)$, we show that we can always find a replacement $\mathbf{d} \in \mathcal{D}$ s.t $s_{\boldsymbol{t}}^{\mathbf{d}}(y_m) < s_{\boldsymbol{t}}^{\mathbf{d}}(y_i)$, or in other words, $\exists\ \mathbf{d} \in \mathcal{D}$ s.t $y_m \notin y_{\boldsymbol{t}}^{\mathbf{d}}$, and therefore y_m is not necessary. Let us consider the two cases

1. $K - \sum_{j\neq m} s_{\boldsymbol{t}}^{max}(y_j) \geq s_{\boldsymbol{t}}^{small}(y_m)$, then for $\forall j \neq m$, we give its the maximum score s.t $s_{\boldsymbol{t}}^{\mathbf{d}}(y_j) = s_{\boldsymbol{t}}^{max}(y_j)$ and give y_m the score $s_{\boldsymbol{t}}^{\mathbf{d}}(y_m) = K - \sum_{j\neq m} s_{\boldsymbol{t}}^{max}(y_j)$. Then it is clear that

$$s_{\boldsymbol{t}}^{\mathbf{d}}(y_m) = K - \sum_{j\neq m} s_{\boldsymbol{t}}^{max}(y_j) = s_{\boldsymbol{t}}^{min}(y_m) < s_{\boldsymbol{t}}^{max}(y_i) = s_{\boldsymbol{t}}^{\mathbf{d}}(y_i).$$

2. $K - \sum_{j\neq m} s_{\boldsymbol{t}}^{max}(y_j) < s_{\boldsymbol{t}}^{small}(y_m)$, then we give y_m a score $s_{\boldsymbol{t}}^{\mathbf{d}}(y_m) = s_{\boldsymbol{t}}^{small}(y_m)$ and give y_i a score $s_{\boldsymbol{t}}^{\mathbf{d}}(y_i) = s_{\boldsymbol{t}}^{smax}(y_i)$. As we have

$$K < \sum_{j\neq\{m,i\}} s_{\boldsymbol{t}}^{max}(y_j) + s_{\boldsymbol{t}}^{small}(y_m) + s_{\boldsymbol{t}}^{max}(y_m)$$

by assumption, we can choose $K - s_{\boldsymbol{t}}^{small}(y_m) - s_{\boldsymbol{t}}^{smax}(y_i)$ nearest neighbours from at most $\sum_{j\neq\{m,i\}} s_{\boldsymbol{t}}^{max}(y_j)$ possible nearest neigbours whose labels are not y_m or Y_n. In such a replacement we have $s_{\boldsymbol{t}}^{\mathbf{d}}(y_m) < s_{\boldsymbol{t}}^{\mathbf{d}}(y_i)$.

($\Leftarrow$) We are going to prove that (2.49) implies that the label $y_m \in \mathbf{NL}_{\boldsymbol{t}}$ is necessary. Let us first note that

$$\min_{\mathbf{d}\in\mathcal{D}} s_{\boldsymbol{t}}^{\mathbf{d}}(y_m) = s_{\boldsymbol{t}}^{min}(y_m) \text{ and } \max_{\mathbf{d}\in\mathcal{D}} s_{\boldsymbol{t}}^{\mathbf{d}}(y_i) = s_{\boldsymbol{t}}^{max}(y_i), \forall i \neq m,$$

then (2.49) ensures that, for any replacement $\mathbf{d} \in \mathcal{D}$,

$$s_{\boldsymbol{t}}^{\mathbf{d}}(y_m) \geq \min_{\mathbf{d}\in\mathcal{D}}(s_{\boldsymbol{t}}^{\mathbf{d}}(y_m)) \geq \max_{\mathbf{d}\in\mathcal{D}}(s_{\boldsymbol{t}}^{\mathbf{d}}(y_i)) \geq s_{\boldsymbol{t}}^{\mathbf{d}}(y_i), \forall i \neq m,$$

which is sufficient to get the proof. □

Example 6. *Consider the data set given in Example 5 with the maximum and minimum scores of the labels are*

$$(s_{\boldsymbol{t}}^{min}(a), s_{\boldsymbol{t}}^{min}(b), s_{\boldsymbol{t}}^{min}(c)) = (0, 1, 0)$$
$$(s_{\boldsymbol{t}}^{max}(a), s_{\boldsymbol{t}}^{max}(b), s_{\boldsymbol{t}}^{max}(c)) = (1, 2, 1).$$

Then (2.49) implies that the necessary label set $\mathbf{NL}_{\boldsymbol{t}} = \{b\}$.

2.5 Conclusion

Our first contribution in this Chapter is an implementation of the maximax approach for the case of partially featured data. Our implementation is computationally tractable and the first experiments indicate that there are cases when the maximax approach can bring a real advantage. Thus, motivating further works on broadening the applications of the maximax approach. Let us note that, we have focused on the setting where only training data are partially specified. Yet, developing similar decision rules for the generic setting, i.e, both training and test data are partially featured, could be a potential direction. Detailing it could be complicated since defining the partial order (2.17) is still a challenge. One reason is that if we have a partially featured test instance $\boldsymbol{t}$, we are no-longer allowed to freely choose and compare the possible positions of its neighbours.

Considering the active learning problem, we have proposed two querying schemes, both based on the computation of an effect score quantifying the impact of a disambiguation on the final result, to query partially labelled data. Our first strategy (neighbour-based) consists in selecting an instance when it is involved in many decisions. A more refined strategy (indecision-based) consists in selecting an instance when it can potentially reduce the ambiguity of one or several decisions. This second strategy is more complex from a computational point of view, and we have therefore proposed an approximate scheme leading to very close performance. The experiments have shown that the accuracy of the maximax method is significantly improved by querying partial label instances and that indecision-based querying strategies are the best-performing schemes.

Yet, our attempt on developing a similar querying scheme for the specific setting where only training data can be partially featured has not provided any significant improvement. The perspectives we presented could be useful for further tackling both learning and active learning problem in the generic setting where both training and

test data are partially featured (and the weighted version of Maximax is employed to make prediction). Thus, there are at least two open issues, in order to completely tackle these problems:

1. to investigate the decision rules for the generic setting where both training and test data can be imprecise.

2. to develop efficient subsequent techniques to determine the possible and necessary label sets and its potential changes when the querying process goes along.

Chapter 3

Racing Algorithms

This Chapter focuses on imprecision modeling in the problem of learning from partially specified data. We first generalise the loss function to cope with partial data and highlight the potential issue of obtaining multiple optimal models, i.e, a set of undominated models. The size of this undominated model set will be considered as a degree of imprecision due to the presence of partial data. We thus focus on developing active learning schemes to identify the partially specified data that should be queried to quickly reduce the undominated model set. We are going to present a generic querying scheme inspired by the racing algorithms and then implement it for two specific settings: binary SVM and decision trees.

3.1 Loss function and expected risk for partial data

Let us remind that, in classical supervised setting, the goal of the learning approach is to extract a model $\theta^* : \mathcal{X} \to \mathcal{Y}$ within a set Θ of models from a data set $\mathbf{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$. The empirical risk $R(\theta \,|\, \mathbf{D})$ associated to a model θ is then evaluated as

$$R(\theta \,|\, \mathbf{D}) = \sum_{n=1}^{N} \ell(y_n, \theta(\mathbf{x}_n)), \tag{3.1}$$

where $\ell(y_n, \theta(\boldsymbol{x}_n))$ is the loss of predicting $\theta(\boldsymbol{x}_n)$ when observing y_n. The selected model is then the one that minimizes (3.1), that is

$$\theta^* = \arg\min_{\theta \in \Theta} R(\theta \,|\, \mathbf{D}). \tag{3.2}$$

Another way to see the model selection problem is to say that a model θ_l is better than θ_k (denoted $\theta_l \succ \theta_k$) if

$$R(\theta_k \,|\, \mathbf{D}) - R(\theta_l \,|\, \mathbf{D}) > 0, \tag{3.3}$$

or, in other words, if the risk of θ_l is lower than the risk of θ_k.

In this proposal, we are interested in a more general case where data is potentially only partially known, that is where general samples are of the kind $(\mathbf{X}_n, Y_n) \subseteq \mathcal{X} \times \mathcal{Y}$. In such a case, Equations (3.1), (3.2) and (3.3) are no longer well-defined, and there are different ways to extend them. Two of the most common ways to extend them is either to use a minimin (optimistic) [44] or a minimax (pessimistic) approach. That

is, if we extend Equation (3.1) to a lower bound

$$\begin{aligned}\underline{R}(\theta \,|\, \mathbf{D}) &= \inf_{(\boldsymbol{x}_n, y_n) \in (\mathbf{X}_n, Y_n)} \sum_{n=1}^{N} \ell(y_n, \theta(\boldsymbol{x}_n)) \\ &= \sum_{n=1}^{N} \inf_{(\boldsymbol{x}_n, y_n) \in (\mathbf{X}_n, Y_n)} \ell(y_n, \theta(\boldsymbol{x}_n)) := \sum_{n=1}^{N} \underline{\ell}(Y_n, \theta(\mathbf{X}_n))\end{aligned} \tag{3.4}$$

and an upper bound

$$\begin{aligned}\overline{R}(\theta \,|\, \mathbf{D}) &= \sup_{(\boldsymbol{x}_n, y_n) \in (\mathbf{X}_n, Y_n)} \sum_{n=1}^{N} \ell(y_n, \theta(\boldsymbol{x}_n)) \\ &= \sum_{n=1}^{N} \sup_{(\boldsymbol{x}_n, y_n) \in (\mathbf{X}_n, Y_n)} \ell(y_n, \theta(\boldsymbol{x}_n)) := \sum_{n=1}^{N} \overline{\ell}(Y_n, \theta(\mathbf{X}_n))\end{aligned} \tag{3.5}$$

then the optimal minimin θ^*_{mm} and minimax θ^*_{mM} models are

$$\theta^*_{mm} = \arg\min_{\theta \in \Theta} \underline{R}(\theta \,|\, \mathbf{D}) \quad \text{and} \quad \theta^*_{mM} = \arg\min_{\theta \in \Theta} \overline{R}(\theta \,|\, \mathbf{D}).$$

The minimin approach usually assumes that data are distributed according to the model, and tries to find the best data replacement (or disambiguation) combined with the best possible model [43]. Conversely, the minimax approach assumes that data are distributed in the worst possible way, and selects the model performing the best in the worst situation, thus guaranteeing a minimum performance of the model [39]. However, such an approach, due to its conservative nature, may lead to sub-optimal models. When having to choose a preferred model in the race, we will follow the optimistic approach, that is also in line with the idea of racing algorithms.

However, in this proposal, we are not primarily interested into learning a single model from partial data, but we want to determine which partial data makes the potentially best models incomparable, in order to complete such data through queries. To define such a set of potentially optimal models, we will say that a model θ_l is better than θ_k (still denoted $\theta_l \succ \theta_k$) if

$$\underline{R}(\theta_{k-l} \,|\, \mathbf{D}) = \inf_{(\mathbf{x}_n, y_n) \in (\mathbf{X}_n, Y_n)} \left[R(\theta_k \,|\, \mathbf{D}) - R(\theta_l \,|\, \mathbf{D}) \right] > 0, \tag{3.6}$$

which is a direct extension of Equation (3.3). That is, $\theta_l \succ \theta_k$ if and only if it is better under every possible precise instances $(\boldsymbol{x}_n, y_n)$ consistent with the partial instances $(\mathbf{X}_n, Y_n)$. Such an approach is similar to decision rules used, for instance, in imprecise probability [87]. We can then denote by

$$\Theta^* = \{\theta \in \Theta \,|\, \nexists \theta' \in \Theta \, s.t. \, \theta' \succ \theta\} \tag{3.7}$$

the set of undominated models within Θ, that is the set of models that are maximum with respect to the partial order $\succ$.

Example 7. *Figure 3.1 illustrates a situation where $\mathcal{Y}$ consists of two different classes (gray and white), and $\mathcal{X}$ of two dimensions. Only imprecise data are numbered. Squares are assumed to have precise features. Points 1, 2 and 3 are imprecise with respect to their second feature. Shaded squares (points 4 and 5) have unknown labels. Assuming that $\Theta = \{\theta_1, \theta_2\}$ (the models could be decision stumps, i.e, one-level decision trees [76]), we would have that $\theta_2 = \theta^*_{mM}$ is the minimax model and $\theta_1 = \theta^*_{mm}$*

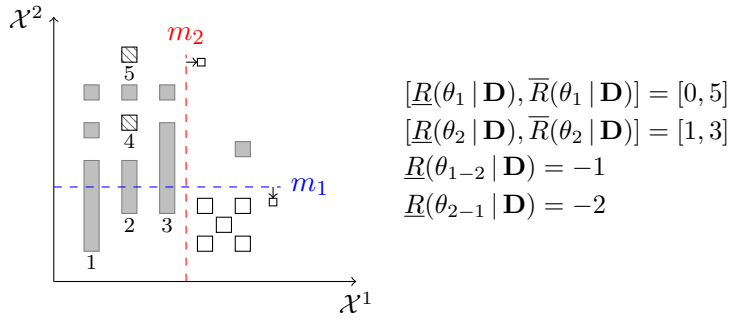


FIGURE 3.1: Illustration of partial data and competing models

the minimin one. The two models would however be incomparable according to (3.6)*, hence* $\Theta^* = \Theta$ *in this case, and the minimax and minimin rules would have given us different answers.*

3.2 Our generic racing approach

We are going to present a generic querying scheme based on racing ideas and then investigate the computational issue of such a scheme for the specific settings of binary SVM and decision trees.

Both the minimin and minimax approaches have the same goal: obtaining a unique model from partially specified data. Our objective in this proposal is different: we want to query those data that will increase the most the accuracy of a learnt model. To do so, we propose to start from a set Θ of potentially optimal models, and to identify in a racing scheme those data that will help the most to select the best model within Θ, hence are likely to be determinant in differentiating model quality. Much like querying-by-committee in classical active learning [57], the purpose of the race is here only to select the query to be made, as Θ is unlikely to contain the risk minimizing model. Once the queries have been made, a new model should be learned from the completed data set. How we quantify the usefulness of a query within the race is formalized in what follows.

Let us recall that we have been considering the setting that $\mathcal{X} = \mathcal{X}^1 \times \ldots \times \mathcal{X}^P$ is a Cartesian product of P real spaces $\mathbb{R}$, that a partial data $(\mathbf{X}_n, Y_n)$ can be expressed as $(\times_{p=1}^P X_n^p, Y_n)$, and furthermore that if $X_n^p \subseteq \mathbb{R}$ is a subset of the real line, then X_n^p is a closed interval.

A query on a partial data $(\times_{p=1}^P X_n^p, Y_n)$ consists in transforming one of its dimension X_n^p or Y_n into the true precise value x_n^p or y_n, thanks to an oracle (an expert, a precise measuring device). More precisely, q_n^p denotes the query made on X_n^p or Y_n, with $p = P + 1$ for Y_n. Given a model θ_l and a data $(\times_{p=1}^P X_n^p, Y_n)$, we are interested in knowing two things:

- whether the result of a query can have an effect on the the empirical risk bounds $[\underline{R}(\theta_l \,|\, \mathbf{D}), \overline{R}(\theta_l \,|\, \mathbf{D})]$, which will be the case only if the query can have an effect on the interval $[\underline{\ell}(Y_n, \theta_l(\mathbf{X}_n)), \overline{\ell}(Y_n, \theta_l(\mathbf{X}_n))]$. We will then speak about the single effect of a query, as we will consider a single model;

- whether the model θ_l can be preferred to θ_k after performing a query, in which case we have to assess whether the query can have an influence on the lower bound $\underline{R}(\theta_{k-l} \,|\, \mathbf{D})$ or not, since θ_l will be preferred to θ_k as soon as this bound becomes positive.

This can be formalized by two functions, $E_{q_n^p} : \Theta \to \{0,1\}$ and $J_{q_n^p} : \Theta \times \Theta \to \{0,1\}$ such that:

$$E_{q_n^p}(\theta_l) = \begin{cases} 1 & \text{if } \exists x_n^p \in X_n^p \text{ that reduces } [\underline{R}(\theta_l \,|\, \mathbf{D}), \overline{R}(\theta_l \,|\, \mathbf{D})] \\ 0 & \text{else} \end{cases} \tag{3.8}$$

and

$$J_{q_n^p}(\theta_k, \theta_l) = \begin{cases} 1 & \text{if } \exists x_n^p \in X_n^p \text{ that increases } \underline{R}(\theta_{k-l} \,|\, \mathbf{D}) \\ 0 & \text{else.} \end{cases} \tag{3.9}$$

When $p = P + 1$, X_n^p is to be replaced by Y_n. $E_{q_n^p}$ simply tells us whether or not the query can affect our evaluation of θ_l performances, while $J_{q_n^p}(\theta_k, \theta_l)$ informs us whether the query can help to differentiate θ_l and θ_k. If we denote by $k^* = \arg\min_{k \in \{1,\ldots,S\}} \underline{R}(\theta_k \,|\, \mathbf{D})$ the currently winning model (racing algorithms do focus on this model, trying to determine if it is really the winner of the race), the total effect of a query q_n^p is defined as

$$Value(q_n^p) = E_{q_n^p}(\theta_{k^*}) + \sum_{k \neq k^*} J_{q_n^p}(\theta_k, \theta_{k^*}). \tag{3.10}$$

This value or utility is then used to assess which data (label or feature) should be queried next. It should be noticed that scores (3.8) and (3.9) can be modified, for example to account for different loss functions. Unless there are other reasons to change it, our choice appears to be the most natural and simple.

Example 8. *In Figure 3.1, questions related to partial classes (points 4 and 5) and to partial features (points 1, 2 and 3) have respectively the same potential effect, so we can restrict our attention to q_4^3 (the class of point 4) and to q_3^2 (the second feature of point 3). For these two questions, we have*

- $E_{q_4^3}(\theta_1) = E_{q_4^3}(\theta_2) = 1$ *and* $J_{q_4^3}(\theta_1, \theta_2) = J_{q_4^3}(\theta_2, \theta_1) = 0$.
- $E_{q_3^2}(\theta_1) = 1$, $E_{q_3^2}(\theta_2) = 0$ *and* $J_{q_3^2}(\theta_1, \theta_2) = J_{q_3^2}(\theta_2, \theta_1) = 1$.

This example shows that while some questions may reduce our uncertainty about many model risks (q_4^3 reduce risk intervals for both models), they may be less useful than other questions to tell two models apart (q_3^2 can actually lead to declare θ_2 better than θ_1, while q_4^3 cannot).

The effect of a query being now formalized, we can propose a method inspired by racing algorithms. To create the initial set of racing models, a convenient method is to sample S times a precise data set $\{(\boldsymbol{x}_n, y_n) \in (\mathbf{X}_n, Y_n)\}_{n=1}^N$ and then to learn an optimal model for each such selection. Algorithm 2 summarises the general procedure applied to find the best query and to update the race. This algorithm simply searches the query that will have the biggest impact on the minimin model and its competitors, adopting the optimistic attitude of racing algorithms. Once a query has been made, the data set as well as the set of competitors are updated, so that only potentially optimal models remain. Note that in practice, such a sampling is close to methods used in query-by-committee approaches [57, 67], and makes no specific assumption about the process that has led to imprecision. Also, as in usual query-by-committee and racing approaches, we also assume that we work with models of the same nature and of comparable complexity.

Algorithm 2: One iteration of the racing algorithm to query data

Input: data (X_n, Y_n), set $\Theta^* := \{\theta_1, \dots, \theta_S\}$ of models
Output: updated data and set of models

$k^* = \arg\min_{k \in \{1,\dots,S\}} \underline{R}(\theta_k \,|\, \mathbf{D})$;
foreach *query* q_n^p **do**
  $Value(q_n^p) = E_{q_n^p}(\theta_{k^*}) + \sum_{k \neq k^*} J_{q_n^p}(\theta_k, \theta_{k^*})$;
$(n^*, p^*) = \arg\max_{(n,p)} Value(q_n^p)$;
Get value $x_{n^*}^{p^*}$ of $X_{n^*}^{p^*}$;
foreach $k, l \in \{1, \dots, S\} \times \{1, \dots, S\}$, $k \neq l$ **do**
  Compute $\underline{R}(\theta_{k-l} \,|\, \mathbf{D})$;
  if $\underline{R}(\theta_{k-l} \,|\, \mathbf{D}) > 0$ **then** remove θ_k from Θ^* ;

3.3 Application to SVM

In this Section, we illustrate our proposed setting and its potential interest with the popular SVM algorithm. We separate the two cases of interval-valued features from set-valued labels, for three reasons: (i) we can expect that imprecision in both aspects is less likely to happen in practice, (ii) this makes the exposure of the methods easier to follow, and (iii) considering both cases at once would quickly induce a too important imprecision in the results. We leave the combination of the two approaches to the reader, especially since binary SVM are here used as an illustration of our general approach.

3.3.1 Interval-valued features

In the binary SVM setting [10], the input space $\mathcal{X} = \mathbb{R}^P$ is the real space and the binary output space is $\mathcal{Y} = \{-1, 1\}$, where $-1, 1$ encode the two possible classes. The model $\theta_l = (\boldsymbol{w}_l, c_l)$ corresponds to the *maximum-margin* hyperplane $\boldsymbol{w}_l \boldsymbol{x} + c_l$ with $\boldsymbol{w}_l \in \mathbb{R}^P$ and $c_l \in \mathbb{R}$. For convenience sake, we will use $(\boldsymbol{w}_l, c_l)$ and θ_l interchangeably from now on. We will also focus in this section on the case of imprecise features and precise labels, and will denote y_n the label of training instances. We will also focus on the classical 0-1 loss function defined as follows for an instance $(\boldsymbol{x}_n, y_n)$:

$$\ell(y_n, \theta_l(\boldsymbol{x}_n)) = \begin{cases} 0 \text{ if } y_n \cdot \theta_l(\boldsymbol{x}_n) \geq 0 \\ 1 \text{ if } y_n \cdot \theta_l(\boldsymbol{x}_n) < 0, \end{cases} \quad := \ell_l(y_n, \boldsymbol{x}_n) \tag{3.11}$$

where $\theta_l(\boldsymbol{x}_n) = \boldsymbol{w}_l \boldsymbol{x}_n + c_l$, and $\ell_l(y_n, \boldsymbol{x}_n)$ is used as a short notation for $\ell(y_n, \theta_l(\boldsymbol{x}_n))$. Similarly, the extreme losses $\underline{\ell}(y_n, \theta_l(\mathbf{X}_n))$ and $\overline{\ell}(y_n, \theta_l(\mathbf{X}_n))$ are shortened to $\underline{\ell}_l(y_n, \mathbf{X}_n)$ and $\overline{\ell}_l(y_n, \mathbf{X}_n)$, respectively.

Instances inducing imprecision in empirical risk

Before entering into the details of how single risk bounds $[\underline{R}(\theta_l \,|\, \mathbf{D}), \overline{R}(\theta_l \,|\, \mathbf{D})]$ and pairwise risk bounds $\underline{R}(\theta_{k-l} \,|\, \mathbf{D})$ given by Equations (3.4)-(3.6), and query effects $E_{q_n^p}(\theta_l)$ and $J_{q_n^p}(\theta_k, \theta_l)$ given by Equations (3.8)-(3.9) can be estimated in practice, we will first investigate under which conditions an instance $(\mathbf{X}_n, y_n)$ induces imprecision in the empirical risk. Such instances are the only ones of interest here, since if $\underline{\ell}_l(y_n, \mathbf{X}_n) = \overline{\ell}_l(y_n, \mathbf{X}_n) = \ell_l(y_n, \mathbf{X}_n)$, then $E_{q_n^p}(\theta_l) = 0$ for all $p = 1, \dots, P$. Furthermore, if an instance $(\mathbf{X}_n, y_n)$ is precise w.r.t both θ_k and θ_l, then $J_{q_n^p}(\theta_k, \theta_l) = 0$ for

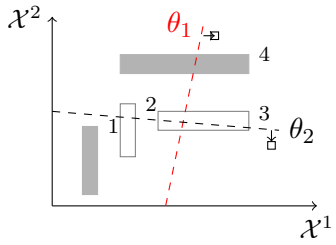


FIGURE 3.2: Illustration of interval-valued instances

all $p = 1, \ldots, P$. Thus, only instances which are imprecise w.r.t at least one model are interested when determining $J_{q_n^p}(\theta_k, \theta_l)$.

Definition 4. *Given a SVM model θ_l, an instance $(\mathbf{X}_n, y_n)$ is called an imprecise instance w.r.t. θ_l if and only if*

$$\exists \boldsymbol{x}_n^{'}, \boldsymbol{x}_n^{''} \in \mathbf{X}_n \ s.t \ \theta_l(\boldsymbol{x}_n^{'}) \geq 0 \ \textit{and} \ \theta_l(\boldsymbol{x}_n^{''}) < 0. \tag{3.12}$$

Instances that do not satisfy Definition 4 will be called precise instances (w.r.t. θ_l). Being precise means that the sign of $\theta_l(\boldsymbol{x}_n)$ is the same for all $\boldsymbol{x}_n \in \mathbf{X}_n$, which implies that the loss $\underline{\ell}_l(y_n, \mathbf{X}_n) = \overline{\ell}_l(y_n, \mathbf{X}_n)$ is precisely known. The next example illustrates the notion of (im)precise instances.

Example 9. *Figure 3.2 illustrates a situation with two models and where the two different classes are represented by grey ($y = +1$) and white ($y = -1$) colours. From the figure, we can say that $(\mathbf{X}_1, y_1)$ is precise w.r.t both θ_1 and θ_2, $(\mathbf{X}_2, y_2)$ is precise w.r.t θ_1 and imprecise w.r.t θ_2, $(\mathbf{X}_3, y_3)$ is imprecise w.r.t both θ_1 and θ_2 and $(\mathbf{X}_4, y_4)$ is imprecise w.r.t θ_1 and precise w.r.t θ_2.*

Determining whether an instance is imprecise w.r.t. θ_l is actually very easy in practice. Let us denote by

$$\underline{\theta}_l(\mathbf{X}_n) := \inf_{\boldsymbol{x}_n \in \mathbf{X}_n} \theta_l(\mathbf{x}_n) \text{ and } \overline{\theta}_l(\mathbf{X}_n) := \sup_{\boldsymbol{x}_n \in \mathbf{X}_n} \theta_l(\boldsymbol{x}_n) \tag{3.13}$$

the lower and upper bounds reached by model θ_l over the space $\mathbf{X}_n$. The following result characterizing imprecise instances, as well as when a hyperplane $\theta_l(\boldsymbol{x}_n) = 0$ intersects with a region $\mathbf{X}_n$, follows from the fact that the image of a compact set by a continuous function is also compact.

Proposition 8. *Given $\theta_l(\boldsymbol{x}_n) = \boldsymbol{w}_l \boldsymbol{x}_n + c_l$ and the set $\mathbf{X}_n$, then $(\mathbf{X}_n, y_n)$ is imprecise w.r.t. θ_l if and only if*

$$\underline{\theta}_l(\mathbf{X}_n) < 0 \ \textit{and} \ \overline{\theta}_l(\mathbf{X}_n) \geq 0. \tag{3.14}$$

Furthermore, we have that the hyperplane $\theta_l(\boldsymbol{x}_n) = 0$ intersects with the region $\mathbf{X}_n$ if and only if (3.14) holds. In other words, $\exists \boldsymbol{x}_n \in \mathbf{X}_n$ s.t. $\theta_l(\boldsymbol{x}_n) = 0$.

Proof. Since continuous functions preserve compactness and connectedness [33], then the image $f(\mathbf{X}) = Y$ of a compact and connected set $\mathbf{X}$ is compact and connected. Furthermore, a set on $\mathbb{R}^P$ is compact if and only if it is closed and bounded (Heine–Borel Theorem [74]), then $\mathbf{X}$ is a closed, bounded and connected set which is exactly a closed interval. Or in other words, we have that

$$\theta_l(\mathbf{X}_n) = \left[\underline{\theta}_l(\mathbf{X}_n), \overline{\theta}_l(\mathbf{X}_n)\right],$$

is an interval consisting of every possible values that can take $\theta_l(\boldsymbol{x}_n)$ for $\boldsymbol{x}_n \in \mathbf{X}_n$. That (3.14) is equivalent to (3.12) then immediately follows. Also, we have that $\exists \boldsymbol{x}_n \in \mathbf{X}_n$ s.t. $\theta_l(\boldsymbol{x}_n) = 0$ if and only if $0 \in \left[\underline{\theta}_l(\mathbf{X}_n), \overline{\theta}_l(\mathbf{X}_n)\right]$. □

This proposition means that to determine whether an instance $(\mathbf{X}_n, y_n)$ is imprecise, we only need to compute values $\underline{\theta}_l(\mathbf{X}_n)$ and $\overline{\theta}_l(\mathbf{X}_n)$, which can be easily done using Proposition 9.

Proposition 9. *Given $(\mathbf{X}_n, y_n)$ with $X_n^p = [a_n^p, b_n^p]$ and SVM model $(\boldsymbol{w}_l, c_l)$, we have*

$$\overline{\theta}_l(\mathbf{X}_n) = \sum_{w_l^p \geq 0} w_l^p b_n^p + \sum_{w_l^p < 0} w_l^p a_n^p + c_l$$
$$\underline{\theta}_l(\mathbf{X}_n) = \sum_{w_l^p \geq 0} w_l^p a_n^p + \sum_{w_l^p < 0} w_l^p b_n^p + c_l.$$

Proof. Since $\theta_l(\boldsymbol{x}_n)$ is a linear function, it is monotonic in each dimension, hence the extreme values are obtained at points $\boldsymbol{x}_n \in \times_{p=1}^{P} \{a_n^p, b_n^p\}$. Furthermore, $\theta_l(\boldsymbol{x}_n)$ decreases (increases) w.r.t $\boldsymbol{x}_n^p$ if $w_l^p < 0$ ($w_l^p > 0$). Hence, Proposition 9 holds. □

Again, it should be noted that only imprecise instances are of interest here, as these are the only instances that, once queried, can result in an increase of the lower empirical risk bounds. We will therefore focus on those in the next sections.

Example 10. *Consider the model θ_l on a 3-dimensional space given by $\boldsymbol{w}_l = (2, -1, 1)$ and the partial instance $\mathbf{X}_n = [1, 3] \times [2, 5] \times [1, 2]$. In this case, we have*

$$\underline{\theta}_l(\mathbf{X}_n) = 1 \times 2 + 5 \times -1 + 1 \times 1 = -2,$$

$$\overline{\theta}_l(\mathbf{X}_n) = 3 \times 2 + 2 \times -1 + 2 \times 1 = 6,$$

hence the instance $\mathbf{X}_n$ is imprecise with respect to θ_l.

Empirical risk bounds and single effect

We are now going to investigate the practical computation of $\underline{R}(\theta_l \,|\, \mathbf{D})$, $\overline{R}(\theta_l \,|\, \mathbf{D})$, as well as the value $E_{q_n^p}(\theta_l)$ of a query on a model θ_l. Equations (3.4) (resp. (3.5)) implies that the computation of $\underline{R}(\theta_l \,|\, \mathbf{D})$ (resp. $\overline{R}(\theta_l \,|\, \mathbf{D})$) can be done by first computing $\underline{\ell}_l(y_n, \mathbf{X}_n)$ (resp. $\overline{\ell}_l(y_n, \mathbf{X}_n)$) for $n = 1, \ldots, N$ and then summing the obtained values. This means that we can focus our attention on computing $\underline{\ell}_l(y_n, \boldsymbol{x}_n)$ and $\overline{\ell}_l(y_n, \boldsymbol{x}_n)$ for a single instance, as obtaining $\underline{R}(\theta_l \,|\, \mathbf{D})$, $\overline{R}(\theta_l \,|\, \mathbf{D})$ from them is straightforward. Note that we have $\underline{\ell}_l(y_n, \mathbf{X}_n) = 0$ and $\overline{\ell}_l(y_n, \mathbf{X}_n) = 1$ if and only if $\mathbf{X}_n$ is imprecise w.r.t. θ_l, a fact that can easily be checked using Proposition 8. The bounds of the loss interval for the model θ_l and datum $(\mathbf{X}_n, y_n)$ is

$$[\underline{\ell}_l(y_n, \mathbf{X}_n), \overline{\ell}_l(y_n, \mathbf{X}_n)] = \begin{cases} [0, 0] \text{ if } \min(y_n \cdot \overline{\theta}_l(\mathbf{X}_n), y_n \cdot \underline{\theta}_l(\mathbf{X}_n)) \geq 0 \\ [0, 1] \text{ if } \overline{\theta}_l(\mathbf{X}_n) \cdot \underline{\theta}_l(\mathbf{X}_n) < 0 \\ [1, 1] \text{ if } \max(y_n \cdot \overline{\theta}_l(\mathbf{X}_n), y_n \cdot \underline{\theta}_l(\mathbf{X}_n)) < 0 \end{cases} \tag{3.15}$$

Let us now focus on estimating the effect of a query. As with the loss bounds, the only situation where a query q_n^p can affect the empirical risk bounds, and hence the only situation where $E_{q_n^p}(\theta_l) = 1$, is when the interval $[\underline{\ell}_l(y_n, \mathbf{X}_n), \overline{\ell}_l(y_n, \mathbf{X}_n)]$ can be reduced by querying X_n^p. Therefore we can also focus on a single instance to evaluate it.

In the case of 0-1 loss, the only case where $E_{q_n^p}(\theta_l) = 1$ is the one where the imprecise loss $[\underline{\ell}_l(y_n, \mathbf{X}_n), \overline{\ell}_l(y_n, \boldsymbol{x}_n)]$ goes from $[0, 1]$ before the query to a precise value after it, or in other words if there is $x_n^p \in X_n^p$ such that $\mathbf{X}_n^{q_n^p} = \times_{p' \neq p} X_n^{p'} \times \{x_n^p\}$ is precise w.r.t. θ_l. According to Proposition 8, this means that either $\underline{\theta}_l(\mathbf{X}_n^{q_n^p})$ should become positive, or $\overline{\theta}_l(\mathbf{X}_n^{q_n^p})$ should become negative after a query q_n^p. The conditions to check whether this is possible are given in the next proposition.

Proposition 10. *Given $(\mathbf{X}_n, y_n)$ with $X_n^p = [a_n^p, b_n^p]$ and a model θ_l s.t. $\mathbf{X}_n$ is imprecise, then $E_{q_n^p}(\theta_l) = 1$ if and only if one of the following conditions holds*

$$\underline{\theta}_l(\mathbf{X}_n) \geq -|w_l^p|(b_n^p - a_n^p) \tag{3.16}$$

or

$$\overline{\theta}_l(\mathbf{X}_n) < |w_l^p|(b_n^p - a_n^p). \tag{3.17}$$

Proof. Let us concentrate on the first condition (the second one can be proved similarly). If we denote by $\underline{\theta}_l(\mathbf{X}_n^{q_n^p})$ the lower bound reached by θ_l on $\mathbf{X}_n^{q_n^p}$ (the set resulting from the query answer), then we have the following inequality

$$\underline{\theta}_l^{q_n^p}(\mathbf{X}_n^{q_n^p}) \leq \underline{\theta}_l(\mathbf{X}_n) + |w_l^p|(b_n^p - a_n^p)$$

giving us a tight upper bound for it. Indeed, if $w_l^p \geq 0$, then $\underline{\theta}_l$ is obtained for $x_n^p = a_n^p$ (by Proposition 9), and it can increase by at most $w_l^p(b_n^p - a_n^p)$ if the result of the query q_n^p is $x_n^p = b_n^p$ (the case $w_l^p \leq 0$ is similar). Since $\underline{\theta}_l(\mathbf{X}_n)$ is known to be negative (from Proposition 8 and the fact that $\mathbf{X}_n$ is imprecise), it can only become positive after a query q_n^p if $\underline{\theta}_l(\mathbf{X}_n) + |w_l^p|(b_n^p - a_n^p)$ is positive.

Finally, by investigating the change of $sign(w_l^p)$, we have:

B1: q_n^p can change the sign of $\underline{\theta}_l(\boldsymbol{x}_n)$ iff

$$\begin{cases} \underline{\theta}_l(\boldsymbol{x}_n) + w_l^p(b_n^p - a_n^p) \geq 0 \text{ if } w_l^p \geq 0, \\ \underline{\theta}_l(\boldsymbol{x}_n) - w_l^p(b_n^p - a_n^p) \geq 0 \text{ if } w_l^p < 0. \end{cases}$$

B2: q_n^p can change the sign of $\overline{\theta}_l(\boldsymbol{x}_n)$ iff

$$\begin{cases} \overline{\theta}_l(\boldsymbol{x}_n) - w_l^p(b_n^p - a_n^p) < 0 \text{ if } w_l^p \geq 0 \\ \overline{\theta}_l(\boldsymbol{x}_n) + w_l^p(b_n^p - a_n^p) < 0 \text{ if } w_l^p < 0. \end{cases}$$

□

$\underline{R}(\theta_l \,|\, \mathbf{D}), \overline{R}(\theta_l \,|\, \mathbf{D})$, needed in the line 1 of Algorithm 2 to identify the most promising model k^*, are computed easily by summing over all training instances the intervals $[\underline{\ell}_l(y_n, \mathbf{X}_n), \overline{\ell}_l(y_n, \mathbf{X}_n)]$ given by Equation (3.15), while Equations (3.16)-(3.17) give easy ways to estimate the values of $E_{q_n^p}(\theta_{k^*})$, needed in line 3 of Algorithm 2.

Example 11. *Let us consider again Example 10, and check whether querying the last ($p = 3$) or second dimension may induce some effect on the emprical risk bounds. Using Proposition 10, we have for q_n^3 that*

$$\underline{\theta}_l(\mathbf{X}_n) = -2 < -1 \times (2 - 1) \text{ and } \overline{\theta}_l(\mathbf{X}_n) = 6 > 1 \times (2 - 1),$$

hence $E_{q_n^3}(\theta_l) = 0$*, as none of the conditions are satisfied. We do have, on the contrary, that*

$$\underline{\theta}_l(\mathbf{X}_n) = -2 \geq -1 \times (5-2),$$

hence $E_{q_n^2}(\theta_l) = 1$*. Indeed, if* $x_n^2 = 2$ *(the query results in the lower bound), then the model becomes positive for any replacement of* $\mathbf{X}_n^{q_n^2} = [1,3] \times 2 \times [1,2]$.

Pairwise risk bounds and effect

Let us now focus on how to compute, for a pair of models θ_k and θ_l, whether a query q_n^p will have an effect on the value $\underline{R}(\theta_{k-l} \,|\, \mathbf{D})$. For this, we will have to compute $\underline{R}(\theta_{k-l} \,|\, \mathbf{D})$, which is a necessary step to estimate the indicator $J_{q_n^p}(\theta_k, \theta_l)$ of a possible effect of q_n^p. To do that, note that $\underline{R}(\theta_{k-l})$ can be rewritten as

$$\underline{R}(\theta_{k-l} \,|\, \mathbf{D}) = \inf_{\boldsymbol{x}_n \in \mathbf{X}_n, n=1,\ldots,N} (R(\theta_k \,|\, \mathbf{D}) - R(\theta_l) \,|\, \mathbf{D}) = \sum_{n=1}^{N} \underline{\ell}_{k-l}(y_n, \mathbf{X}_n) \tag{3.18}$$

with

$$textwith \underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = \inf_{\boldsymbol{x}_n \in \mathbf{X}_n} \Big(\ell_k(y_n, \boldsymbol{x}_n) - \ell_l(y_n, \boldsymbol{x}_n) \Big), \tag{3.19}$$

meaning that computing $\underline{R}(\theta_{k-l} \,|\, \mathbf{D})$ can be done by summing up $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$ over all $\mathbf{X}_n$, similarly to $\underline{R}(\theta_l \,|\, \mathbf{D})$ and $\overline{R}(\theta_l \,|\, \mathbf{D})$. Also, $J_{q_n^p}(\theta_k, \theta_l) = 1$ if and only if q_n^p can increase $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$. We can therefore focus on the computation of $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$ and its possible changes.

First note that if $\mathbf{X}_n$ is precise w.r.t. both θ_k and θ_l, then $\ell_k(y_n, \mathbf{X}_n) - \ell_l(y_n, \mathbf{X}_n)$ is a well-defined value, as each loss is precise, and in this case $J_{q_n^p}(\theta_k, \theta_l) = 0$. Therefore, the only cases of interest are those where $\mathbf{X}_n$ is imprecise w.r.t. at least one model. We will first treat the case where it is imprecise for only one, and then we will proceed to the more complex one where it is imprecise w.r.t. both. Note that imprecision with respect to each model can be easily established using Proposition 8.

Case 1*: Imprecision with respect to one model*

Let us consider the case where $\mathbf{X}_n$ is imprecise w.r.t. either θ_k or θ_l. In each of these two cases, the loss induced by $(\mathbf{X}_n, y_n)$ on the model for which it is precise is fixed. Hence, to estimate the lower loss $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$, as well as the effect of a possible query q_n^p, we only have to look at the model for which $(\mathbf{X}_n, y_n)$ is imprecise. The next proposition establishes the lower bound $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$, necessary to compute $\underline{R}(\theta_{k-l})$.

Proposition 11. *Given* $(\mathbf{X}_n, y_n)$ *with* $X_n^p = [a_n^p, b_n^p]$ *and two models* θ_k *and* θ_l *s.t* $(\mathbf{X}_n, y_n)$ *is imprecise w.r.t. one and only one model, then we have*

$$\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = \ell_k(y_n, \mathbf{X}_n) - 1 \qquad \text{if } \mathbf{X}_n \text{ imprecise w.r.t. } \theta_l \tag{3.20}$$

$$\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = 0 - \ell_l(y_n, \mathbf{X}_n) \qquad \text{if } \mathbf{X}_n \text{ imprecise w.r.t. } \theta_k. \tag{3.21}$$

Proof. We will only prove Equation (3.20), the proof for Equation (3.21) being similar. First note that if $\mathbf{X}_n$ is precise with respect to θ_k, then $\ell_k(y_n, \mathbf{X}_n)$ is precise. Second, the value of $\ell_l(y_n, \mathbf{X}_n) \in \{0, 1\}$, since $\mathbf{X}_n$ is imprecise with respect to θ_l, hence the lower bound is obtained for $\boldsymbol{x}_n \in \mathbf{X}_n$ such that $\ell_l(y_n, \boldsymbol{x}_n) = 1$. □

We kept the 0 in Equation (3.21) to make clear that we take the lower bound of the loss w.r.t. θ_k, and the precise value of $\ell_l(y_n, \mathbf{X}_n)$. Let us now study under

which conditions a query q_n^p can increase $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$, hence under which conditions $J_{q_n^p}(\theta_k, \theta_l) = 1$. The two next propositions respectively address the case of imprecision w.r.t. θ_k and θ_l. Given a possible query q_n^p on $\mathbf{X}_n$, the only possible way to increase $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$ is for the updated $\mathbf{X}_n^{q_n^p}$ to become precise w.r.t. the model for which $\mathbf{X}_n$ was imprecise, and moreover to be so that $\ell_l(y_n, \mathbf{X}_n^{q_n^p}) = 0$ ($\ell_k(y_n, \mathbf{X}_n^{q_n^p}) = 1$) if $\mathbf{X}_n$ is imprecise w.r.t. θ_l (θ_k).

Proposition 12. *Given $(\mathbf{X}_n, y_n)$ with $X_n^p = [a_n^p, b_n^p]$ and two models θ_k and θ_l s.t. $(\mathbf{X}_n, y_n)$ is imprecise w.r.t. θ_l, the question q_n^p is such that $J_{q_n^p}(\theta_k, \theta_l) = 1$ if and only if one of the two following conditions holds*

$$y_n = 1 \text{ and } \underline{\theta}_l(\mathbf{X}_n) \geq -|w_l^p|(b_n^p - a_n^p) \tag{3.22}$$

$$\text{or}$$

$$y_n = -1 \text{ and } \overline{\theta}_l(\mathbf{X}_n) < |w_l^p|(b_n^p - a_n^p). \tag{3.23}$$

Proof. First note that if $\mathbf{X}_n$ is imprecise w.r.t. θ_l, then the only case where $\underline{\ell}_{k-l}(\mathbf{X}_n)$ increases is when the updated instance $\mathbf{X}_n^{q_n^p}$ is precise w.r.t. θ_l after the query q_n^p is performed and the precise loss becomes $\ell_l(y_n, \mathbf{X}_n^{q_n^p}) = 0$.

Let us consider the case $y_n = 1$ (the case $y_n = 0$ is similar). To have $\ell_l(y_n, \mathbf{X}_n^{q_n^p}) = 0$, we must have $\underline{\theta}_l(\mathbf{X}_n^{q_n^p}) \geq 0$. Using the same argument as in Proposition 10, we easily get the result. □

Proposition 13. *Given $(\mathbf{X}_n, y_n)$ with $X_n^p = [a_n^p, b_n^p]$ and two models θ_k and θ_l s.t. $(\mathbf{X}_n, y_n)$ is imprecise w.r.t. θ_k, the query q_n^p is such that $J_{q_n^p}(\theta_k, \theta_l) = 1$ if and only if one of the two following condition holds*

$$y_n = 1 \text{ and } \overline{\theta}_k(\mathbf{X}_n) < |w_k^p|(b_n^p - a_n^p) \tag{3.24}$$

$$\text{or}$$

$$y_n = -1 \text{ and } \underline{\theta}_k(\mathbf{X}_n) \geq -|w_k^p|(b_n^p - a_n^p). \tag{3.25}$$

The proof is analogous to the one of Proposition 12.

In summary, if $\mathbf{X}_n$ is imprecise w.r.t. only one model, estimating $J_{q_n^p}(\theta_k, \theta_l)$ comes down to identify whether the $\mathbf{X}_n$ can become precise with respect to such a model, in such a way that the lower bound is possibly increased. Propositions 12 and 13 show that this can be checked easily using our previous results investigated in the Proposition 10 concerning the empirical risk. Actually, in this case, the problem essentially boils down to the problem of determining the empirical risk bounds and single effect.

Case 2: Imprecision with respect to both models

Given $\mathbf{X}_n$ and two models θ_k, θ_l, we define :

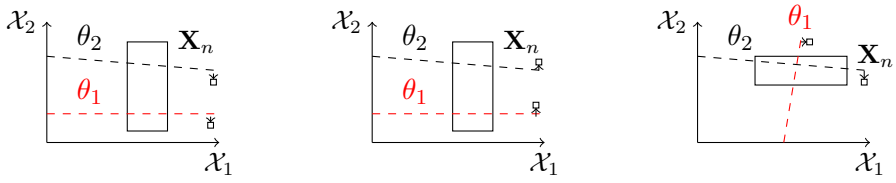
$$\theta_{k-l}(\mathbf{X_n}) = \theta_k(\mathbf{X}_n) - \theta_l(\mathbf{X}_n). \tag{3.26}$$

We thus have:

$$\theta_{k-l}(\mathbf{X_n}) > 0 \text{ if } \theta_k(\boldsymbol{x}_n) - \theta_l(\boldsymbol{x}_n) > 0 \quad \forall \boldsymbol{x}_n \in \mathbf{X}_n \tag{3.27}$$

$$\theta_{k-l}(\mathbf{X_n}) < 0 \text{ if } \theta_k(\boldsymbol{x}_n) - \theta_l(\boldsymbol{x}_n) < 0 \quad \forall \boldsymbol{x}_n \in \mathbf{X}_n. \tag{3.28}$$

In the other cases, this means that there are $\boldsymbol{x}_n^{'}, \boldsymbol{x}_n^{''} \in \mathbf{X}_n$ for which the model difference have different signs. The reason for introducing such differences is that, if



(a) $\theta_{1-2}(\mathbf{X_n}) > 0$ (b) $m_{1-2}(\mathbf{X_n}) < 0$ (c) Non-constant sign

FIGURE 3.3: Illustrations for the different possible cases corresponding to the pairwise difference

$\theta_{k-l}(\mathbf{X_n}) > 0$ or $\theta_{k-l}(\mathbf{X}_n) < 0$, then not all combinations in $\{0,1\}^2$ are possible for the pair $(\ell_k(y_n, \boldsymbol{x}_n), \ell_l(y_n, \boldsymbol{x}_n))$, while they are in the other case. These various situations are depicted in Figure 3.3, where the white class is again the negative one ($y_n = -1$).

Since $\theta_k(\boldsymbol{x}_n) - \theta_l(\boldsymbol{x}_n)$ is also of linear form (with weights $w_k^p - w_l^p$), we can easily determine whether the sign of $\theta_{k-l}(\mathbf{X}_n)$ is constant: it is sufficient to compute the interval

$$\left[\inf_{\boldsymbol{x}_n \in \mathbf{X}_n} (\theta_k(\boldsymbol{x}_n) - \theta_l(\boldsymbol{x}_n)), \sup_{\boldsymbol{x}_n \in \mathbf{X}_n} (\theta_k(\boldsymbol{x}_n) - \theta_l(\boldsymbol{x}_n))\right]$$

that can be computed similarly to $[\underline{\theta}_l(\mathbf{X}_n), \overline{\theta}_l(\mathbf{X}_n)]$ in the Proposition 9. If zero is not within this interval, then $\theta_{k-l}(\mathbf{X}_n) > 0$ if the lower bound is positive, otherwise $\theta_{k-l}(\mathbf{X}_n) < 0$ if the upper bound is negative. The next proposition indicates how to easily compute the lower bound $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$ for the different possible situations.

Proposition 14. *Given $(\mathbf{X}_n, y_n)$ with $X_n^p = [a_n^p, b_n^p]$ and two models θ_k, θ_l s.t. $(\mathbf{X}_n, y_n)$ is imprecise w.r.t. both models, then the minimal difference value is*

$$\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = \begin{cases} \min(0, -y_n) & \text{if } \theta_{k-l}(\mathbf{X}_n) > 0 \\ \min(0, y_n) & \text{if } \theta_{k-l}(\mathbf{X}_n) < 0 \\ -1 & \text{if } \theta_{k-l}(\mathbf{X}_n) \text{ can take both signs} \end{cases} \tag{3.29}$$

Proof. First note that when neither $\theta_{k-l}(\mathbf{X}_n) > 0$ nor $\theta_{k-l}(\mathbf{X}_n) < 0$ hold, then there are values $\boldsymbol{x}_n$ for which $\theta_k(\boldsymbol{x}_n)$ and $\theta_l(\boldsymbol{x}_n)$ are either positive and negative, or negative and positive, or of the same sign. Hence there is always a value $\boldsymbol{x}_n$ such that $\ell_k(y_n, \boldsymbol{x}_n) = 0$ and $\ell_l(y_n, \boldsymbol{x}_n) = 1$.

Let us then deal with the situation where $\theta_{k-l}(\mathbf{X}_n) > 0$ (the case $\theta_{k-l}(\mathbf{X}_n) < 0$ can be treated similarly). In this case, there are values $\boldsymbol{x}_n \in \mathbf{X}_n$ such that $\theta_k(\boldsymbol{x}_n)$ and $\theta_l(\boldsymbol{x}_n)$ have the same sign (0/1 loss difference is then null), or $\theta_k(\boldsymbol{x}_n)$ is positive and $\theta_l(\boldsymbol{x}_n)$ negative, but no values for which $\theta_k(\boldsymbol{x}_n)$ is negative and $\theta_l(\boldsymbol{x}_n)$ positive. When $\theta_k(\boldsymbol{x}_n)$ is positive and $\theta_l(\boldsymbol{x}_n)$ negative, the loss difference is -1 if $y_n = +1$, and 1 if $y_n = -1$. □

The next question is to know under which conditions a query q_n^p can increase $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$ (or equivalently $\underline{R}(\theta_{k-l})$), or in other words to determine a pair (n, p) s.t $J_{q_n^p}(\theta_k, \theta_l) = 1$. Proposition 14 tells us that $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$ can be either 0 or -1 if $\theta_{k-l}(\mathbf{X}_n) > 0$ or $\theta_{k-l}(\mathbf{X}_n) < 0$, and is always -1 if $\theta_{k-l}(\mathbf{X}_n)$ can take both signs. The next proposition establishes conditions under which $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$ can increase.

Proposition 15. *Given* $(\mathbf{X}_n, y_n)$ *with* $X_n^p = [a_n^p, b_n^p]$ *and two models* θ_k *and* θ_l *s.t* $(\mathbf{X}_n, y_n)$ *is imprecise w.r.t both of the given models, then* $J_{q_n^p}(\theta_k, \theta_l) = 1$ *if the following conditions hold*

if $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = -1$ ***and*** $y_n = 1$***:***

$$\overline{\theta}_k(\mathbf{X}_n) < |w_l^p|(b_n^p - a_n^p) \text{ or } \underline{\theta}_l(\mathbf{X}_n) \geq -|w_l^p|(b_n^p - a_n^p) \tag{3.30}$$

if $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = -1$ ***and*** $y_n = -1$***:***

$$\underline{\theta}_k(\mathbf{X}_n) \geq -|w_k^p|(b_n^p - a_n^p) \text{ or } \overline{\theta}_l(\mathbf{X}_n) < |w_l^p|(b_n^p - a_n^p). \tag{3.31}$$

if $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = 0$ ***and*** $\theta_{k-l}(\mathbf{X}_n) < 0$***:***

$$\overline{\theta}_k(\mathbf{X}_n) < |w_l^p|(b_n^p - a_n^p) \text{ and } \underline{\theta}_l(\mathbf{X}_n) \geq -|w_l^p|(b_n^p - a_n^p) \tag{3.32}$$

if $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = 0$ ***and*** $\theta_{k-l}(\mathbf{X}_n) > 0$***:***

$$\underline{\theta}_k(\mathbf{X}_n) \geq -|w_k^p|(b_n^p - a_n^p) \text{ and } \overline{m}_l(\mathbf{X}_n) < |w_l^p|(b_n^p - a_n^p). \tag{3.33}$$

Proof. Let us first investigate the case where $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = -1$ and $y_n = 1$ (the case $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = -1$ and $y_n = -1$ is similar). In this case, $J_{q_n^p}(\theta_k, \theta_l) = 1$ if and only if q_n^p can either increase $\underline{\ell}_k(y_n, \mathbf{X}_n) = 0$ or decrease $\overline{\ell}_l(y_n, \mathbf{X}_n) = 1$, that is become precise for at least one of them, with $\underline{\ell}_k(y_n, \mathbf{X}_n^{q_n^p}) = 1$ or $\overline{\ell}_l(y_n, \mathbf{X}_n^{q_n^p}) = 0$. The conditions are then obtained by following arguments similar to those of Proposition 10.

The second case $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = 0$ only happens when either $\theta_{k-l}(\mathbf{X}_n) < 0$ or $\theta_{k-l}(\mathbf{X}_n) > 0$, and we will treat the first case. According to Proposition 14, this means that $y_n = -1$. Also, since according to Proposition 11 the value 0 is an upper bound of $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$ when $\mathbf{X}_n$ is imprecise with either θ_k or θ_l, to go from $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = 0$ to $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n^{q_n^p}) = 1$, we need a value $x_n^p \in X_n^p$ such that $\theta_k(\mathbf{X}_n^{q_n^p}) < 0$ and $\theta_l(\mathbf{X}_n^{q_n^p}) > 0$, as $y_n = -1$. Again, we can get the conditions to have such a value by deriving arguments similar to those of Proposition 10. □

For instance, in Figure 3.3(a) and 3.3(b), $J_{q_n^1}(\theta_2, \theta_1) = 0$ and $J_{q_n^2}(\theta_2, \theta_1) = 1$ for both cases. The whole procedure is summed up in Algorithm 2. Algorithm 3 summarizes how to determine the query effect q_n^p, which can be considered as the main computational difficulty when performing the querying step (line $2-3$ in Algorithm 2). Determining the set of undominated models (line $6-8$ in Algorithm 2) is summarized in Algorithm 4.

Let us now study the complexity of the whole approach. Lines 2 and 4 of Algorithm 3 are in $\mathcal{O}(P)$, since they correspond to linear operations. Iterations from $5-10$ are in $\mathcal{O}(S \times P)$, since we must check all undominated models once. Iterations from $13-15$ are also in $\mathcal{O}(S \times P)$, for the same reason. Thus, one run of Algorithm 3 is in $\mathcal{O}(S \times P)$. If we have I partial features in the data, then loop $2-3$ of Algorithm 2 takes $\mathcal{O}(I \times S \times P)$ in the case of SVM, so it remains linear in each of the parameter. Algorithm 4 corresponds to lines 6-8 of Algorithm 2, and computing $\underline{R}(\theta_{k-l})$ can be done in $\mathcal{O}(N \times P)$ since we must compute $\underline{\ell}$ for each data point. Finally, since this must be done for every pair of models in the worst case, performing Algorithm 4 is in $\mathcal{O}(S^2 \times N \times P)$, which is quadratic in S and linear in the other parameters. This can be approximated by only comparing intervals $[\underline{R}(\theta_k), \overline{R}(\theta_k)]$ of every models, that would bring down the complexity to $\mathcal{O}(S \times N \times P)$, but would provide a super-set of the set of undominated models.

Algorithm 3: Determining the query effect $Value(q_n^p)$

Input: partial data $(\mathbf{X}_n, y_n)$, set $\Theta = \{\theta_1, \ldots, \theta_S\}$ of models, the best potential model θ_{k^*}

Output: the query effect $Value(q_n^p)$

initialize $E_{q_n^p}(\theta_{k^*}) = 0$, $J_{q_n^p}(\theta_k, \theta_{k^*}) = 0$, $Value(q_n^p) = 0$, $\forall k \neq k^*$;
check whether $(\mathbf{X}_n, y_n)$ is imprecise w.r.t θ_{k^*} using Prop. 8 and 9;
if $(\mathbf{X}_n, y_n)$ *is imprecise w.r.t* θ_{k^*} **then**
    compute $E_{q_n^p}(\theta_{k^*})$ using Prop. 10 ;
    foreach $k \neq k^*$ **do**
        if $(\mathbf{X}_n, y_n)$ *is imprecise w.r.t* θ_k **then**
            use Prop. 14 to get $\underline{\ell}_{k-k^*}(y_n, \mathbf{X}_n)$;
            use Prop. 15 to get $J_{q_n^p}(\theta_k, \theta_{k^*})$;
        else
            use Prop. 12 to get $J_{q_n^p}(\theta_k, \theta_{k^*})$;
    compute $Value(q_n^p)$ using Definition 3.10;
else
    foreach $k \neq k^*$ **do**
        if $(\mathbf{X}_n, y_n)$ *is imprecise w.r.t* θ_k **then**
            use Prop. 13 to get $J_{q_n^p}(\theta_k, \theta_{k^*})$;
    compute $Value(q_n^p)$ using Definition 3.10;

Algorithm 4: Determining the undominated set

Input: data $\mathbf{D} = \{(\mathbf{X}_n, y_n)\}_{n=1}^N$, set $\Theta = \{\theta_1, \ldots, \theta_S\}$ of models

Output: the set of undominated model Θ^*

foreach $k, l \in \{1, \ldots, S\} \times \{1, \ldots, S\}$, $k \neq l$ **do**
    $\underline{R}(\theta_{k-l} \,|\, \mathbf{D}) = 0$;
    foreach *data* $(\mathbf{X}_n, y_n)$ **do**
        if $(\mathbf{X}_n, y_n)$ *is imprecise w.r.t both* θ_k *and* θ_l **then**
            use Prop. 14 to get $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$;
        else if $(\mathbf{X}_n, y_n)$ *is imprecise w.r.t only one of* θ_k *and* θ_l **then**
            use Prop. 11 to get $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$;
        else
            compute $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = \ell_k(y_n, \mathbf{X}_n) - \ell_l(y_n, \mathbf{X}_n)$ using (3.15);
        $\underline{R}(\theta_{k-l} \,|\, \mathbf{D}) = \underline{R}(\theta_{k-l} \,|\, \mathbf{D}) + \underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$;
    if $\underline{R}(\theta_{k-l} \,|\, \mathbf{D}) > 0$ **then**
        remove θ_k from $\{\theta_1, \ldots, \theta_S\}$;

3.3.2 Set-valued labels

This section investigates the computations of racing algorithms to query set-valued labels when using binary SVM with precise features and when labels are partially given. Let us first note that, in the binary case, the problem of querying partial label data is identical to classical active learning as label data is either precise or fully partial (completely missing). One suitable technique in such a case is query-by-committee [83]. However, the strategies of query-by-committee technique and our racing technique are different. The previous one focus on missing labels that are the least consensual or the most ambiguous among a given set of models, while racing algorithms focus on labels having the most effect on reducing the uncertainty about the best potential model performance, as well as its difference to other models. From such intuitions, we could hope that, in practice, query-by-committee provide a quick reduction on the size of the set of undominated models while racing algorithms give faster convergence on determining the best potential model. In any case, it is worth exploring whether the two techniques perform similarly or if they show significant differences.

Before investigating the detailed computations of racing algorithms, let us recall that we focus here on binary SVM with 0/1 loss function (3.11). Also, as the output is partially given and inputs are precise, from now on and to facilitate exposure, we will adopt the notation $(\boldsymbol{x}_n, Y_n)$ where $Y_n \subseteq \{-1, 1\} = \mathcal{Y}$ and $\boldsymbol{x}_n \in \mathcal{X}$. Let us first note that, in case of precise label (i.e, $Y_n = y_n$), it is clear that the corresponding loss score is precisely given as in (3.34) and such an instance cannot be queried.

$$\ell_l(Y_n, \boldsymbol{x}_n) = \underline{\ell}_l(Y_n, \boldsymbol{x}_n) = \overline{\ell}_l(Y_n, \boldsymbol{x}_n) = \begin{cases} 0 \text{ if } Y_n \theta_l(\boldsymbol{x}_n) \geq 0, \\ 1 \text{ otherwise.} \end{cases} \tag{3.34}$$

We are now going to determine the imprecise loss function,

$$[\underline{\ell}_l(Y_n, \boldsymbol{x}_n), \overline{\ell}_l(Y_n, \boldsymbol{x}_n)]$$

and investigate under which conditions an imprecise label can have an effect on the risk bounds.

Proposition 16. *Given a model θ_l and an instance $(\boldsymbol{x}_n, Y_n)$, if $Y_n = \{-1, 1\}$, then the following results hold*

A1 $[\underline{\ell}_l(Y_n, \boldsymbol{x}_n), \overline{\ell}_l(Y_n, \boldsymbol{x}_n)] = [0, 1]$

A2 $E_{q_n}(\theta_l) = 1$.

Proof. It is clear that, in the binary case, if $Y_n = \{-1, 1\}$, whatever the prediction of the given model is (either 1 or -1), there always exist element y_n and $y_n^{'}$ in Y_n s.t

$$\ell_l(y_n, \boldsymbol{x}_n) = 0 \text{ and } \ell_l(y_n^{'}, \boldsymbol{x}_n) = 1,$$

or in other words, $[\underline{\ell}_l(Y_n, \boldsymbol{x}_n), \overline{\ell}_l(Y_n, \boldsymbol{x}_n)] = [0, 1]$. Furthermore, querying Y_n always help to modify $[\underline{\ell}_l(Y_n, \boldsymbol{x}_n), \overline{\ell}_l(Y_n, \boldsymbol{x}_n)]$ into single value (either to 0 or 1). Or, in other words, **A2** holds. □

Proposition 16 simply points out that all partial labels give the same (interval-valued) losses and have an effect on modifying the corresponding losses. In the next Proposition, we show that if the predictions of two given models for a partially labelled instance are different, then the corresponding lower pairwise difference is -1 and the effect of querying such labels is 1. Otherwise, both values are 0.

Proposition 17. *Given two models θ_k and θ_l and an imprecise instance $(\boldsymbol{x}_n, Y_n)$ ($Y_n = \{-1, 1\}$) then the following properties hold*

B1 *if $\theta_k(\boldsymbol{x}_n) = \theta_l(\boldsymbol{x}_n)$ then*

$$\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n) = 0 \text{ and } J_{q_n}(\theta_k, \theta_l) = 0.$$

B2 *if $\theta_k(\boldsymbol{x}_n) \neq \theta_l(\boldsymbol{x}_n)$ then*

$$\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n) = -1 \text{ and } J_{q_n}(\theta_k, \theta_l) = 1.$$

Proof. **B1** follows from the fact that if $\theta_k(\boldsymbol{x}_n) = \theta_l(\boldsymbol{x}_n)$, then $\ell_{k-l}(y_n, \boldsymbol{x}_n) = 0$ for all $y_n \in Y_n$. Furthermore, for any $y_n^{q_n} \in Y_n$ to be returned after performing q_n, we always have $\ell_{k-l}(y_n^{q_n}, \boldsymbol{x}_n) = 0$, or in other words $J_{q_n}(\theta_k, \theta_l) = 0$.

We are now going to give the proof for **B2**. Let us first notice that when $\theta_k(\boldsymbol{x}_n) \neq \theta_l(\boldsymbol{x}_n)$, there always exists $y_n \in Y_n$ (i.e $y_n = \theta_l(\boldsymbol{x}_n)$) s.t $\ell_{k-l}(y_n) = -1$. Then it is clear that $\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n) = -1$. Furthermore, if $y_n^{q_n} = \theta_l(\boldsymbol{x}_n)$ is the given label after performing q_n, then the pairwise difference $\ell_{k-l}^{q_n}(y_n^{q_n}, \boldsymbol{x}_n) = 1$. In other words, we have $J_{q_n}(\theta_k, \theta_l) = 1$. □

Propositions 16 and 17 provide an interesting property of $Value(q_n)$. In fact, for any given partial label Y_n, the corresponding total effect ($Value(q_n)$) is exactly $1 + u_i$ where u_i is the number of models in the undominated set that give predictions against the best potential model (θ_{k^*}). This means that while the query-by-committee approach does consider the consensus between all models for each instance, the racing algorithms are based on the consensus of each model w.r.t. to the best potential model, for all instances. Again, we can see similarities and differences between the two approaches, and comparing them makes sense.

The whole procedure is again summed up in Algorithm 2. Similarly to the case of interval-valued features, we summarize how to determine the query effect q_n (line $2-3$ in Algorithm 2) and the set of undominated models (line $6-8$ in Algorithm 2) in Algorithm 5 and 6, respectively. The complexity analysis is similar to the one of interval-valued features.

Algorithm 5: Determining the query effect $Value(q_n)$

Input: partial data $(\boldsymbol{x}_n, Y_n)$ with $Y_n = \{-1, 1\}$, set $\Theta = \{\theta_1, \ldots, \theta_S\}$ of models, the best potential model θ_{k^*}

Output: the query effect $Value(q_n)$

1 initialize $E_{q_n}(\theta_{k^*}) = 1$;
2 **foreach** $k \neq k^*$ **do**
3 use Prop. 17 to get $J_{q_n}(\theta_k, \theta_{k^*})$;
4 compute $Value(q_n)$ using Definition 3.10;

3.3.3 Experimental evaluation

We run experiments on a "contaminated" version of 7 standard benchmark (binary classes) data sets that are described in Table 3.1. The next two paragraphs present the details of the experiments and the results obtained in the two cases of interval-valued features and set-valued labels.

Algorithm 6: Determining the undominated set

Input: data $\mathbf{D} = \{(\boldsymbol{x}_n, Y_n)\}_{n=1}^N$, set $\Theta = \{\theta_1, \ldots, \theta_S\}$ of models
Output: the set of undominated model $\mathcal{M}^*$

foreach $k, l \in \{1, \ldots, S\} \times \{1, \ldots, S\}$, $k \neq l$ **do**
  $\underline{R}(\theta_{k-l} \,|\, \mathbf{D}) = 0$;
  foreach *data* $(\boldsymbol{x}_n, Y_n)$ **do**
    if $(\boldsymbol{x}_n, Y_n)$ *is imprecise* **then**
      use Prop. 17 to get $\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n)$;
    else
      compute $\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n) = \ell_k(Y_n, \boldsymbol{x}_n) - \ell_l(Y_n, \boldsymbol{x}_n)$ using (3.34)
    $\underline{R}(\theta_{k-l} \,|\, \mathbf{D}) = \underline{R}(\theta_{k-l} \,|\, \mathbf{D}) + \underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n)$;
  if $\underline{R}(\theta_{k-l} \,|\, \mathbf{D}) > 0$ **then** remove θ_k from $\{\theta_1, \ldots, \theta_S\}$;

Name	# instances	# features
parkinsons	197	22
vertebral-column	310	6
ionosphere	351	34
climate-model	540	18
breast-cancer	569	30
blood-transfusion	784	4
banknote-authentication	1372	4

TABLE 3.1: Data set used in the experiments

Interval-valued features case

Given a data set, we randomly chose a training set $\mathbf{D}$ consisting of 10% of instances and the rest (90%) as a test set $\mathbf{T}$. For each training instance $\boldsymbol{x}_n \in \mathbf{D}$, and each dimension $p = 1, ..., P$, a biased coin is flipped in order to decide whether or not x_n^p will be contaminated; the probability of contamination is ϵ (ϵ is fixed to 0.4 in all the experiments). Note that the probability that an instance has at least one contaminated feature is equal to $1 - 0.6^P$ (the complement of having no features contaminated), which is quite high: 0.87 when $P = 4$, our lowest number of features in any data set. In case x_n^p is contaminated, a width η_n^p will be generated from a uniform distribution. Then, the generated interval valued data is $X_n^p = [x_n^p + \eta_n^p(\underline{D}^p - x_n^p), x_n^p + \eta_n^p(\overline{D}^p - x_n^p)]$ where $\underline{D}^p = \min_n(x_n^p)$ and $\overline{D}^p = \max_n(x_n^p)$.

Example 12. *Assume that the initial precise observed value is $x = 1$, that the domain is $[\underline{D}, \overline{D}] = [0, 10]$, and that we have randomly picked $\eta = 0.5$. In this case, the resulting interval-valued data is $X = [0.5, 5.5]$.*

The set of undominated models is generated as follows: we randomly choose 100 precise replacements from the interval-valued training data. From each replacement, one linear SVM model is trained. The set of such 100 models is considered as the initial set Θ of undominated models.

After each query, the efficiency of the querying scheme is assessed based on the two following criteria:

- the proportion on the test set $\mathbf{T} = \{(\boldsymbol{x}_t, y_t)\}_{t=1}^T$ of identical predictions between the current best potential model θ_{k^*} and a reference model θ_{ref}. This similarity

is computed as

$$\frac{|\{(\boldsymbol{x}_t, y_t)|\theta_{ref}(\boldsymbol{x}_t) = \theta_{k^*}(\boldsymbol{x}_t)\}|}{T}.$$

This similarity is 1 if the two models make identical predictions on the test set (hence have the same performances), and 0 if they systematically disagree. The reference model is chosen to be the one in the initial undominated set that has the best accuracy on the fully precise training set. It is thus the model towards which any querying strategy, and the race in particular, should converge;

- the size of the undominated set.

To make comparisons about the convergence of the two criteria, two baseline algorithms are also used to query interval-valued features:

- a random querying strategy where, each time, an interval feature to be queried is chosen randomly;
- the most partial querying strategy i.e, each time, the feature with the largest imprecision (i.e., the largest sampled value η_n^p) is queried.

Because the training set is randomly chosen and contaminated, the results may be affected by random components. Then, for each data set, we repeat the above procedure 10 times and compute the average results.

Set-valued labels case

Experiments for the case of set-valued labels is performed in a similar way. Firstly, we randomly chose a training set $\mathbf{D}$ consisting of 20% of instances and the rest (of 80%) as a test set $\mathbf{T}$. Then, each label y_n in the training set $\mathbf{D}$ will be contaminated with probability ϵ (ϵ is fixed to 0.8 in all the experiments). Since the label is binary, if a label is contaminated, it becomes completely missing.

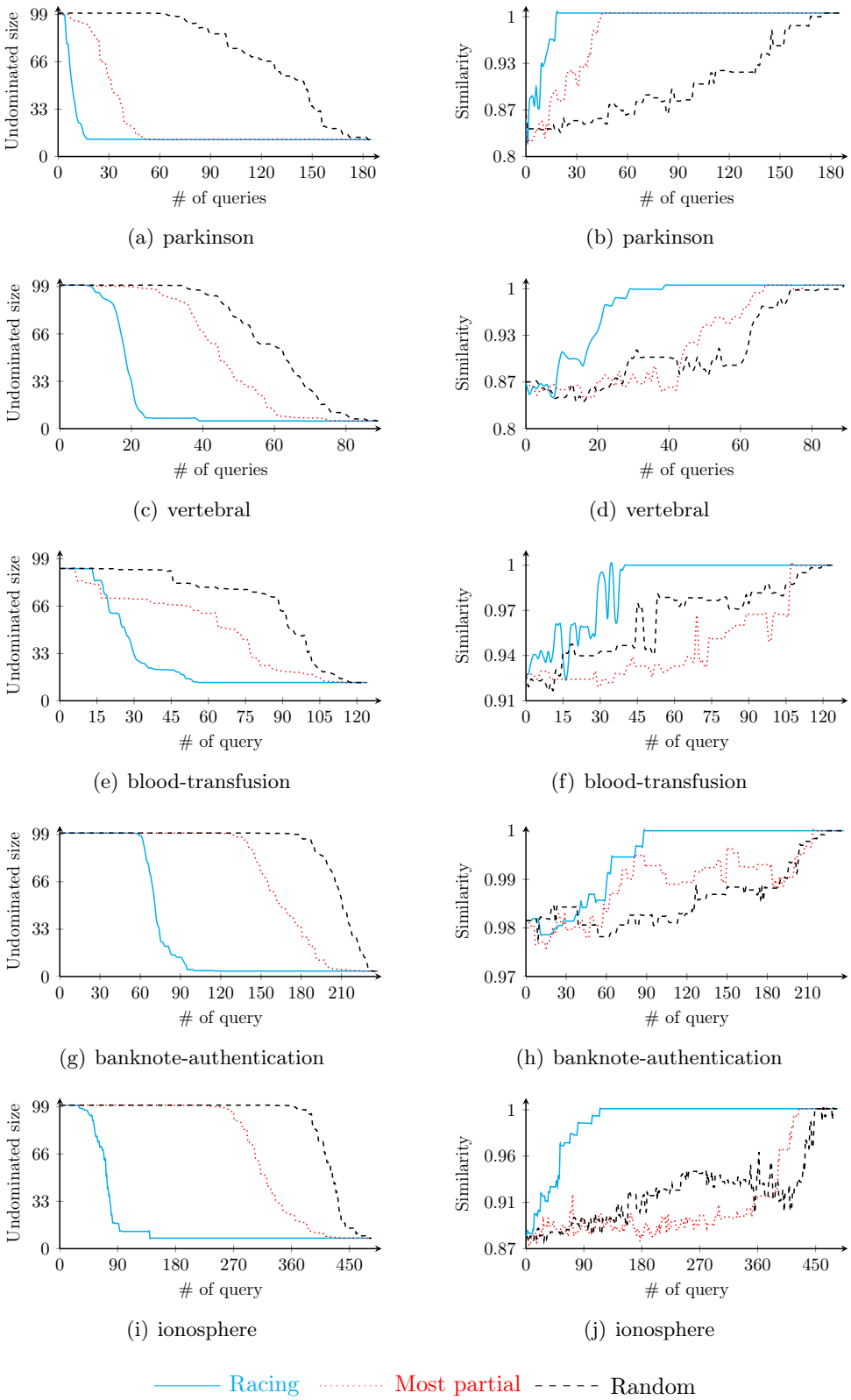
To make comparisons, the two following baseline querying schemes are also used:

- a random querying strategy, where, each time, a set-valued label is chosen randomly,
- and a query-by-committee (QBC) strategy in which each model is allowed to vote on the labellings of query candidates. The most informative query is considered to be the instance for which they most disagree. The disagreement measure used is the **vote entropy**:

$$\boldsymbol{x}^*_{\text{VE}} = \arg\max_{\boldsymbol{x}} -\sum_{m=1}^{M} \frac{V_{\boldsymbol{x}}(y_m)}{S} \log \frac{V_{\boldsymbol{x}}(y_m)}{S}$$

where $V_{\boldsymbol{x}}(y_m)$ denotes the number of models predicting class y_m for a given instance $\boldsymbol{x}$, and $S = |\Theta^*|$ denotes the number of models in the committee.

The experimental results for the case of interval-valued features and set-valued labels are given in Figures 3.4-3.5 and 3.6-3.7, respectively.



(a) parkinson (b) parkinson

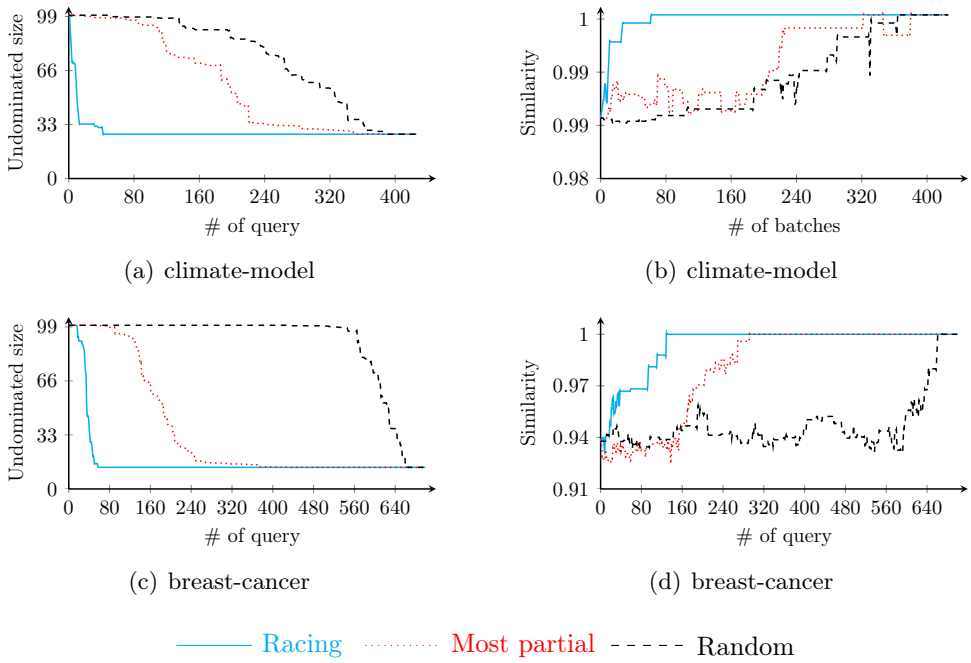
(c) vertebral (d) vertebral

(e) blood-transfusion (f) blood-transfusion

(g) banknote-authentication (h) banknote-authentication

(i) ionosphere (j) ionosphere

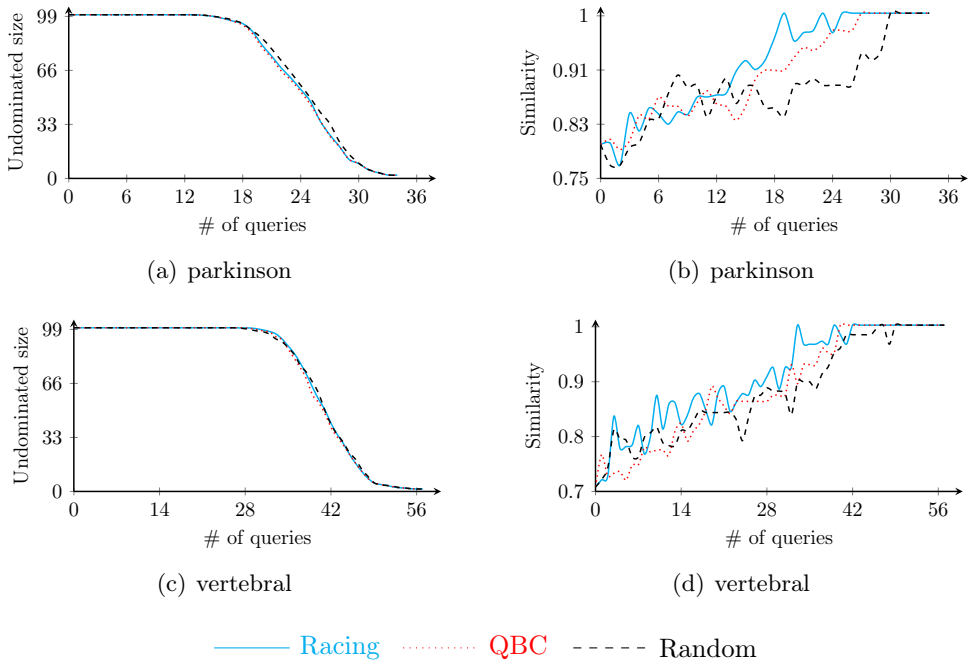
FIGURE 3.4: Experiments for interval-valued features data with preferred model



(a) climate-model (b) climate-model

(c) breast-cancer (d) breast-cancer

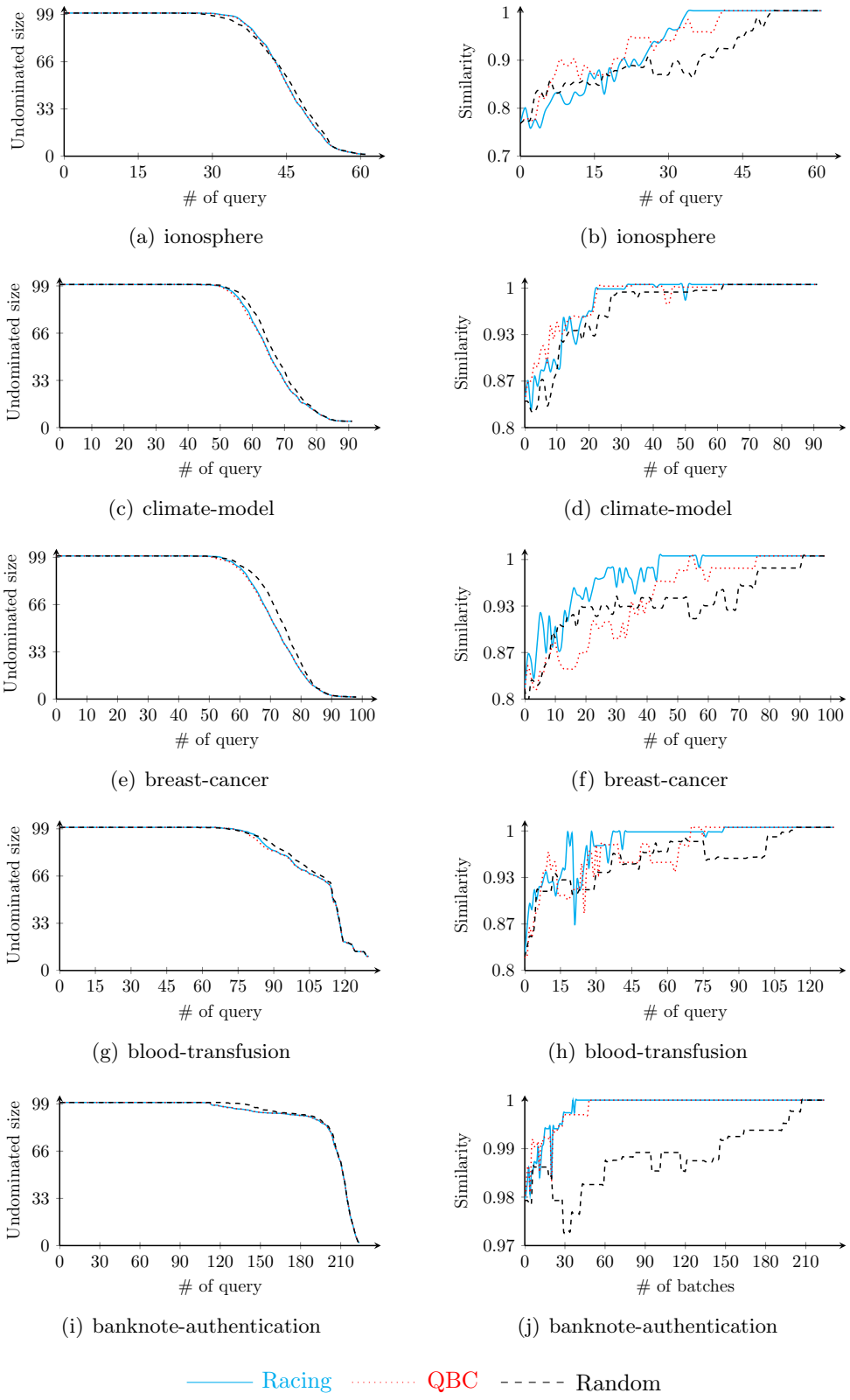
FIGURE 3.5: Experiments for interval-valued features data with preferred model



(a) parkinson (b) parkinson

(c) vertebral (d) vertebral

FIGURE 3.6: Experiments for set-valued labels data with preferred model



(a) ionosphere (b) ionosphere

(c) climate-model (d) climate-model

(e) breast-cancer (f) breast-cancer

(g) blood-transfusion (h) blood-transfusion

(i) banknote-authentication (j) banknote-authentication

FIGURE 3.7: Experiments for set-valued labels data with preferred model

In the case of set-valued labels, we can see that there are only slight differences between the methods. This result was expected, since, in the case of binary classification, partial labels are completely missing labels. Querying partial labels is thus equivalent to standard active learning methods like QBC. A lot of queries are needed to significantly reduce the set of undominated models and to converge through the best model. Also, the random strategy has performances that are often comparable to the active learning ones. In contrast, the performances of our approach are much better than the others in the case of interval-valued features. One can see that the size of the set of undominated models is very quickly reduced and that our racing algorithm converges faster than the other approaches to the winning model.

In order to provide some insights about the potential difficulties of adapting our method to other models, the next section discuss briefly computational issues by building upon the results obtained for SVM.

3.3.4 Discussion on computational issues

The reader may have noticed that the section devoted to SVM with interval-valued features was quite long, and presented more complex methods than the one about set-valued labels. Such an observation extends beyond SVM, and we try in this section to give some reasons why we may expect the problem of interval-valued features to be more complex than the problem of set-valued labels. As with the previous sections, we will stick to the case of $0-1$ loss functions. We will first provide some general remarks about the implementation of our generic approach, and then will shortly discuss how results obtained for the SVM case could be extended to monotone models in general.

General discussion

A first remark is that when we have a partial data $(\mathbf{X}_n, y_n)$ with interval-valued features, a query q_n^p will not make the data precise unless only one feature is partial, but will transform $\mathbf{X}_n$ into $\mathbf{X}_n^{q_n^p} = \times_{p' \neq p} X_n^{p'} \times x_n^p$. In contrast, querying a partial data $(\boldsymbol{x}_n, Y_n)$ with set-valued label Y_n guarantees that the queried data becomes the precise data $(\boldsymbol{x}_n, y_n^{q_n})$, hence guaranteeing that the loss with respect to any model θ_l will also become precise.

Let us now consider the problem of computing bounds of loss functions and potential effect of queries, with a focus on pairs of models and on the case where partial data will induce imprecision in the loss functions of both models, which constitute the most difficult aspects of our approach (our conclusions also apply to other calculations, yet these are typically easier to solve for both interval-valued features and set-valued labels).

Let us first consider the computations of $\underline{\ell}_{k-l}$: in the case of set-valued label Y_n, we do have

$$\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n) = \begin{cases} 0 & \text{if } \theta_k(\boldsymbol{x}_n) = \theta_l(\boldsymbol{x}_n) \vee \{\theta_k(\boldsymbol{x}_n), \theta_l(\boldsymbol{x}_n)\} \cap Y_n = \emptyset \\ -1 & \text{else} \end{cases} \tag{3.35}$$

as the first case describes the only situations where we cannot find a label $y_n \in Y_n$ such that $\theta_k(\boldsymbol{x}_n) = y_n$ and $\theta_l(\boldsymbol{x}_n) \neq y_n$. These conditions are rather easy to check in practice. In contrast, when one has interval-valued features, or more generally

set-valued features $\mathbf{X}_n$ with a precise label y_n, we have that

$$\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = \begin{cases} 1 & \text{if } \forall \boldsymbol{x}_n \in \mathbf{X}_n, \theta_k(\boldsymbol{x}_n) \neq y_n \wedge \theta_l(\boldsymbol{x}_n) = y_n \\ -1 & \text{if } \exists \boldsymbol{x}_n \in \mathbf{X}_n \text{ s.t. } \theta_k(\boldsymbol{x}_n) = y_n \wedge \theta_l(\boldsymbol{x}_n) \neq y_n \\ 0 & \text{else} \end{cases} \tag{3.36}$$

with the last case corresponding to the situation where we can only find[1] $\boldsymbol{x}_n \in \mathbf{X}_n$ such that either $\theta_k(\boldsymbol{x}_n) = \theta_l(\boldsymbol{x}_n) = y_n$, or $\theta_k(\boldsymbol{x}_n) \neq y_n$ and $\theta_l(\boldsymbol{x}_n) \neq y_n$. In contrast with Equation (3.35) whose conditions are easily checked provided $\theta_k(\boldsymbol{x}_n)$ and $\theta_l(\boldsymbol{x}_n)$ are easy to compute (this is the greatest majority of model-based learning methods), identifying which case of Equation (3.36) does apply is more complex and highly depends on the properties of the considered learning method.

Similar conclusions can be drawn to compute the effect $J_{q_n^p}(\theta_k, \theta_l)$ of a possible query. In the case of a set-valued label Y_n, we can directly extend the observation made in Proposition 17 for SVM to have that

$$J_{q_n}(\theta_k, \theta_l) = 1 \text{ iff } \underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n) = -1$$

where $\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n) = -1$ is given by the general and usually easy to estimate Equation (3.35). In contrast, we cannot extend Proposition 15 to arbitrary models when we have interval-valued features. Of course we still have that $J_{q_n^p}(\theta_k, \theta_l) = 0$ when $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = 1$, as it cannot be increased by any query. Yet, in the other cases, one must check that the conditions to have an increase of $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$ are met at least for one value $x_n^p \in X_n^p$, and we do not see how to provide a generic, efficient algorithmic procedure to check them without considering the specificities of the considered model.

The case of monotone models

In the case of the SVM methods, Proposition 14 uses the fact that linear functions are monotonic in every dimension $\mathcal{X}^p$. Note that our analysis could be extended easily to all monotonic models, such as logistic regression or models based on Choquet Integral [86] and more generally on non-additive and fuzzy integrals [37].

As an illustration of this fact, let us consider the case of the logistic regression model. Keeping $\mathcal{X} = \mathbb{R}^P$ and the output space $\mathcal{Y} = \{-1, 1\}$ encoding the two possible classes, the logistic regression corresponding to a model θ_l can be read[2] as

$$\theta_l(\boldsymbol{x}_n) = \ln \frac{P_l(1 \,|\, \boldsymbol{x}_n)}{P_l(-1 \,|\, \boldsymbol{x}_n)} = \sum_{p=0}^{P} w_l^p x_n^p,$$

with $P_l(. \,|\, \boldsymbol{x}_n)$ the posterior probabilities induced by model θ_l, and vector $\boldsymbol{w}_l$ its parameters with the convention $x_n^0 = 1$. This model obviously shares with the SVM that it is monotone in each of its parameters, and in the case of the $0-1$ loss function, we also have

$$\ell_l(y_n, \boldsymbol{x}_n) = \begin{cases} 0 \text{ if } y_n \cdot \theta_l(\boldsymbol{x}_n) \geq 0 \\ 1 \text{ if } y_n \cdot \theta_l(\boldsymbol{x}_n) < 0. \end{cases} . \tag{3.37}$$

Indeed, if $\theta_l(\boldsymbol{x}_n) > 0$, we have $P_l(1 \,|\, \boldsymbol{x}_n) \geq P_l(-1 \,|\, \boldsymbol{x}_n)$, hence predicting $\hat{y}_n = 1$. If we consider now that the features $\boldsymbol{x}_n$ are imprecisely known (as said in the previous

[1] In addition to those possible $\boldsymbol{x}_n$ for which $\theta_k(\boldsymbol{x}_n) \neq y_n$ and $\theta_l(\boldsymbol{x}_n) = y_n$.

[2] The adopted formulation allows us to better shows the similarities with the SVM case.

section, the major computational difficulties will mostly happen in the case of set-valued features), and that $X_n^p = [a_n^p, b_n^p]$ (note that we still have $X_n^0 = [1, 1]$), we can again easily determine when $(\mathbf{X}_n, y_n)$ will be imprecise (1) w.r.t. a model θ_l and (2) w.r.t. both models θ_k and θ_l. Clearly, for the first case, we will have

$$[\underline{\theta}_l(\mathbf{X}_n), \overline{\theta}_l(\mathbf{X}_n)] = \left[\sum_{w_l^p \geq 0} w_l^p a_n^p + \sum_{w_l^p < 0} w_l^p b_n^p, \sum_{w_l^p \geq 0} w_l^p b_n^p + \sum_{w_l^p < 0} w_l^p a_n^p \right],$$

and $(\mathbf{X}_n, y_n)$ will be imprecise w.r.t. θ_l if and only if it contains the value 0 (arguments are similar to the one of the SVM case). Let us now consider the case of not one but two models θ_k and θ_l, $(\mathbf{X}_n, y_n)$ being imprecise w.r.t. both of them (in the other situations, the same remarks as the one done for the SVM case apply). Without loss of generality, we can assume that $y_n = 1$, and we then have that

$$\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = \begin{cases} 1 & \text{if } \forall \boldsymbol{x}_n \in \mathbf{X}_n, \theta_k(\boldsymbol{x}_n) < 0 \wedge \theta_\ell(\boldsymbol{x}_n) > 0 \\ -1 & \text{if } \exists \boldsymbol{x}_n \in \mathbf{X}_n, \theta_k(\boldsymbol{x}_n) > 0 \wedge \theta_\ell(\boldsymbol{x}_n) < 0 \\ 0 & \text{else .} \end{cases}$$

It is clear that the first case will never happen, as $(\mathbf{X}_n, y_n)$ is imprecise w.r.t. θ_k (so there is an $\boldsymbol{x}_n$ for which θ_k is positive). To check the second condition, we have to know whether we can find $\boldsymbol{x}_n$ with $\theta_l(\boldsymbol{x}_n) < 0$, under the constraint that $\theta_k(\boldsymbol{x}_n) > 0$. This comes down to solve the following linear optimisation problem

$$\inf_{\substack{\boldsymbol{x}_n \in \mathbf{X}_n \\ \theta_k(\boldsymbol{x}_n) > 0}} \sum_{p=0}^{P} w_l^p x_n^p$$

and to check whether it is negative, in which case the lower bound is -1, and 0 otherwise. The methodology is here slightly different than in the SVM case, but still takes advantage of the monotonicity and linearity of the model. Completely implementing our proposal in the case of logistic regression would of course require some additional work (left here to the interested reader), but seems quite doable in the light of the above remarks.

3.4 Application to decision trees

We are going to implement our generic racing approach to the particular case of decision tree classifiers [73, 78]. Decision trees are well-known to be sensitive to changes in the data, hence the importance of querying meaningful data for such classifiers. Similar to the case of binary SVM, we will focus on the settings when either the labels or the features are imprecise.

3.4.1 Set-valued labels

In this section, we consider that the labels of some instances are partially given, but that all the features are precise. A query will simply be denoted by q_n meaning that the label of instance $\boldsymbol{x}_n$ is queried. In the classical setting of decision trees, the input space is $\mathcal{X} = \mathcal{X}^1 \times \ldots \times \mathcal{X}^P \subseteq \mathbb{R}^P$ (where $\mathbb{R}$ is the real line) and the output space is $\mathcal{Y} = \{y_1, \ldots, y_M\}$, where y_m, $m = 1, \ldots, M$, encode all the possible classes. A decision tree θ_l is formally a rooted tree structure consisting of terminal nodes and non-terminal nodes [73, 78]:

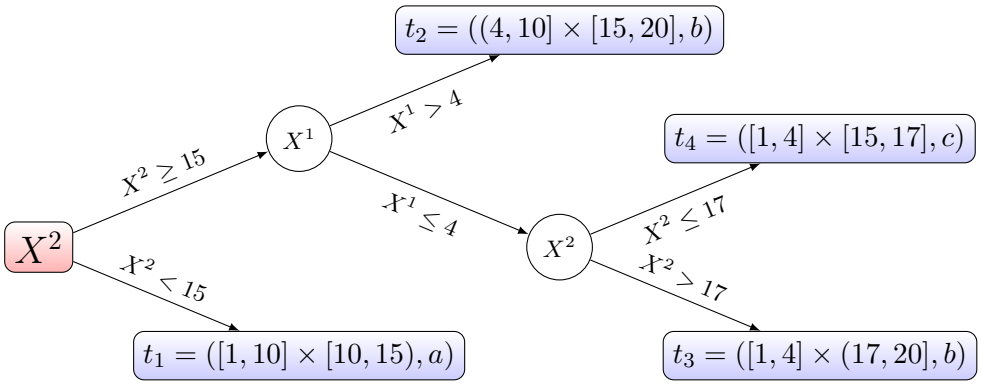


FIGURE 3.8: Decision tree illustration θ_l

- each non-terminal node of the tree is associated to an attribute $\mathcal{X}^p$ ($p \in \{1, \ldots, P\}$), and to each branch issued from this node is associated a condition on this attribute that determines which data of the sample $\mathbf{D}$ go into that branch.

- terminal nodes are called leaves. Each leaf is associated to a predicted class $y_h \in \mathcal{Y}$ and a partition element $\mathbf{A}_h = A_h^1 \times \ldots \times A_h^P$ where $A_h^p \subseteq \mathcal{X}^p$. In the rest of this paper, we will adopt, for each leaf t_h, the following notation

$$t_h = (\mathbf{A}_h, y_h) \tag{3.38}$$

  as such information is enough for the purpose of making prediction for new instances: we have $\theta_l(\boldsymbol{x}_n) = y_h$ for any instance $\boldsymbol{x}_n \in \mathbf{A}_h$.

The next small example illustrates those notations.

Example 13. *Let us consider a given tree trained from data set $\mathbf{D} \in \mathcal{X}^P$ with $P = 2$ attributes, and $M = 3$ classes. Input and output spaces are described as follows:*

$$\mathcal{X}^1 = [1, 10], \mathcal{X}^2 = [10, 20], \mathcal{Y} = \{a, b, c\}.$$

Figure 3.8 illustrates a possible decision tree θ_l for the above setting.

Assume we have new instances $\boldsymbol{x}_1 = (2, 17)$ and $\boldsymbol{x}_2 = (6, 11)$. Then $\boldsymbol{x}_1$ will reach leaf t_4 and be assigned to class $\theta_l(\boldsymbol{x}_1) = c$ while $\boldsymbol{x}_2$ will reach leaf t_1 with an assigned class $\theta_l(\boldsymbol{x}_2) = a$.

We will focus on the classical $0 - 1$ loss function defined as follows: for a given instance $(\boldsymbol{x}_n, y_n)$,

$$\ell_l(y_n, \boldsymbol{x}_n) = \begin{cases} 0 \text{ if } y_n = \theta_l(\boldsymbol{x}_n) \\ 1 \text{ otherwise.} \end{cases} \tag{3.39}$$

In case of partially labelled data, the label is a set $Y_n \subseteq \mathcal{Y}$ instead of a single label. Then the loss in (3.39) becomes an interval $\left[\underline{\ell}_l(Y_n, \boldsymbol{x}_n), \overline{\ell}_l(Y_n, \boldsymbol{x}_n)\right]$ where

$$\underline{\ell}_l(Y_n, \boldsymbol{x}_n) = \min_{y_n \in Y_n} \ell_l(y_n, \boldsymbol{x}_n), \tag{3.40}$$

$$\overline{\ell}_l(Y_n, \boldsymbol{x}_n) = \max_{y_n \in Y_n} \ell_l(y_n, \boldsymbol{x}_n). \tag{3.41}$$

Example 14. *Let us now continue with the data set and the decision tree from Example 13. Assume that instances $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are partially labelled with $Y_1 = \{a, c\}$ and*

$Y_2 = \{b, c\}$, *respectively. Then using (3.40) and (3.41), we can easily get*

$$\left[\underline{\ell}_l(Y_1, \boldsymbol{x}_1), \overline{\ell}_l(Y_1, \boldsymbol{x}_1)\right] = [0, 1],$$
$$\left[\underline{\ell}_l(Y_2, \boldsymbol{x}_2), \overline{\ell}_l(Y_2, \boldsymbol{x}_2)\right] = [1, 1].$$

Let us note that the detail computations in this case is quite similar to the case of SVM, as highlighted in the Section 3.3.4. We first study under which conditions a given partial label introduces imprecision in the empirical risks, before detailing the computation of querying value scores.

Instances introducing imprecision in empirical risk

For a given instance $(\boldsymbol{x}_n, Y_n)$ and a decision tree θ_l, the lower and upper losses in (3.40) and (3.41) can be determined as follows:

$$\underline{\ell}_l(Y_n, \boldsymbol{x}_n) = \begin{cases} 0 \text{ if } \theta_l(\boldsymbol{x}_n) \in Y_n, \\ 1 \text{ otherwise,} \end{cases} \tag{3.42}$$

$$\overline{\ell}_l(Y_n, \boldsymbol{x}_n) = \begin{cases} 0 \text{ if } \{\theta_l(\boldsymbol{x}_n)\} = Y_n, \\ 1 \text{ otherwise.} \end{cases} \tag{3.43}$$

Given a decision tree θ_l, we will say that an instance is imprecise w.r.t. θ_l if

$$\underline{\ell}_l(Y_n, \boldsymbol{x}_n) \neq \overline{\ell}_l(Y_n, \boldsymbol{x}_n). \tag{3.44}$$

The next proposition characterizes simple conditions under which an instance is imprecise w.r.t. θ_l.

Proposition 18. *Given a model θ_l and instance $(\boldsymbol{x}_n, Y_n)$, then $(\boldsymbol{x}_n, Y_n)$ is imprecise w.r.t. θ_l if and only if*

$$\theta_l(\boldsymbol{x}_n) \in Y_n \text{ and } |Y_n| > 1. \tag{3.45}$$

Proof. Let us first note that by definitions we always have

$$\underline{\ell}_l(Y_n, \boldsymbol{x}_n) \leq \overline{\ell}_l(Y_n, \boldsymbol{x}_n).$$

Then combining with condition (3.44) the lower and upper losses of an imprecise instance can be determined explicitly by

$$\underline{\ell}_l(Y_n, \boldsymbol{x}_n) = 0 \text{ and } \overline{\ell}_l(Y_n, \boldsymbol{x}_n) = 1. \tag{3.46}$$

Conditions in (3.39) guarantee that $\underline{\ell}_l(Y_n, \boldsymbol{x}_n) = 0$ is equivalent to condition that $\theta_l(\boldsymbol{x}_n) \in Y_n$. Furthermore, condition $|Y_n| > 1$ ensures that $\overline{\ell}_l(Y_n, \boldsymbol{x}_n) = 1$ (otherwise, $\{\theta_l(\boldsymbol{x}_n)\} = Y_n$, and both lower and upper losses will be 0). □

Proposition 18 simply translates the fact that imprecision can happen only if a partial label could contain the prediction of θ_l. Using Proposition 18, we can conclude that in Example 14, instance $\boldsymbol{x}_1$ is imprecise w.r.t. model θ_l while $\boldsymbol{x}_2$ is precise, even if it has a partial label.

We are now going to investigate the practical computation of the empirical risk bounds of a single model, the pairwise risk bounds in a given set Θ of models and the effect of querying partial labels on those risks. It is easy to see that the empirical risk

bound of a given model can be changed only by querying imprecise instances and the pairwise risk bounds can be changed if the chosen instance is imprecise w.r.t. at least one model. We will then focus on those cases in the next Sections.

Empirical risk bounds and single effect

Equation (3.4) (resp. (3.5)) implies that the computation of $\underline{R}(\theta_l \,|\, \mathbf{D})$ (resp. $\overline{R}(\theta_l \,|\, \mathbf{D})$) can be done by computing $\underline{\ell}_l(Y_n, \boldsymbol{x}_n)$ (resp. $\overline{\ell}_l(Y_n, \boldsymbol{x}_n)$) for $n = 1, \ldots, N$ and then by summing the obtained values. Therefore, the computation of the lower and upper risks of a given model can be carried out easily after determining the lower and upper losses of each instance.

Before going to present conditions under which a query q_n have an effect on modifying the interval $[\underline{R}(\theta_l \,|\, \mathbf{D}), \overline{R}(\theta_l \,|\, \mathbf{D})]$ (or in other words $E_{q_n}(\theta_l) = 1$), let us first note that a query q_n is effective if and only if $[\underline{\ell}_l(Y_n, \boldsymbol{x}_n), \overline{\ell}_l(Y_n, \boldsymbol{x}_n)]$ can be modified. Then, as pointed out in the next proposition, such effect (i.e $E_{q_n}(\theta_l) = 1$) will simply hold for all imprecise instances.

Proposition 19. *Given a model θ_l and an instance $(\boldsymbol{x}_n, Y_n)$, then $E_{q_n}(\theta_l) = 1$ if and only if $(\boldsymbol{x}_n, Y_n)$ is imprecise w.r.t. θ_l.*

Proof. Firstly, it is easy to see that querying any instance that is precise w.r.t. θ_l will not help to modify $[\underline{\ell}_l(Y_n, \boldsymbol{x}_n), \overline{\ell}(Y_n, \theta_l(\boldsymbol{x}_n)]$. Furthermore, (3.46) implies that q_n have an effect by either increasing $\underline{\ell}_l(Y_n, \boldsymbol{x}_n)$ or decreasing $\overline{\ell}_l(Y_n, \boldsymbol{x}_n)$. We will now show that at least one of such losses can be changed after querying any imprecise instance $(\boldsymbol{x}_n, Y_n)$.

Assuming that $y_n^{q_n}$ is the label we get after query q_n, then either $y_n^{q_n} = \theta_l(\boldsymbol{x}_n)$ or $y_n^{q_n} \neq \theta_l(\boldsymbol{x}_n)$. In the first case, both of lower and upper losses will be 0 after performing q_n while both lower and upper losses will be 1 in the latter case. In other words, $E_{q_n}(\theta_l) = 1$ if $(\boldsymbol{x}_n, Y_n)$ is imprecise w.r.t. θ_l. □

Computation of pairwise risk bounds and the effect $J_{q_n}(\theta_k, \theta_l)$ will be investigated in the next Section. Again, if an instance is precise w.r.t. both models, then querying it will not affect the pairwise risk bounds. Therefore, we will focus our interest on instances that are imprecise with respect to at least one model.

Pairwise risk bounds and effect

Let us now focus on how to compute, for a pair of models θ_k and θ_l, the corresponding pairwise risk $\underline{R}(\theta_{k-l} \,|\, \mathbf{D})$ and whether a query q_n can increase this risk. The computation will be treated in two cases: when an instance is imprecise w.r.t. only one model and when an instance is imprecise w.r.t. both.

First note that, similarly to the empirical risk bounds of a unique model, the computation of $\underline{R}(\theta_{k-l} \,|\, \mathbf{D})$ can be carried out by simply summing up the values $\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n)$ for all $(\boldsymbol{x}_n, Y_n)$ with

$$\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n) = \inf_{y_n \in Y_n} \big[\ell_k(y_n, \boldsymbol{x}_n) - \ell_l(y_n, \boldsymbol{x}_n)\big].$$

Furthermore, a query q_n can increase $\underline{R}(\theta_{k-l} \,|\, \mathbf{D})$ if and only if it can increase the value $\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n)$. This is why, in this section, we will focus on computing $\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n)$ and its possible change after a query q_n.

Case 1: Imprecision with respect to one model

We are now going to present the computation of $\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n)$ and the conditions under which $J_{q_n}(\theta_k, \theta_l) = 1$.

Proposition 20. *Given $(\boldsymbol{x}_n, Y_n)$ and two models θ_k and θ_l s.t. $\boldsymbol{x}_n$ is imprecise w.r.t. one and only one model, then we have*

$$\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n) = \ell_k(Y_n, \boldsymbol{x}_n) - 1 \qquad \text{if } \boldsymbol{x}_n \text{ imprecise w.r.t. } \theta_l, \tag{3.47}$$

$$\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n) = 0 - \ell_l(Y_n, \boldsymbol{x}_n) \qquad \text{if } \boldsymbol{x}_n \text{ imprecise w.r.t. } \theta_k. \tag{3.48}$$

Proof. Since instance $\boldsymbol{x}_n$ is imprecise w.r.t. only one model, the imprecision is only associated to such a model, and we can select the worst case: $\ell_l(y_n, \boldsymbol{x}_n) = 1$ for Equation (3.47) and $\ell_k(y_n, \boldsymbol{x}_n) = 0$ for Equation (3.48). □

Now we are going to study under which conditions a query q_n can increase $\underline{R}(\theta_k, \theta_l)$. As the given instance is imprecise w.r.t. only one model, it can only increase the pairwise risk by either increasing $\underline{\ell}_k(Y_n, \boldsymbol{x}_n)$ or decreasing $\overline{\ell}_l(Y_n, \boldsymbol{x}_n)$. As shown in the next Proposition, this can always happen, meaning that we systematically have $J_{q_n}(\theta_k, \theta_l) = 1$ in this case.

Proposition 21. *Given $(\boldsymbol{x}_n, Y_n)$ and two models θ_k and θ_l s.t $\boldsymbol{x}_n$ is imprecise w.r.t. one and only one model, then query q_n can always increase $\underline{\ell}_{k-l}$, or in other words $J_{q_n}(\theta_k, \theta_l) = 1$.*

Proof. We investigate the case where $(\boldsymbol{x}_n, Y_n)$ is imprecise w.r.t. θ_k, the case for θ_l can be treated similarly.

Assuming that $(\boldsymbol{x}_n, Y_n)$ is imprecise w.r.t. θ_k, then Proposition 19 ensures that there always exists a label $y_n \in Y_n$ such that the lower bound $\underline{\ell}_k(Y_n, \boldsymbol{x}_n)$ will be increased to 1 after query q_n.

Similar claim about decreasing the upper bound $\overline{\ell}_l(Y_n, \boldsymbol{x}_n)$ can be carried when $(\boldsymbol{x}_n, Y_n)$ is imprecise w.r.t. θ_l. □

Case 2: Imprecision with respect to both models

For the cases where $\boldsymbol{x}_n$ is imprecise w.r.t. both models θ_k and θ_l, the computation of $\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n)$ and the conditions under which $J_{q_n}(\theta_k, \theta_l) = 1$ will be investigated separately in two circumstances: when $\theta_k(\boldsymbol{x}_n) = \theta_l(\boldsymbol{x}_n)$ and when $\theta_k(\boldsymbol{x}_n) \neq \theta_l(\boldsymbol{x}_n)$.

Proposition 22. *Given $(\boldsymbol{x}_n, Y_n)$ and two models θ_k and θ_l s.t $\boldsymbol{x}_n$ is imprecise w.r.t. both models, then the following results hold*

- *if $\theta_k(\boldsymbol{x}_n) = \theta_l(\boldsymbol{x}_n)$, then*

$$\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n) = 0 \text{ and } J_{q_n}(\theta_k, \theta_l) = 0.$$

- *if $\theta_k(\boldsymbol{x}_n) \neq \theta_l(\boldsymbol{x}_n)$, then*

$$\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n) = -1 \text{ and } J_{q_n}(\theta_k, \theta_l) = 1.$$

Proof.

- When $\theta_k(\boldsymbol{x}_n) = \theta_l(\boldsymbol{x}_n)$, then $\ell(y_n, \theta_k(\boldsymbol{x}_n)) = \ell(y_n, \theta_l(\boldsymbol{x}_n))$ for any value of $y_n \in Y_n$. Therefore, we always have

$$\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n) = \overline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n) = \ell_{k-l}(Y_n, \boldsymbol{x}_n) = 0.$$

  Furthermore, for any label $y \in Y_n$ to be given after performing query q_n, the lower difference (i.e, $\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n)$) will be 0. Or in other words, if $\theta_k(\boldsymbol{x}_n) = \theta_l(\boldsymbol{x}_n)$, then we can simply conclude that $\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n) = 0$ and $J_{q_n}(\theta_k, \theta_l) = 0$.

- In case $\theta_k(\boldsymbol{x}_n) \neq \theta_l(\boldsymbol{x}_n)$, as pointed out in Proposition 18, $\boldsymbol{x}_n$ being imprecise w.r.t. both models implies that

$$\theta_k(\boldsymbol{x}_n) \in Y_n \text{ and } \theta_l(\boldsymbol{x}_n) \in Y_n. \tag{3.49}$$

Then there always exists a label y_n in Y_n (i.e $y_n = \theta_k(\boldsymbol{x}_n)$) s.t. model θ_k returns a true prediction while θ_l returns a wrong one. In other words, we have $\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n) = -1$. The effect $J_{q_n}(\theta_k, \theta_l) = 1$ follows simply by assuming that label $y = \theta_l(\boldsymbol{x}_n)$ will be given after querying $\boldsymbol{x}_n$ which implies that $\underline{\ell}_{k-l}(Y_n, \boldsymbol{x}_n)$ will be increased into 1.

□

The next section provides practical algorithms to perform a single querying step.

Algorithms

Algorithm 7 summarizes the complete procedure to perform an iteration of our querying strategy. Sub-routines are described in other algorithms. Algorithm 8 computes the individual risk bounds of every model, according to the corresponding values of $\underline{\ell}_l(Y_n, \boldsymbol{x}_n), \overline{\ell}_l(Y_n, \boldsymbol{x}_n)$. Algorithm 9 simply summarises the model selection procedure, that will also be used in the case of interval-valued features.

Finally, Algorithm 10 summarises the main procedure that determines the value of the different possible queries, allowing us to pick the best one among all the possible ones. Let us now analyse the complexity of this procedure. Lines 1-2 of Algorithm 7 is in $\mathcal{O}(S \times N)$, as Algorithm 8 is called S times and is in $\mathcal{O}(N)$. Line 3 of Algorithm 7 is in $\mathcal{O}(S)$. Finally, since Algorithm 10 is in $\mathcal{O}(S)$, lines 4-5 of Algorithm 10 are in $\mathcal{O}(S \times N)$. So the overall complexity of Algorithm 10 is in $\mathcal{O}(S \times N)$, meaning that the approach is computationally affordable.

Algorithm 7: A single step to query set-valued data.

Input: Training data set $\mathbf{D} = \{(\mathbf{x}_n, Y_n)\}_{n=1}^N$,
label set $\mathcal{Y} = \{y_1, \ldots, y_M\}$, set of undominated models Θ^*.
Output: The optimal query q_{n^*}

1 **foreach** $\theta_k \in \mathcal{Y}$ **do**
2 Compute empirical risk $(\underline{R}(\theta_k \,|\, \mathbf{D}), \overline{R}(\theta_k))$ bounds using Alg. 8;
3 Determine the best model m_{k^*} and the undominated model set Θ using Alg. 9;
4 **foreach** $n = 1, \ldots, N$ **do**
5 Determine the query effect value $Value(q_n)$ using Alg. 10;
6 Determine $q_{n^*} = \arg\max_n Value(q_n)$;

3.4.2 Interval-valued features

We now deal with the case of interval-valued features, which is much more involved than the case of partial labels, yet still manageable from a computational point of view. Such additional difficulties may explain why there are very few active learning methods dealing with missing features, and none (to our knowledge) dealing with partially known features, at least to our knowledge.

Algorithm 8: Compute the empirical risk bounds $(\underline{R}(\theta_l \,|\, \mathbf{D}), \overline{R}(\theta_l \,|\, \mathbf{D}))$.

Input: Training data set $\mathbf{D} = \{(\boldsymbol{x}_n, Y_n)\}_{n=1}^{N}$,
label set $\mathcal{Y} = \{y_1, \ldots, y_M\}$, model θ_l.
Output: Empirical risk bounds $(\underline{R}(\theta_l \,|\, \mathbf{D}), \overline{R}(\theta_l \,|\, \mathbf{D}))$

$\underline{R}(\theta_l \,|\, \mathbf{D}) = 0$, $\overline{R}(\theta_l \,|\, \mathbf{D}) = 0$;
foreach $n = 1, \ldots, N$ **do**
  if $|Y_n| > 1$ **then**
    if $\theta_l(\boldsymbol{x}_n) \notin Y_n$ **then** $\underline{R}(\theta_l \,|\, \mathbf{D}) = \underline{R}(\theta_l \,|\, \mathbf{D}) + 1$;
    $\overline{R}(\theta_l \,|\, \mathbf{D}) = \overline{R}(\theta_l \,|\, \mathbf{D}) + 1$
  else if $\{\theta_l(\boldsymbol{x}_n)\} \neq Y_n$ **then** $\underline{R}(\theta_l \,|\, \mathbf{D}) = \underline{R}(\theta_l \,|\, \mathbf{D}) + 1$,
  $\overline{R}(\theta_l \,|\, \mathbf{D}) = \overline{R}(\theta_l \,|\, \mathbf{D}) + 1$;

Algorithm 9: Determine the best model θ_{k^*} and the undominated model set Θ^*.

Input: Model set Θ, empirical risk bounds $\{(\underline{R}(\theta_k \,|\, \mathbf{D}), \overline{R}(\theta_k \,|\, \mathbf{D}))|\forall \theta_k \in \Theta\}$
Output: The best model θ_{k^*} and the undominated set Θ^*

$\theta_{k^*} = \arg\min_{\theta_k \in \Theta} \underline{R}(\theta_k \,|\, \mathbf{D})$;
$\overline{R}_{\min} = \min_{\theta_k \in \Theta} \overline{R}(\theta_k \,|\, \mathbf{D})$;
foreach $\theta_k \in \Theta$ **do**
  if $\underline{R}(\theta_k \,|\, \mathbf{D}) > \overline{R}_{\min}$ **then Remove** θ_k **from** Θ;

Algorithm 10: Determine the effect value of a query $Value(q_n)$.

Input: Training instance $(\boldsymbol{x}_n, Y_n)$, undominated model set Θ^*.
Output: The querying effect value $Value(q_n)$

Initialize $E_{q_n}(\theta_{k^*}) = 0$, $J_{q_n} = 0$;
if $|Y_n| > 1$ *and* $\theta_{k^*}(\boldsymbol{x}_n) \in Y_n$ **then**
  $E_{q_n}(\theta_{k^*}) = 1$;
  foreach $\theta_k \in \Theta$ *and* $k \neq k^*$ **do**
    if $\theta_k(\boldsymbol{x}_n) \in Y_n$ *and* $\theta_k(\boldsymbol{x}_n) \neq \theta_{k^*}(\boldsymbol{x}_n)$ **then** $J_{q_n} = J_{q_n} + 1$;
    else if $\theta_k(\boldsymbol{x}_n) \notin Y_n$ **then** $J_{q_n} = J_{q_n} + 1$;
else if $|Y_n| > 1$ **then**
  foreach $\theta_k \in \Theta$ *and* $k \neq k^*$ **do**
    if $\theta_k(\boldsymbol{x}_n) \in Y_n$ **then** $J_{q_n} = J_{q_n} + 1$;
$Value(q_n) = E_{q_n}(\theta_{k^*}) + J_{q_n}$;

Instances introducing imprecision in empirical risk

Before going further, let us remind that, for a given tree θ_l, each terminal node (which is sufficient in later analysis) is associated with a partition element

$$\mathbf{A}_h = A_h^1 \times \ldots, \times A_h^P, \tag{3.50}$$

where A_h^p can be a closed, open or semi-closed interval in our case. However, for the sake of practical implementation and exposure, we will from now on assume that A_h^p is a closed interval.

Since we work with interval-valued feature data, for each instance $(\mathbf{X}_n, y_n)$, its feature $\mathbf{X}_n$ can be represented as a hyper-cube (similar to terminal node in (3.50)) denoted by

$$\mathbf{X}_n = X_n^1 \times \ldots \times X_n^P. \tag{3.51}$$

Then, the intersection between partition elements and/or partial instances is nothing else but the one of two hyper-cubes. Given two such hyper-cubes $\mathbf{U} = U^1 \times \ldots \times U^P$ and $\mathbf{V} = V^1 \times \ldots \times V^P$, their corresponding intersection, denoted by $U \cap V$ is

$$U \cap V = \times_{p=1}^P U^p \cap V^p. \tag{3.52}$$

(3.52) provides a practical way to check whether the intersection of two cubic forms is non-empty. More precisely, we have that $\mathbf{U} \cap \mathbf{V} \neq \emptyset$ iff

$$U^p \cap V^p \neq \emptyset, \forall p = 1 \ldots, P. \tag{3.53}$$

As for the case of partial labels, an instance $(\mathbf{X}_n, y_n)$ is said to be imprecise w.r.t. a decision tree θ_l if

$$\exists \boldsymbol{x}_n, \boldsymbol{x}_n^{'} \in \mathbf{X}_n \text{ s.t } \ell_l(y, \boldsymbol{x}_n) \neq \ell_l(y, \boldsymbol{x}_n^{'}). \tag{3.54}$$

Furthermore, as an instance can intersect several partition elements which are possibly associated to different labels, then (3.54) is equivalent to the following relation

$$\exists \mathbf{A}_h, \mathbf{A}_{h'} \text{ s.t } \mathbf{X}_n \cap \mathbf{A}_h \neq \emptyset, \mathbf{X}_n \cap \mathbf{A}_{h'} \neq \emptyset \text{ and } y_h = y_n \text{ and } y_{h'} \neq y_n. \tag{3.55}$$

Note that (3.55) can be easily determined using (3.53). The following Example gives illustrations of an imprecise instance w.r.t. a given decision tree.

Example 15. *Figure 3.9 gives an example of a tree θ_l and two instances,* $(\mathbf{X}_1, 0)$, $(\mathbf{X}_2, 1)$.

It is easy to see that $(\mathbf{X}_1, 0)$ *is a precise instance since it only intersects with a partition element associated to label* 0. *However* $\mathbf{X}_2$ *is imprecise since* (3.55) *holds. More precisely,* $(\mathbf{X}_2, 1)$ *intersects with* $\mathbf{A}_5$ *and* $\mathbf{A}_6$ *whose associated labels are different.*

Empirical risk bounds and single effect

We are now going to investigate how risk bounds $[\underline{R}(\theta_l), \overline{R}(\theta_l)]$ can be computed efficiently from data $\{(\mathbf{X}_1, y_1)\}_{n=1}^N$ by computing extreme bounds $\underline{\ell}_l(y_n, \mathbf{X}_n)$, $\overline{\ell}_l(y_n, \mathbf{X}_n)$, and how the potential effect of a query q_n^p (q_n^p corresponding to ask the true value within X_n^p) on those bounds can be estimated.

Let us first study how bounds on loss functions can be estimated. Similarly to the case of set-valued labels, an instance $\mathbf{X}_n$ will get the imprecise empirical risk

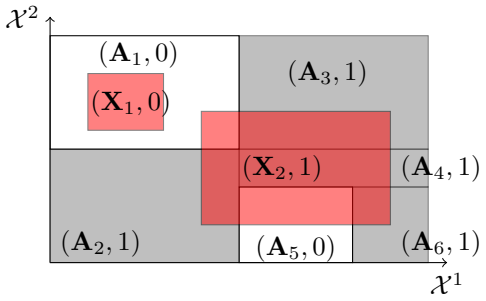


FIGURE 3.9: Example of imprecise instance

bounds $\left[\underline{\ell}_l(y_n, \mathbf{X}_n), \overline{\ell}_l(y_n, \mathbf{X}_n)\right] = [0, 1]$ iff it satisfies condition (3.55). Otherwise, the corresponding loss is precise and such an instance can be discarded from the querying process. For example, in Figure 3.9, we can see that $\left[\underline{\ell}_l(y_1, \mathbf{X}_1), \overline{\ell}_l(y_1, \mathbf{X}_1)\right] = [0, 0]$ while $\left[\underline{\ell}_l(y_2, \mathbf{X}_2), \overline{\ell}_l(y_2, \mathbf{X}_2)\right] = [0, 1]$. Note that a training instance $(\mathbf{X}_n, y_n)$ is precise if and only if partition elements that intersect with it either are all of label y_n or all different from y_n. To determine whether such a condition holds, let us firstly introduce the following information vectors

$$\mathbf{K} = (k^1, \ldots, k^H) \text{ with } k^h = \begin{cases} 1 \text{ if } \mathbf{X}_n \cap \mathbf{A}_h \neq \emptyset, \\ 0 \text{ otherwise,} \end{cases} \tag{3.56}$$

$$\mathbf{B}_{y_n} = (b^1_{y_n}, \ldots, b^H_{y_n}) \text{ with } b^h_{y_n} = \begin{cases} 0 \text{ if } y_h = y_n, \\ 1 \text{ otherwise,} \end{cases} \tag{3.57}$$

$$\mathbf{C}_{y_n} = (c^1_{y_n}, \ldots, c^H_{y_n}) \text{ with } c^h_{y_n} = \begin{cases} 1 \text{ if } y_h = y_n, \\ 0 \text{ otherwise,} \end{cases} \tag{3.58}$$

with H the number of terminal nodes of the decision tree θ_l. Note that $\mathbf{K}$ can easily be built using (3.53), and that $\mathbf{B}, \mathbf{C}$ have to be built only once. A given training instance $\mathbf{X}_n$ is imprecise w.r.t. θ_l if and only if $(\mathbf{K}\mathbf{B}^\top_{y_n})(\mathbf{K}\mathbf{C}^\top_{y_n}) \neq 0$, where $\mathbf{a}\mathbf{b}^\top$ is the dot product of two vectors $\mathbf{a}$ and $\mathbf{b}$. Before going further, let us note that we can use information vectors to deduce that $\ell_l(y_n, \mathbf{X}_n)$ has the precise value 0 and 1, as this happens when $\mathbf{K}\mathbf{B}^\top_{y_n} = 0$ and $\mathbf{K}\mathbf{C}^\top_{y_n} = 0$, respectively.

One can see that performing a query q^p_n can only change $\mathbf{K}$. Denoting by $\mathbf{K}_{q^p_n}$ the vector resulting from q^p_n, the single effect $E_{q^p_n}(\theta_l) = 1$ if and only if $(\mathbf{K}\mathbf{B}^\top_{y_n})(\mathbf{K}\mathbf{C}^\top_{y_n}) \neq 0$ and $\exists x^p_n \in X^p_n$ s.t $(\mathbf{K}_{q^p_n}\mathbf{B}^\top_{y_n})(\mathbf{K}_{q^p_n}\mathbf{C}^\top_{y_n}) = 0$. Verifying whether such a situation happens can be done by checking the two following conditions

$$\exists x^p_n \in X^p_n \text{ s.t } (\mathbf{K}_{q^p_n}\mathbf{B}^\top_{y_n}) = 0 \text{ or } \exists x^p_n \in X^p_n \text{ s.t } (\mathbf{K}_{q^p_n}\mathbf{C}^\top_{y_n}) = 0 \tag{3.59}$$

We will present detailed developments and computations for the first condition and then present the result for the second one (which can be developed in a similar manner). The definition of $\mathbf{K}$ ensures that only elements of value 1 can change to zero after a query, since reducing X^p_n can only lead to the fact that a non-empty intersection with A_h becomes empty. Furthermore, (3.53) implies that if $k^h = 1$ then for all dimensions $p = 1, \ldots, P$, we have $X^p_n \cap A^p_h \neq \emptyset$. Such an observation ensures that the results after performing q^p_n, $k^h = 0$ if and only if $\exists x^p_n \in X^p_n$ s.t $x^p_n \cap A^p_h = \emptyset$, that is if the intersection with A_h on dimension p can become empty after querying

X_n^p. It then implies that the condition $\exists x_n^p \in X_n^p$ s.t $(\mathbf{K}_{q_n^p}\mathbf{B}_{y_n}^\top) = 0$ is equivalent to the following condition

$$\exists x_n^p \in X_n^p \text{ s.t } x_n^p \cap A_h^p = \emptyset, \forall h \text{ where } k^h b_{y_n}^h = 1, \tag{3.60}$$

or, in other words, there is a value $x_n^p \in X_n^p$ s.t x_n^p does not belong to any of A_h^p for which the condition $k^h b_{y_n}^h = 1$ holds, that is for this value the resulting hyper-cube intersects with no leaves having y_n as prediction. Such a condition comes down to check whether the following assertion is true:

$$X_n^p \setminus \left(\cup_{k^h b_{y_n}^h = 1} A_h^p \right) \neq \emptyset. \tag{3.61}$$

Similarly, to determine whether $\exists x_n^p \in X_n^p$ s.t $(\mathbf{K}_{q_n^p}\mathbf{C}_{y_n}^\top) = 0$, we can simply investigate whether

$$\exists x_n^p \in X_n^p \text{ s.t } x_n^p \cap A_h^p = \emptyset, \forall h \text{ when } k^h c_{y_n}^h = 1, \tag{3.62}$$

which can be done by checking the condition

$$X_n^p \setminus \left(\cup_{k^h c_{y_n}^h = 1} A_h^p \right) \neq \emptyset. \tag{3.63}$$

The general problem we have to solve is to check whether an interval $X_n^p = \left[a_n^p, b_n^p\right]$ contains a value that is outside the union of some collection of intervals $\left[\underline{d}^i, \overline{d}^i\right]$ (here, the intervals A_h^p satisfying the conditions in (3.61) and (3.63)). Once we notice this, we can rewrite the computational problem in the following form

$$\left[a_n^p, b_n^p\right] \setminus \cup_{i=1}^I \left[\underline{d}^i, \overline{d}^i\right] \neq \emptyset, \text{ when } \forall i = 1, \ldots, I, \left[a_n^p, b_n^p\right] \cap \left[\underline{d}^i, \overline{d}^i\right] \neq \emptyset. \tag{3.64}$$

The intuitive idea is that (3.64) is not satisfied if and only if $\cup_{i=1}^I \left[\underline{d}^i, \overline{d}^i\right]$ is a closed interval including $\left[a_n^p, b_n^p\right]$. Then to check whether (3.64) is satisfied, we just have to firstly check whether $\cup_{i=1}^I \left[\underline{d}^i, \overline{d}^i\right]$ is a closed interval, and if it is, whether it includes $\left[a_n^p, b_n^p\right]$. To check that $\cup_{i=1}^I \left[\underline{d}^i, \overline{d}^i\right]$ is a closed interval comes down to check whether there is a gap in the union of intervals. Let $\{\underline{d}^{(1)}, \ldots, \underline{d}^{(I)}\}$ be the ordered list of lower bounds, or starts of intervals. A gap happens if, when increasing values from a_n^p to b_n^p, all intervals that have been opened are closed before another one starts (as illustrated in Figure 3.10). In formal terms, there exists an index j such that

$$\left|\{\overline{d}^i : \overline{d}^i < \underline{d}^{(j)}\}\right| = j - 1,$$

which expresses the fact that before the jth interval $[\underline{d}^{(j)}, \overline{d}^{(j)}]$ starts, the $j-1$ previous ones are closed, hence their union is not a closed interval. Provided $\cup_{i=1}^I \left[\underline{d}^i, \overline{d}^i\right]$ is a closed interval, then checking whether it includes $\left[a_n^p, b_n^p\right]$ can simply be done by checking that

$$\underline{d}^{(1)} \leq a_n^p \leq b_n^p \leq \overline{d}^{(I)}.$$

For a given interval $\left[a_n^p, b_n^p\right]$ and a set of interval $\left\{\left[\underline{d}^i, \overline{d}^i\right] | i = 1, \ldots, I\right\}$, then whether there is a value within $\left[a_n^p, b_n^p\right]$ that is not included in $\cup_i \left[\underline{d}^i, \overline{d}^i\right]$ (i.e., whether condition (3.64) is satisfied) can be checked using Algorithm 11.

Let us now illustrate how to practically determine the single effect using a simple example.

Example 16. *Consider the tree θ_l and two instances $\mathbf{X}_1$, $\mathbf{X}_2$ illustrated in Figure*

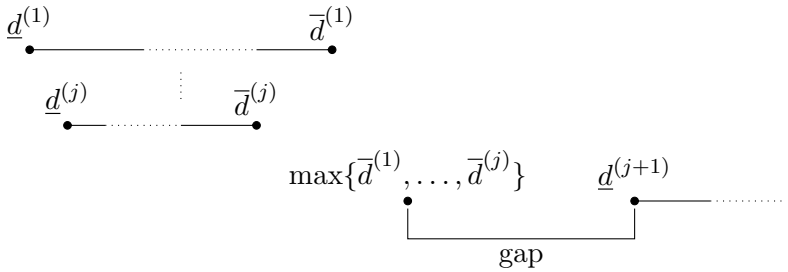


FIGURE 3.10: Case where the union of intervals is not an interval

Algorithm 11: Checking whether the condition (3.64) is satisfied

Input: $\left[a_n^p, b_n^p\right]$, sets $\left\{\left[\underline{d}^i, \overline{d}^i\right] | i = 1, \ldots, I\right\}$ s.t, for $\forall i$, $\left[a_n^p, b_n^p\right] \cap \left[\underline{d}^i, \overline{d}^i\right] \neq \emptyset$
Output: Return $In = 1$ if (3.64) is satisfied and 0 otherwise

```
Order {d̲^1, ..., d̲^I} into {d̲^(1), ..., d̲^(I)} ;
foreach i = 1, ..., I do
    if |{d̄^k : d̄^k < d̲^(i)}| = i − 1, then
        Return In = 1 and Stop the Algorithm
if min_i d̲^i > a_n^p then
    Return In = 1 and Stop the Algorithm
else if b_n^p > max_i d̄^i then
    Return In = 1 and Stop the Algorithm
Return In = 0;
```

3.9. Instance $\mathbf{X}_1$ *is precise w.r.t. the model* θ_l*, hence querying its feature is useless for this model. We then focus on determining the effect of querying the features of* $\mathbf{X}_2$*.*

Using (3.56)-(3.58)*, the information vectors associated to* $\mathbf{X}_2$ *are*

$$\mathbf{K} = (1,1,1,1,1,1) \text{ and } \mathbf{B}_{y_2} = (1,0,0,0,1,0) \text{ and } \mathbf{C}_{y_2} = (0,1,1,1,0,1).$$

Let us now investigate whether $\mathbf{X}_2$ *can become precise (w.r.t. the model* m_k*) by querying its feature* X_2^1*. We have that*

$$\cup_{k^h b_{y_n}^h = 1} A_h^1 = A_1^1 \cup A_5^1$$

as leaves A_1 *and* A_5 *are overlapping with* $\mathbf{X}_2$ *and predict a different class from its true one. We can see on the picture that* $A_1^1 \cup A_5^1$ *is a closed interval that does not includes* X_2^1*. Then, for any value* x_2^1 *belonging to the interval* $(\underline{x}_2^1, \overline{x}_2^1]$ *as illustrated in the Figure 3.11, we have that* $\mathbf{K}_{q_n^p} \mathbf{B}_{y_2}^\top = 0$*. In other words, we have that instance* $\mathbf{X}_2$ *can become a precise instance after querying its feature* X_2^1*.*

Similarly, for the case of querying X_2^2*, we have that*

$$\cup_{k^h b_{y_n}^h = 1} A_h^2 = A_1^2 \cup A_5^2.$$

Since $A_1^2 \cup A_5^2$ *is not a closed interval, then, for any value* x_2^2 *belonging to the interval* $(\underline{x}_2^2, \overline{x}_2^2)$ *(illustrated in the Figure 3.11), we have that* $\mathbf{K}_{q_n^p} \mathbf{B}_{y_2}^\top = 0$*.*

Finally, we conclude that instance $\mathbf{X}_2$ *can become a precise instance after querying either* X_2^1 *or* X_2^2*.*

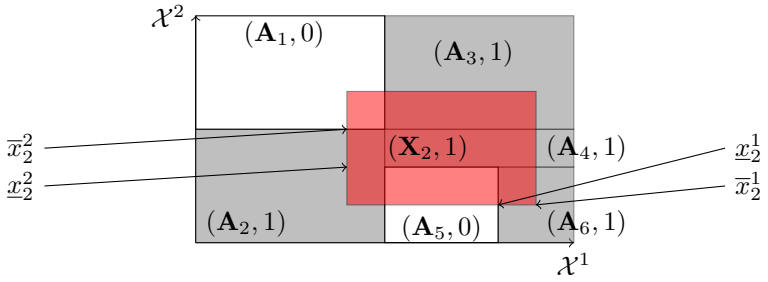


FIGURE 3.11: Example of determining the single effect

Pairwise risk bounds and effect

This section focuses on how to compute, for a pair of models θ_k and θ_l, the corresponding pairwise risk bounds $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$ for all instance $\mathbf{X}_n$ and whether a query q_n^p can increase this risk. In a way similar to the case of set-valued labels (Section 3.4.1), computations will be treated in two cases: when the instance is imprecise w.r.t. only one model; and when it is imprecise for both.

Case 1*: Imprecision with respect to one model*

In case an instance $\mathbf{X}_n$ is imprecise w.r.t. one model (either θ_k or θ_l), the pairwise risk bound $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$ can be determined in a way similar to the case of set-valued labels (Proposition 20). Note that this bound is, in the context of imprecise features, defined as:

$$\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = \inf_{\boldsymbol{x}_n \in \mathbf{X}_n} \big[\ell_k(y_n, \boldsymbol{x}_n) - \ell_l(y_n, \boldsymbol{x}_n)\big].$$

Proposition 23. *Given $(\mathbf{X}_n, y_n)$, and two models θ_k and θ_l s.t $\mathbf{X}_n$ is imprecise w.r.t. one and only one model, we have*

$$\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = \ell_k(y_n, \mathbf{X}_n) - 1 \qquad \text{if } \mathbf{X}_n \text{ imprecise w.r.t. } \theta_l \tag{3.65}$$

$$\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = -\ell_l(y_n, \mathbf{X}_n) \qquad \text{if } \mathbf{X}_n \text{ imprecise w.r.t. } \theta_k. \tag{3.66}$$

Proof. Similar to proof of Proposition 20. □

Then a query q_n^p will have an effect $J_{q_n^p}(\theta_k, \theta_l) = 1$ if either it increases $\underline{\ell}_k(y_n, \mathbf{X}_n)$ or decreases $\overline{\ell}_l(y_n, \mathbf{X}_n)$. The detailed arguments can be found in the next proposition.

Proposition 24. *Given $(\mathbf{X}_n, y_n)$ and two models θ_k and θ_l s.t. $\mathbf{X}_n$ is imprecise w.r.t. one and only one model, then $J_{q_n^p}(\theta_k, \theta_l) = 1$ if and only if one of the following conditions holds*

- *if $\mathbf{X}_n$ is imprecise w.r.t. model θ_k, then $J_{q_n^p}(\theta_k, \theta_l) = 1$ if and only if Equation (3.63) holds for the model m_k.*
- *if $\mathbf{X}_n$ is imprecise w.r.t. model θ_l, then $J_{q_n^p}(\theta_k, \theta_l) = 1$ if and only if Equation (3.61) holds for the model θ_l.*

Proof. Let us start with the case when $\mathbf{X}_n$ is imprecise w.r.t. model θ_k. The condition that Equation (3.63) holds for the model θ_k simply implies that after performing a query q_n^p, the loss $\ell_k(y_n, \mathbf{X}_n)$ becomes precisely 1. Hence it is clear that the pairwise risk bound is increased.

Similarly, when $\mathbf{X}_n$ is imprecise w.r.t. model θ_l, that Equation (3.61) holds implies that after performing a query q_n^p, the loss $\ell_l(y_n, \mathbf{X}_n)$ is precisely 0 which results in increasing $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$. □

Case 2: Imprecision with respect to both models

Note that when an instance $\mathbf{X}_n$ is imprecise with respect to both models θ_k and θ_l, the pairwise risk bounds $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$ can get values in $\{-1, 0, 1\}$. Let us denote by $y_h^{\theta_l}$ the label associated to the partition $\mathbf{A}_h^{\theta_l}$ of a tree θ_l, then the relation between $\mathbf{X}_n$ and leaves of θ_k and θ_l can be encoded in matrix form as follows

$$\mathbf{W}^{k,l} = \left(w_{i,j}^{k,l}\right)_{i=1,\dots,H_k, j=1,\dots,H_l} \tag{3.67}$$

s.t

$$w_{i,j}^{k,l} = \begin{cases} 2 \text{ if } \mathbf{X}_n \cap \mathbf{A}_i^{\theta_k} \cap \mathbf{A}_j^{\theta_l} = \emptyset, \\ 1 \text{ if } \mathbf{X}_n \cap \mathbf{A}_i^{\theta_k} \cap \mathbf{A}_j^{\theta_l} \neq \emptyset, y_i^{\theta_k} \neq y_n, y_j^{\theta_l} = y_n, \\ 0 \text{ if } \mathbf{X}_n \cap \mathbf{A}_i^{\theta_k} \cap \mathbf{A}_j^{\theta_l} \neq \emptyset, y_i^{\theta_k} = y_j^{\theta_l}, \\ -1 \text{ if } \mathbf{X}_n \cap \mathbf{A}_i^{\theta_k} \cap \mathbf{A}_j^{\theta_l} \neq \emptyset, y_i^{\theta_k} = y_n, y_j^{\theta_l} \neq y_n. \end{cases} \tag{3.68}$$

It is easy to see that the matrix $\mathbf{W}^{k,l}$ covers all possible values of $\ell_{k-l}(y_n, \mathbf{X}_n)$, with 2 being an arbitrary value to denote that $\mathbf{X}_n$ prediction does not depend on $\mathbf{A}_i^{\theta_k} \cap \mathbf{A}_j^{\theta_l}$. The pairwise lower risk bound is then simply the minimum value of elements in matrix $\mathbf{W}^{k,l}$ i.e.,

$$\underline{\ell}_{k-l}(y_n, \mathbf{X}_n) = \min_{i,j} w_{i,j}^{k,l}. \tag{3.69}$$

Before going to determine whether a query q_n^p can increase the pairwise risk bound $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$, note that whether $\mathbf{X}_n \cap \mathbf{A}_i^{\theta_k} \cap \mathbf{A}_j^{\theta_l} = \emptyset$ can be easily determined as a consequence of Equation (3.52), as we have

$$\mathbf{X}_n \cap \mathbf{A}_i^{\theta_k} \cap \mathbf{A}_j^{\theta_l} = \times_{p=1}^{P} X_n^p \cap A_{i,p}^{\theta_k} \cap A_{j,p}^{\theta_l}. \tag{3.70}$$

Then for an instance $\mathbf{X}_n$, its corresponding pairwise risk bound w.r.t. two models θ_k and θ_l can be determined explicitly using Equations (3.67) and (3.68). A query q_n^p can increase the pairwise risk bound if and only if it can increase the value of all elements of value $\min_{i,j} w_{i,j}^{k,l}$. Let

$$\mathbf{S}_{min} = \left\{ w_{i',j'}^{k,l} | w_{i',j'}^{k,l} = \min_{i,j} w_{i,j}^{k,l} \right\} \tag{3.71}$$

be the set of such elements, then $J_{q_n^p}(\theta_k, \theta_l) = 1$ if $\exists\ x_n^p \in X_n^p$ s.t after querying X_n^p, all elements in the set $\mathbf{S}_{min}$ are increased.

Note that for a given pair $\left(\mathbf{A}_i^{\theta_k}, \mathbf{A}_j^{\theta_l}\right)$, using (3.53), we have that their intersection is

$$\mathbf{A}_{i,j} = \mathbf{A}_i^{\theta_k} \cap \mathbf{A}_j^{\theta_l} = \times_{p=1}^{P} A_{i,p}^{\theta_k} \cap A_{j,p}^{\theta_l} := \times_{p=1}^{P} A_{i,j}^{p}, \tag{3.72}$$

where $A_{i,j}^p$ is a closed interval, for $p = 1, \dots, P$. Furthermore, a query q_n^p can increase the pairwise risk bound if $\exists\ x_n^p \in X_n^p$ s.t $x_n^p \notin A_{i,j}^p$, for all $w_{i,j}^{k,l} \in \mathbf{S}_{min}$. The next Proposition provides a practical procedure to check whether such a condition holds.

Proposition 25. *Given a training instance $\mathbf{X}_n$ which is imprecise w.r.t. both models θ_k and θ_l, the corresponding $\mathbf{W}^{k,l}$ matrix, assuming that $\min_{i,j} w_{i,j}^{k,l} < 1$, then*

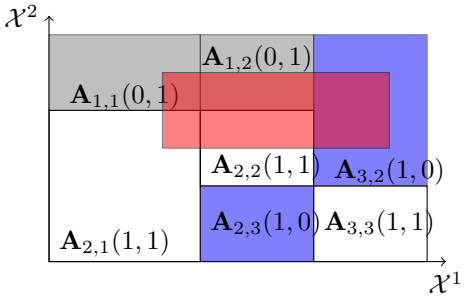


FIGURE 3.12: Example of determining the pairwise effect

$J_{q_n^p}(\theta_k, \theta_l) = 1$ *if and only if*

$$X_n^p \setminus \left(\cup_{i,j|w_{i,j}^{k,l} \in \mathbf{S}_{min}} A_{i,j}^p \right) \neq \emptyset. \tag{3.73}$$

Proof. Let us first note that if (3.73) holds, then $\exists\ x_n^p \in X_n^p$ s.t $x_n^p \notin A_{i,j}^p$, for all $w_{i,j}^{k,l} \in \mathbf{S}_{min}$. Then the corresponding elements $w_{i,j}^{k,l}$ are increased to be 2. It is then resulting in the increasing of $\min_{i,j} w_{i,j}^{k,l}$, or in other words, the pairwise risk bound $\underline{\ell}_{k-l}(y_n, \mathbf{X}_n)$. □

Checking whether Equation (3.73) is true can easily be reformulated in the form of Equation (3.64), Algorithm 11 can then be used to perform the check. In practice, such a check is quadratic in the number of leaves of the models θ_k, θ_l, which remains affordable from a computational standpoint. The next example illustrates how to practically determine the effect of queries on the pairwise risk bounds.

Example 17. *Assume that we have two models θ_1 and θ_2 with 3 leaves each, whose intersection of partition elements is illustrated in Figure 3.12.*

Instance $\mathbf{X}$ covers the red region and has label $y = 1$. From Figure 3.12, we can see that $\mathbf{X}$ is imprecise w.r.t. both models m_1 and m_2, and its corresponding information matrix $\mathbf{W}^{1,2}$ can be determined as follows

$$\mathbf{W}^{1,2} = \begin{bmatrix} 1 & 1 & 2 \\ 0 & 0 & 2 \\ 2 & -1 & 2 \end{bmatrix}$$

Then it is clear that $\underline{\ell}_{1-2}(y, \mathbf{X}) = -1$ and

$$\mathbf{S}_{min} = \{ w_{i',j'}^{k,l} | w_{i',j'}^{k,l} = \min_{i,j} w_{i,j}^{k,l} \} = w_{3,2}^{1,2}.$$

Let us now investigate whether the empirical risk bound $\underline{\ell}_{1-2}(y, \mathbf{X})$ can increase by querying the features of $\mathbf{X}$. It is easy to see that $A_{3,2}^1$ is a closed interval that does not include X^1. Then we always can find value $x^1 \in X^1$ s.t $A_{3,2}^1 \cap x^1 = \emptyset$. In other words, we can increase the bound $\underline{\ell}_{1-2}(y, \mathbf{X})$ by querying X^1.

However, as $A_{3,2}^2$ is a closed interval that includes X^2, there is no value $x^2 \in X^2$ s.t $A_{3,2}^2 \cap x^2 = \emptyset$, or in other words, the bound $\underline{\ell}_{1-2}(y, \mathbf{X})$ can not be increased by querying X^2.

Algorithms

Algorithm 12 summarizes how to determine the optimal query for a single querying step in case of imprecise features. It is very similar to Algorithm 7, but takes different sub-routines specific to the case of partially known features.

Algorithm 13 summarises how risk bounds, from which can be deduced the best potential model (through Algorithm 9, that remains unchanged), can be computed. Algorithms 14 and 15 describe how potential effects of querying an instance $(\mathbf{X}_n, y_n)$, respectively on empirical risk bounds and on pairwise risk bound, can be determined. Note that Algorithm 15 computes the sum of the pairwise effects between the best potential model θ_{k^*} and the other ones. Let us now look at the complexity of Algorithm 13, assuming that all decision trees have H leaves. Before doing that, note that checking whether two hyper-cubes do intersect is in $\mathcal{O}(P)$, according to Equation (3.53). Lines 2-3 are in $\mathcal{O}(S \times N \times H \times P)$, since Algorithm 13 is in $\mathcal{O}(N \times H \times P)$, as computing vector $\mathbf{K}$ (line 3 of Algorithm 13) is in $\mathcal{O}(H \times P)$. Algorithm 9 remains in $\mathcal{O}(S)$. Lines 4-9 of Algorithm 13 is in $\mathcal{O}(N \times S \times P \times H^4)$: indeed, in Algorithm 15, lines 7-9 are in $\mathcal{O}(P \times H^4)$, as we must apply Algorithm 11 to at most H^2 intervals.

In particular, Algorithm 15 treats both the cases of an instance that is imprecise with respect to both models, as well as the other cases (other loops): Line 2 determines whether the instance is imprecise w.r.t θ_{k^*}, Line 4 whether it is imprecise w.r.t θ_k. So Lines 4-9 corresponding to imprecision with respect to both models, lines 10-13 to imprecision w.r.t only θ_{k^*}, and lines 15-19 w.r.t only θ_k.

Algorithm 12: A single step to query interval-valued data.

Input: Training data set $\mathbf{D} = \{(\mathbf{X}_n, y_n)\}_{n=1}^N$,
label set $\mathcal{Y} = \{y_1, \ldots, y_M\}$, set of undominated model Θ^*.
Output: The optimal query q_{n^*}

1 **foreach** $\theta_k \in \Theta$ **do**
2 Compute empirical risk $[\underline{R}(\theta_k \,|\, \mathbf{D}), \overline{R}(\theta_k \,|\, \mathbf{D})]$ bounds using Alg. 13;
3 Determine the best model θ_{k^*} and the undominated model set Θ^* using Alg. 9;
4 **foreach** $n = 1, \ldots, N$ **do**
5 Determine $(E_{q_n^1}(\theta_{k^*}), \ldots, E_{q_n^P}(\theta_{k^*}))$ using Alg. 14 with model θ_{k^*};
6 Determine the cumulative pairwise effects $(J_{q_n^1}, \ldots, J_{q_n^P})$ using Alg. 15;
7 **foreach** $p = 1, \ldots, P$ **do**
8 $Value(q_n^p) = E_{q_n^p}(\theta_{k^*}) + J_{q_n^p}$;
9 Determine $(n^*, p^*) = \arg\max_{(n,p)} Value(q_n^p)$;

Algorithm 13: Compute the empirical risk bounds $(\underline{R}(\theta_k \,|\, \mathbf{D}), \overline{R}(\theta_k \,|\, \mathbf{D}))$.

Input: Training data set $\mathbf{D} = \{(\mathbf{X}_n, y_n)\}_{n=1}^N$,
label set $\mathcal{Y} = \{y_1, \ldots, y_M\}$, model θ_k.
Output: Empirical risk bounds $(\underline{R}(\theta_k \,|\, \mathbf{D}), \overline{R}(\theta_k \,|\, \mathbf{D}))$

1 $\underline{R}(\theta_k) = 0$, $\overline{R}(\theta_k) = 0$;
2 **foreach** $n = 1, \ldots, N$ **do**
3 Compute $\mathbf{K}$, $\mathbf{B}_{y_n}$ and $\mathbf{C}_{y_n}$ using (3.56)-(3.58);
4 **if** $\mathbf{K}\mathbf{C}_{y_n}^\top = 0$ **then** $\underline{R}(\theta_k \,|\, \mathbf{D}) = \underline{R}(\theta_k \,|\, \mathbf{D}) + 1$, $\overline{R}(\theta_k \,|\, \mathbf{D}) = \overline{R}(\theta_k \,|\, \mathbf{D}) + 1$;
5 **if** $\left(\mathbf{K}\mathbf{B}_{y_n}^\top\right)\left(\mathbf{K}\mathbf{C}_{y_n}^\top\right) \neq 0$ **then** $\overline{R}(\theta_k \,|\, \mathbf{D}) = \overline{R}(\theta_k \,|\, \mathbf{D}) + 1$;

Algorithm 14: Determine the single effects $(E_{q_n^1}(\theta_l), \ldots, E_{q_n^P}(\theta_l))$.

Input: Training instance $(\mathbf{X}_n, y_n)$, a model θ_l.
Output: The single effects $(E_{q_n^1}(\theta_l), \ldots, E_{q_n^P}(\theta_l))$

Initialize $(E_{q_n^1}, \ldots, E_{q_n^P}) = (0, \ldots, 0)$;
if $\underline{\ell}_l(y_n, \mathbf{X}_n) \neq \overline{\ell}_l(y_n, \mathbf{X}_n)$ **then**
  foreach $p = 1, \ldots, P$ *with* $\|X_n^p\| > 0$ **do**
    $In \leftarrow$ Alg. 11 with inputs X_n^p, $\{A_h^p | k^h c_{y_n}^h = 1\}$;
    if $In = 1$ **then** $E_{q_n^p}(\theta_l) = 1$;
    $In \leftarrow$ Alg. 11 with inputs X_n^p, $\{A_h^p | k_{y_n}^h b_{y_n}^h = 1\}$;
    if $In = 1$ **then** $E_{q_n^p}(\theta_l) = 1$;

Algorithm 15: Determine the cumulative pairwise effects $(J_{q_n^1}, \ldots, J_{q_n^P})$.

Input: Training instance $(\mathbf{X}_n, y_n)$, undominated model set Θ^*, best model θ_{k^*}.
Output: The cumulative pairwise effects $(J_{q_n^1}, \ldots, J_{q_n^P})$

Initialize $(J_{q_n^1}, \ldots, J_{q_n^P}) = (0, \ldots, 0)$;
if $\underline{\ell}_{k^*}(y_n, \mathbf{X}_n) \neq \overline{\ell}_{k^*}(y_n, \mathbf{X}_n)$ **then**
  foreach $\theta_k \in \Theta$ *and* $k \neq k^*$ **do**
    if $\underline{\ell}_k(y_n, \mathbf{X}_n)) \neq \overline{\ell}_k(y_n, \mathbf{X}_n)$ **then**
      Compute matrix $\mathbf{W}^{k,k^*}$ defined in (3.67);
      if $\min \mathbf{W}^{k,k^*} < 1$ **then**
        foreach $p = 1, \ldots, P$ *and* $\|X_n^p\| > 0$ **do**
          $In \leftarrow$ Alg. 11 with inputs X_n^p, $\{A_{i,j}^p : w_{i,j}^{k,k^*} = \min \mathbf{W}^{k,k^*}\}$;
          if $In = 1$ **then** $J_{q_n^p} = J_{q_n^p} + 1$;
      else
        foreach $p = 1, \ldots, P$ *and* $\|X_n^p\| > 0$ **do**
          $In \leftarrow$ Alg. 11 with inputs X_n^p, $\{A_h^p$ of $\theta_{k^*} | k_{y_n}^h b_{y_n}^h = 1\}$;
          if $In = 1$ **then** $J_{q_n^p} = J_{q_n^p} + 1$;
else
  foreach $\theta_k \in \Theta$ *and* $k \neq k^*$ **do**
    if $\underline{\ell}_k(y_n, \mathbf{X}_n) \neq \overline{\ell}_k(y_n, \mathbf{X}_n)$ **then**
      foreach $p = 1, \ldots, P$ *and* $\|X_n^p\| > 0$ **do**
        $In \leftarrow$ Alg. 11 with inputs X_n^p, $\{A_h^p$ of $\theta_k | k_{y_n}^h c_{y_n}^h = 1\}$;
        if $In = 1$ **then** $J_{q_n^p} = J_{q_n^p} + 1$;

Name	# instances	# features	# classes
wine	178	13	3
breast-cancer	569	30	2
vowel	990	10	11
segment	2310	19	7

TABLE 3.2: Data set used in the experiments

The overall complexity is polynomial in all parameters, which may be considered as reasonable when the number of partial data, and the complexity of the trees both remain limited. Also, this is a worst-case complexity, assuming that every feature of every training data is imprecise, and that every resulting hyper-cube intersect all leaves of all the decision trees in Θ. In practice, we may expect partial features to be quite less numerous, as well as their intersections with tree leaves.

It should also be noticed that since the models will not change during the race, and that data will only be queried iteratively, one can in principle compute all matrices at the start of the race, and then proceed to a minimal update at each query, thus considerably reducing the time to determine optimal queries. Finally, it should be noticed that querying data mainly makes sense when data are scarce (as an increased quantity of data improves the model accuracy even in the presence of imperfections).

3.4.3 Experimental evaluation

In this section, we run experiments on a "contaminated" version of 4 standard benchmark data sets as described in Table 3.2. To evaluate the efficiency of our proposal, we compare our racing algorithm with baseline algorithms whose details will be described separately in each setting of partial data. Note that when data are partial and, in contrast with classical active learning, it is usually difficult to divide the data between a set of training data and a set of data with missing values, especially if all data are partial. This is why we will do the queries on the same data we use to train the models. As the situation where both input and output are partially given rarely happens in practice, we only focus on two settings: partiality in inputs; and partiality in outputs. The next two subsections present details about the experimental settings and the results for interval-valued features and set-valued labels data, respectively.

Interval-valued features

We follow a 2×5 fold cross-validation procedure: Each data set is randomly split into 5 folds. Each fold is in turn considered as the training set $\mathbf{D}$, while other folds are used for testing $\mathbf{T}$. For each feature x_n^p in the training set, a biased coin is flipped in order to decide whether or not this example will be contaminated; the probability of contamination is ϵ. The level of partiality ϵ is fixed to two values (0.3 and 0.6) which correspond to a low and a high level of imprecision. Similarly to the SVM experimental parts, in case x_n^p is contaminated, a width η_n^p is generated from a uniform distribution on the unit interval and the generated interval valued data is $X_n^p = [x_n^p + \eta_n^p(\underline{D}^p - x_n^p), x_n^p + \eta_n^p(\overline{D}^p - x_n^p)]$ where $\underline{D}^p = \min_n(x_n^p)$ and $\overline{D}^p = \max_n(x_n^p)$.

Similar to the case of binary SVM, we generate an initial set of undominated models from 100 completions of interval-valued data. From each completion, one tree model (with a minimal number of training observations in any terminal node fixed to 3 for first two small data sets and 5 for the two later ones) is trained. The budget will be fixed to be the total number of partially featured values. After each query,

we discard the dominated models and determine the best potential model. In case of multiple minimum risk models, the one with a minimum value of $\overline{R}(\theta_k \mid \mathbf{D})$ will be chosen as the best potential model.

The two following baseline algorithms are employed to query interval-valued data and make comparison about the evolution of the size of the sets of undominated models and the performance of the best potential model:

- **a random querying strategy** where, at each iteration, the queried example and feature will be chosen randomly,

- and the **most partial querying strategy** designed such that, at each iteration, examples with the largest imprecision will be queried.

In practice, it may be the case that not all features appear in the set of racing trees. In those cases, keeping all the features in the instances would disadvantage both random and most partial querying in the race, since in this latter only the features present in the trees are relevant (i.e., will play a role to discard racing models). To make a comparable setting and to not give an unfair advantage to our method, we thus eliminate the features that do not appear in the trained trees.

In order to evaluate the performances of those different strategies, we will use three measures:

- the similarity of the best potential model θ_{k^*} with a reference model θ_{ref} is computed on the precise test set $\mathbf{T}$.

- the size of the undominated set Θ^*, that should decrease as fast as possible, both to ensure computational efficiency and model performances.

- the accuracy on the test set. The above criteria aim to assess the effect of querying strategies in the learning step. To evaluate the relevance of the queries on unseen data, we consider the queried data set after querying 5% of the partial values, and this up to 30% (so, we test our queried data set for 5%, 10%, 15%, ... queries). Since some partial data remain, we first impute those ones (replacing X_n^p by their middle values), learn a model θ^* on the obtained fully precise training set, and evaluate its accuracy on the test set.

The 5-folds process is repeated 2 times and the average size of the sets of models, the average similarity of the best potential model and the average acuracy on the test set are reported.

The experimental results are presented in Figure 3.13 to 3.15. They show that, using the racing approach, the size of the undominated set can be quickly reduced and that the best potential model converges very fast to the desired model when knowing a small number of the precise data. The reduction of the size of the set is much slower for other querying strategies. This is true for the four tested data sets, and the advantage of using the racing approach is obvious whether we have little ($\epsilon = 0.3$) or a lot ($\epsilon = 0.6$) of imprecision. The exception observed for high imprecision ($\epsilon = 0.6$) in the case of the segment data set is due to the fact that few features are used in the different trees, hence all models are quite similar, and all querying strategies focus on those features, converging at comparable speeds. Regarding the interest of the queries on the final learnt model, we can see that the racing approach provides better improvements in most cases, in particular when the accuracy difference before and after querying is significant.

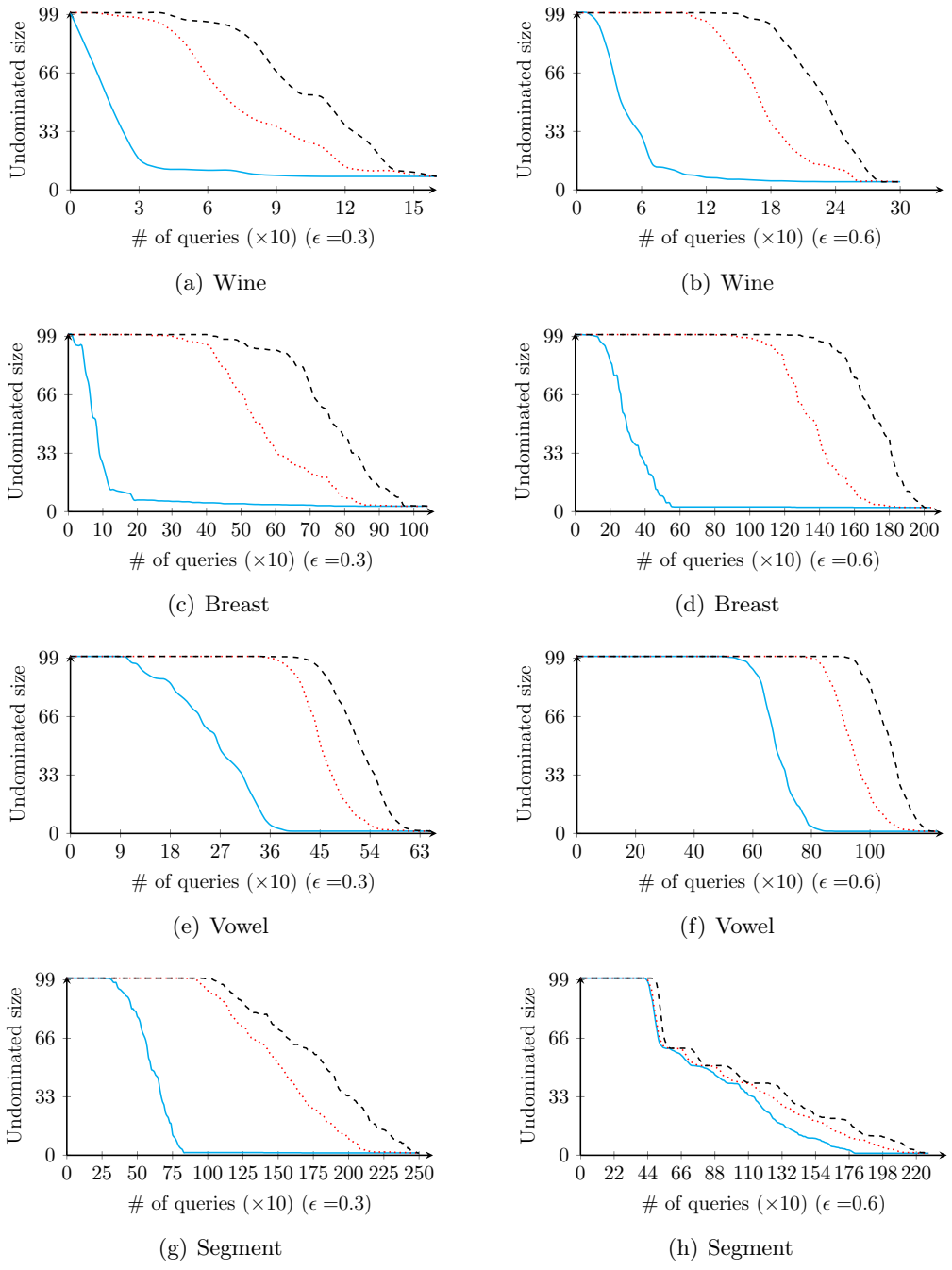
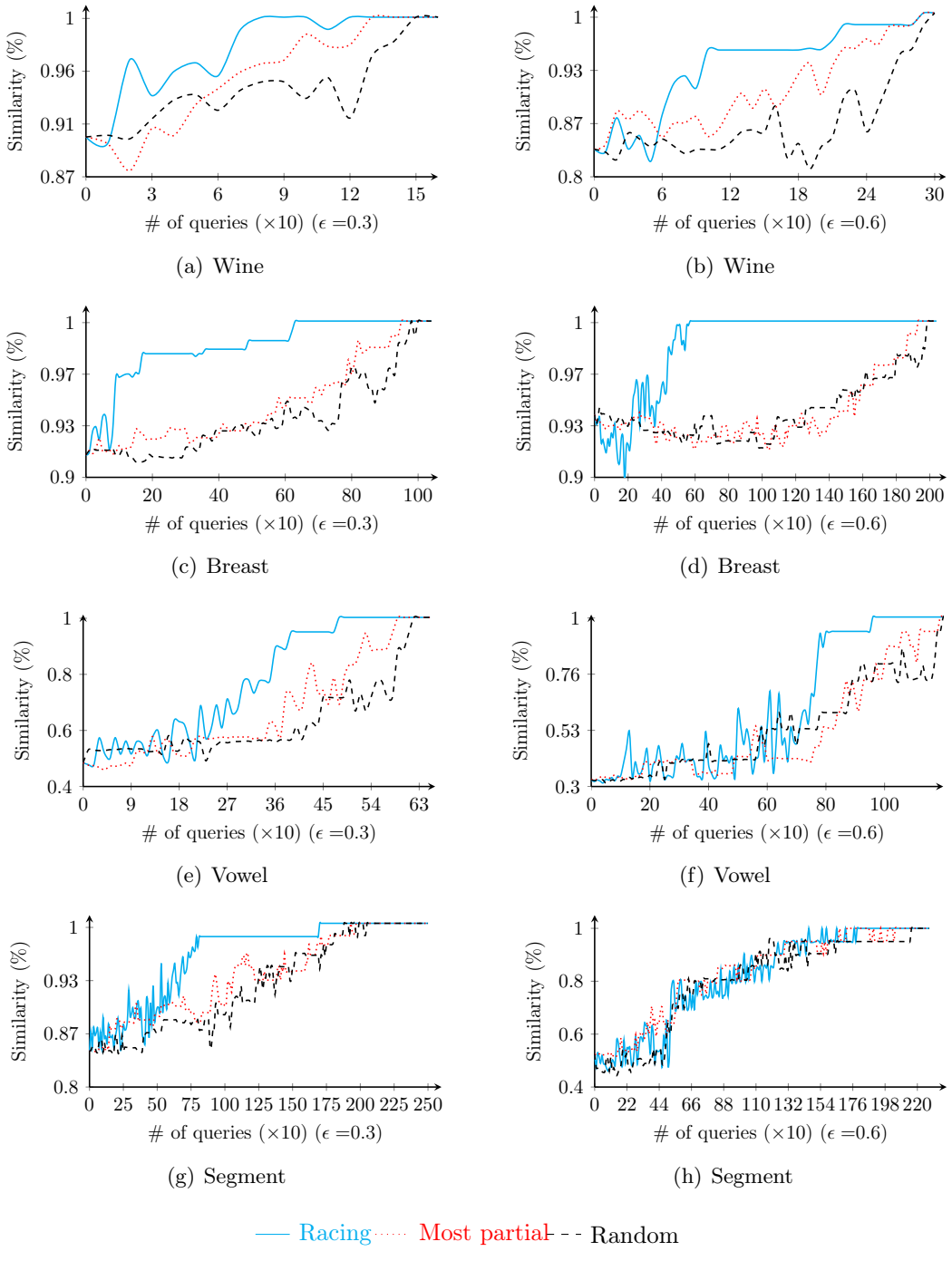


FIGURE 3.13: Interval-valued features: Size of undominated model sets



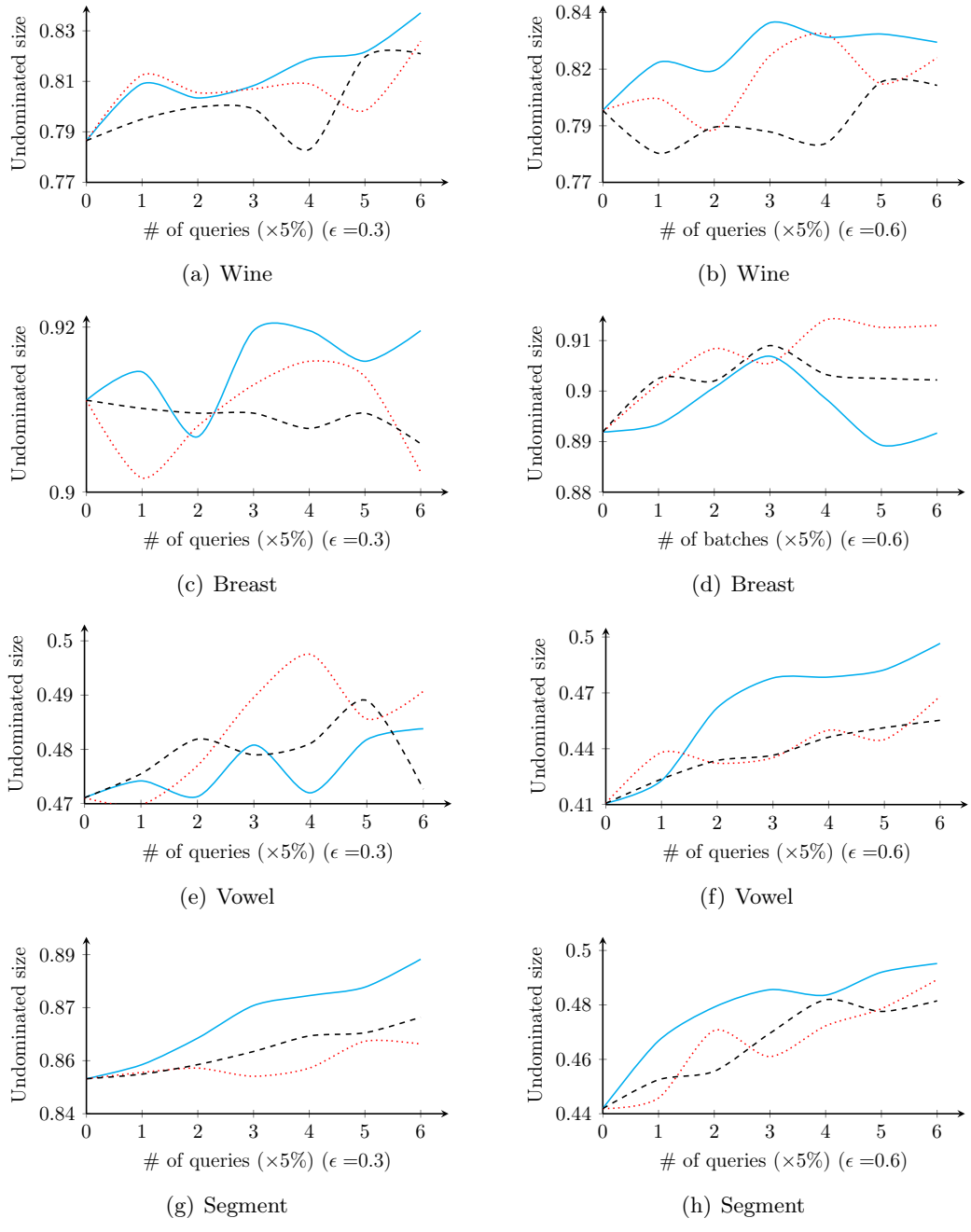
(a) Wine (b) Wine

(c) Breast (d) Breast

(e) Vowel (f) Vowel

(g) Segment (h) Segment

FIGURE 3.14: Interval-valued features: Similarity between the current best and reference models



(a) Wine (b) Wine

(c) Breast (d) Breast

(e) Vowel (f) Vowel

(g) Segment (h) Segment

FIGURE 3.15: Interval-valued features: Accuracy on the test set

Set-valued labels

We perform on the same data sets as before (cf. Table 3.2) and the 2×5 cross-validation procedure as described for partially featured data (without the feature filtering step as we only consider the partial labels here). In order to contaminate a given data set, we used the following strategy: for each example in the training set, a biased coin is flipped in order to decide whether or not this example will be contaminated; the probability of contamination is ϵ. When an example is contaminated, the class candidates are added with probability η, independently of each other. Thus, the contamination procedure is parametrized by the probabilities ϵ and η, where ϵ corresponds to the expected fraction of imprecise examples in a data set, and η reflects the average number of classes added to contaminated examples. The expected cardinality of a label set, in case of contamination, is given by $1 + (M - 1)\eta$. In all experiments, ϵ and η are fixed respectively to 0.3 and 0.8. To start the race, 100 precise replacements for each imprecise labels are randomly chosen. From each selection, one classification tree is trained. Similarly to the case of partial features, the minimal number of observations in any terminal node is fixed to 3 and 5 for the first two data sets and the later ones, respectively.

Similar to the case of binary SVM, we compare our racing approach with two baseline querying schemes: **a random query** and **a query by committee** approach (QBC). Finally, the size of the sets of models and the similarity of the best potential model θ_{k^*} w.r.t. the reference model θ_{ref} are reported and used to make comparison. Since in the case of partial labels there is almost no difference between the approaches, we did not evaluate their performances on test sets.

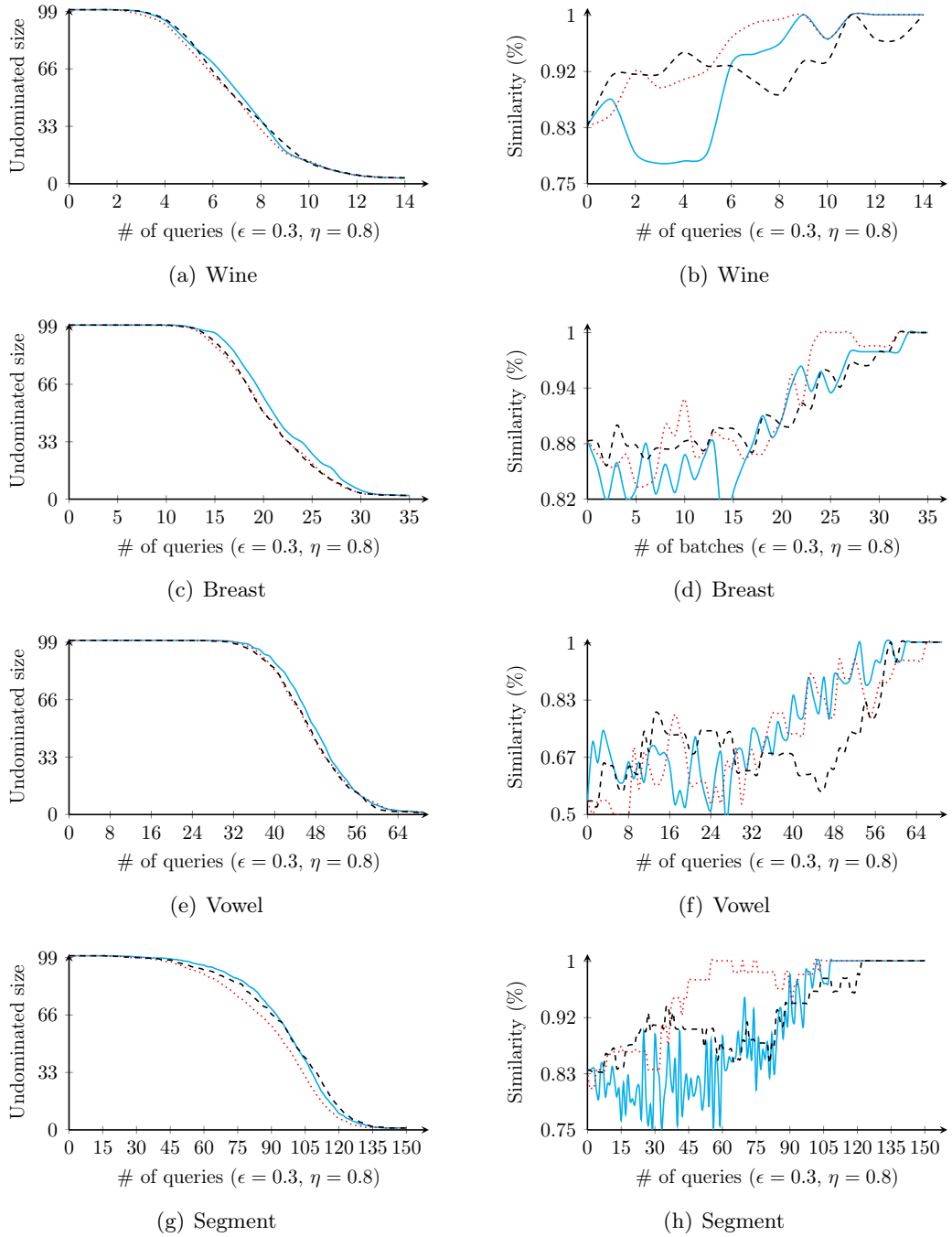
The experimental results, presented in Figure 3.16, show that, among the three approaches, random queries usually converge more slowly towards the reference model (except for the vowel data set), while the set of undominated models decreases similarly for all data sets and all strategies (with a slight advantage for the QBC strategy, and a poorly performing random queries for the segment data set). This contrasts with the partial feature case, where our approach significantly outperforms the others. A reason for that maybe that the case of partial labels offers much less degrees of freedom, hence the impact of the querying strategy may be quite less important than for the feature case.

3.5 Conclusion

The problem of actively learning with partial data has been little explored in the literature, in particular the case of partially known features. Indeed, active learning techniques usually focus on the case where a part of the labels are completely missing, while a few are precisely known. To solve the problem, we have proposed in this Chapter a generic querying approach based on the idea of racing algorithms. Our generic approach has been then detailed for the specific cases of binary SVM and decision trees. To do so, we have developed a number of efficient algorithms to detect which data should be queried, in order to identify as soon as possible the best model among a set of racing ones.

We have then made some experiments to study the behaviour of our approach, compared to other querying strategies, starting from the same set of initial models. Our conclusion is that our approach significantly outperforms simpler strategies in the case of partially specified features, while it achieves similar performances in the case of partially specified labels. We think that this is due to the fact that partial labels offer much less degrees of freedom to the learning algorithms, meaning that

most smart strategies, or even random ones will perform similarly. This is not the case for partial features, where purely random strategies performs poorly.



(a) Wine (b) Wine

(c) Breast (d) Breast

(e) Vowel (f) Vowel

(g) Segment (h) Segment

FIGURE 3.16: Experiments for set-valued label data with preferred model

Chapter 4

Epistemic uncertainty for active learning and cautious inferences

As mentioned in the introduction, we think that differentiating sources of uncertainty should benefit to machine learning applications. For instances, it could be useful for developing querying criteria when doing active learning or balancing the informativeness and cautiousness when making cautious inferences. This research direction has been well-studied in the literature on uncertainty and machine learning. We will restrict ourselves, in this chapter, to a distinction between two sources of uncertainty: *epistemic*, caused by a lack of training data and, and *aleatoric*, due to intrinsic randomness. After summarizing the basic concepts and presenting the practical procedures to estimate these degrees of uncertainty, we will explain how these estimates can be used to solve two machine learning problems: active learning and cautious inferences.

4.1 Likelihood to estimate epistemic and aleatoric uncertainties

We are going to provide a quick literature review on this line of research and then recall the basics of a contour-likelihood based approach which will be adopted in the later proposals. To facilitate subsequent applications, we will propose practical procedures to estimate the degrees of uncertainty for popular classifiers.

4.1.1 A formal framework for uncertainty modeling

Epistemic uncertainty in learning theory

As already said, the problem of differentiating sources of uncertainty has been increasingly investigated [42, 51, 53, 79, 85]. In this line of research, several approaches exist and have been successfully implemented for different applications, including classification [53, 79] and active learning [85]. We are going to quickly mention approaches related to our interests as well as their subsequent applications.

- Sharma and Bilgic [85] recently proposed an evidence-based approach to active learning, in which *conflicting-evidence* uncertainty is distinguished from *insufficient-evidence* uncertainty. Roughly speaking, a high conflicting evidence captures the case where the evidences are close and both of large magnitude. In other words, it refers to the situation where a model is highly uncertain about an instance, and has strong but conflicting evidence for both classes. On the other hand, a high insufficient-evidence uncertainty refers to the case where a model is highly uncertain about an instance because of not having enough evidence for either class. Experimentally, they support their conjecture that the

conflicting-evidence uncertainty is more informative for an active learner than the *conflicting-evidence* one.

- Conformal prediction [4, 84] is a generic approach to reliable (set-valued) prediction that combines ideas from probability theory (specifically the principle of exchangeability), statistics (hypothesis testing, order statistics), and algorithmic complexity. The basic version of conformal prediction is designed for sequential prediction in an online setting, and comes with certain correctness guarantees (predictions are correct with probability $1-\xi$, where ξ is a confidence parameter). Roughly speaking, given an instance $\boldsymbol{t}$, it assigns a *non-conformity* score to each candidate output. Then, considering each of these outcomes as a hypothesis, those outcomes for which the hypothesis can be rejected with high confidence are eliminated. The set-valued prediction is given by the set of those outcomes that cannot be rejected.

- Cautious (set-valued) prediction methods based on imprecise probabilities, such as [15], augment the probabilistic predictions into probability intervals or sets of probabilities, the size of which reflects the lack of information (reflecting epistemic uncertainty). Similar to this are approaches based on confidence bands in calibration models, for instance [53, 99], which usually control the amount of imprecision by adjusting some certain parameters, e.g., a confidence value.

- Credal uncertainty sampling [3] is another approach that seeks to differentiate between parts of the uncertainty. This approach assumes that a *credal set* $C \subseteq \Theta$ is given and learns for each label $y \in \mathcal{Y}$, an interval-valued probability $[\underline{p}_C(y \,|\, \boldsymbol{t}), \overline{p}_C(y \,|\, \boldsymbol{t})]$. In this case, $\underline{p}_C(y \,|\, \boldsymbol{t})$ and $\overline{p}_C(y \,|\, \boldsymbol{t})$ are the minimum and maximum conditional probabilities that can be given for y by candidates of C, respectively. The widths of the interval-valued probabilities reflect the *reducible* part of uncertainty while its extreme probabilities reflect the *irreducible* one. More precisely, we can shrink the interval $[\underline{p}_C(y \,|\, \boldsymbol{t}), \overline{p}_C(y \,|\, \boldsymbol{t})]$ (or, reduce the epistemic part) by acquiring additional training data, eventually getting a precise value. We will only illustrate how the extreme probabilities reflect the *irreducible* part (of uncertainty) in the case of binary classification, i.e, $\mathcal{Y} := \{0, 1\}$ (the case of multi-class classification could be rather complicated and will not be investigated here). In this case, we could image that if the interval probabilities $[\underline{p}_C(y \,|\, \boldsymbol{t}), \overline{p}_C(y \,|\, \boldsymbol{t})]$ is symmetrical around 0.5, shrinking such intervals is not very helpful in distinguishing the classes (i.e, a high irreducible uncertainty), especially when its extreme probabilities are close to 0.5. However, if $[\underline{p}_C(y \,|\, \boldsymbol{t}), \overline{p}_C(y \,|\, \boldsymbol{t})]$ is highly asymmetrical around 0.5, shrinking it should benefit in distinguishing the classes (i.e, a low irreducible uncertainty).

- The distinction between the *epistemic* and *aleatoric* uncertainty involved in the prediction for an instance $\boldsymbol{t}$ is well-accepted in the literature on uncertainty [42, 79] and has been considered in only few recently machine learning works, e.g, [51]. Roughly speaking, the *aleatoric* uncertainty refers to the notion of randomness, that is, the variability in the outcome of an experiment which is due to inherently random effects. As opposed to this, the *epistemic* uncertainty refers to the uncertainty caused by a lack of knowledge. Thus, the distinction considered here appears to be quite related to the one between *conflicting-evidence* and *insufficient-evidence* uncertainty [85], and the one considered in credal uncertainty sampling [3]. Detailed comparisons will be given in our proposal for active learning.

Yet, the concepts of the degrees of epistemic and aleatoric uncertainty are, in principle, defined in the literature. Practically quantifying these degrees for different classifiers still remains a challenge. We are going to summarize the contour-likelihood based approach [79] and then detail it for the local model, logistic regression and Naive Bayes classifiers.

Contour-likelihood based approach

In the rest of this chapter, we adopt the contour-likelihood based approach proposed by Senge et al. [79], which is based on the use of relative likelihoods, historically proposed by Birnbaum [6] and then justified in other settings such as possibility theory [94]. In the following, the essence of this approach is briefly recalled.

We proceed from an instance space $\mathcal{X} = \mathbb{R}^P$, an output space $\mathcal{Y} = \{0, 1\}$ encoding the two classes, and a hypothesis space Θ consisting of probabilistic classifiers $\theta : \mathcal{X} \longrightarrow [0, 1]$. We denote by $p_\theta(1 \,|\, \boldsymbol{x}) = \theta(\boldsymbol{x})$ and $p_\theta(0 \,|\, \boldsymbol{x}) = 1 - \theta(\boldsymbol{x})$ the (predicted) probability that instance $\boldsymbol{x} \in \mathcal{X}$ belongs to the positive and negative class, respectively. Given a set of training data $\mathbf{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, the contour likelihood of a model θ is defined as

$$\pi_\Theta(\theta \,|\, \mathbf{D}) = \frac{L(\theta \,|\, \mathbf{D})}{L(\theta^* \,|\, \mathbf{D})} = \frac{L(\theta \,|\, \mathbf{D})}{\max_{\theta' \in \Theta} L(\theta' \,|\, \mathbf{D})} \;, \tag{4.1}$$

where $L(\theta \,|\, \mathbf{D}) = \prod_{n=1}^N p_\theta(y_n \,|\, \boldsymbol{x}_n)$ is the likelihood of θ, and $\theta^* \in \Theta$ the maximum likelihood estimation on the training data $\mathbf{D}$. For a given instance $\boldsymbol{t}$, the degrees of support (plausibility) of the two classes are defined as follows:

$$\pi(1 \,|\, \boldsymbol{t}) = \sup_{\theta \in \Theta} \min \big[\pi_\Theta(\theta \,|\, \mathbf{D}), p_\theta(1 \,|\, \boldsymbol{t}) - p_\theta(0 \,|\, \boldsymbol{t})\big], \tag{4.2}$$

$$\pi(0 \,|\, \boldsymbol{t}) = \sup_{\theta \in \Theta} \min \big[\pi_\Theta(\theta \,|\, \mathbf{D}), p_\theta(0 \,|\, \boldsymbol{t}) - p_\theta(1 \,|\, \boldsymbol{t})\big]. \tag{4.3}$$

So, $\pi(1 \,|\, \boldsymbol{t})$ is high if and only if a highly plausible model supports the positive class much stronger (in terms of the assigned probability mass) than the negative class (and $\pi(0 \,|\, \boldsymbol{t})$ can be interpreted analogously)[1]. Note that, with $f(a) = 2a - 1$, we can also rewrite (4.2)-(4.3) as follows:

$$\pi(1 \,|\, \boldsymbol{t}) = \sup_{\theta \in \Theta} \min \big[\pi_\Theta(\theta \,|\, \mathbf{D}), f(\theta(\boldsymbol{t}))\big], \tag{4.4}$$

$$\pi(0 \,|\, \boldsymbol{t}) = \sup_{\theta \in \Theta} \min \big[\pi_\Theta(\theta \,|\, \mathbf{D}), f(1 - \theta(\boldsymbol{t}))\big]. \tag{4.5}$$

Given the above degrees of support, the degrees of epistemic uncertainty u_e and aleatoric uncertainty u_a are defined as follows [79]:

$$u_e(\boldsymbol{t}) = \min \big[\pi(1 \,|\, \boldsymbol{t}), \pi(0 \,|\, \boldsymbol{t})\big] \,, \tag{4.6}$$

$$u_a(\boldsymbol{t}) = 1 - \max \big[\pi(1 \,|\, \boldsymbol{t}), \pi(0 \,|\, \boldsymbol{t})\big] \,. \tag{4.7}$$

Thus, the epistemic uncertainty refers to the case where both the positive and the negative class appear to be plausible, while the degree of aleatoric uncertainty (4.7) is the degree to which none of the classes is supported. These uncertainty degrees are completed with degrees $s_1(\boldsymbol{t})$ and $s_0(\boldsymbol{t})$ of (strict) preference in favor of the positive

[1] Technically, we assume that, for each $\boldsymbol{t} \in \mathcal{X}$, there are hypotheses $\theta, \theta' \in \Theta$ such that $\theta(\boldsymbol{t}) \geq 0.5$ and $\theta'(\boldsymbol{t}) \leq 0.5$, which implies $\pi(1 \,|\, \boldsymbol{t}) \geq 0$ and $\pi(0 \,|\, \boldsymbol{t}) \geq 0$.

and negative class, respectively:

$$s_1(\boldsymbol{t}) = \begin{cases} 1 - (u_a(\boldsymbol{t}) + u_e(\boldsymbol{t})) & \text{if } \pi(1 \,|\, \boldsymbol{t}) > \pi(0 \,|\, \boldsymbol{t}), \\ \frac{1-(u_a(\boldsymbol{t})+u_e(\boldsymbol{t}))}{2} & \text{if } \pi(1 \,|\, \boldsymbol{t}) = \pi(0 \,|\, \boldsymbol{t}), \\ 0 & \text{if } \pi(1 \,|\, \boldsymbol{t}) < \pi(0 \,|\, \boldsymbol{t}). \end{cases}$$

With an analogous definition for $s_0(\boldsymbol{t})$, we have

$$s_0(\boldsymbol{t}) + s_1(\boldsymbol{t}) + u_a(\boldsymbol{t}) + u_e(\boldsymbol{t}) \equiv 1\,,$$

i.e., the quadruple $(s_1(\boldsymbol{t}), s_0(\boldsymbol{t}), u_e(\boldsymbol{t}), u_a(\boldsymbol{t}))$ defines a partition of unity. Besides, it has the following properties:

- $s_1(\boldsymbol{t})$ ($s_0(\boldsymbol{t})$) will be high if and only if, for all plausible models, the probability of the positive (negative) class is significantly higher than the one of the negative (positive) class;
- $u_e(\boldsymbol{t})$ will be high if the class probabilities strongly vary within the set of plausible models, i.e., if we are unsure how to compare these probabilities. In particular, it will be 1 if and only if we have $\theta(\boldsymbol{t}) = 1$ and $\theta'(\boldsymbol{t}) = 0$ for two totally plausible models θ and θ';
- $u_a(\boldsymbol{t})$ will be high if the class probabilities are similar for all plausible models, i.e., if there is strong evidence that $\theta(\boldsymbol{t}) \approx 0.5$. In particular, it will be close to 1 if all plausible models allocate their probability mass around $\theta(\boldsymbol{t}) = 0.5$.

Roughly speaking, the aleatoric uncertainty is due to influences on the data-generating process that are inherently random, whereas the epistemic uncertainty is caused by a lack of knowledge. Or, stated differently, u_e and u_a measure the *reducible* and the *irreducible* part of the total uncertainty, respectively.

As said in [79], determining the degrees of support (4.4)-(4.5) comes down to solving optimization problems, the complexities of which strongly depend on the model space Θ, and may become rather complex. We are going to summarize the details for the case of Parzen window classifiers, whose details is given in [79] and presents our proposals for the cases of logistic regression and Naive Bayes.

4.1.2 Estimation for local models

By local learning, we refer to a class of non-parametric models that derive predictions from the training information in a local region of the instance space, for example the local neighborhood of a query instance [8, 20]. As a simple example, we consider the Parzen window classifier [11], to which our approach can be applied in a quite straightforward way. To this end, for a given instance $\boldsymbol{t}$, we define the set of its neighbours as follows:

$$R(\boldsymbol{t}, \delta) = \left\{ (\boldsymbol{x}_n, y_n) \in \mathbf{D} \,|\, \|\boldsymbol{x}_n - \boldsymbol{t}\| \leq \delta \right\}, \tag{4.8}$$

where δ is the width of the Parzen window (a practical method to determine such a width will be given latter).

In binary classification, a local region R can be associated with a constant hypothesis $\theta \in \Theta = [0, 1]$, where $\theta(\boldsymbol{t})$ is the probability of the positive class in the region; thus, θ predicts the same probabilities $\theta(1 \,|\, \boldsymbol{t}) = \theta$ and $\theta(0 \,|\, \boldsymbol{t}) = 1 - \theta$ for all $\boldsymbol{t} \in R$. The underlying hypothesis space is given by $\Theta = \{\theta \,|\, 0 \leq \theta \leq 1\}$.

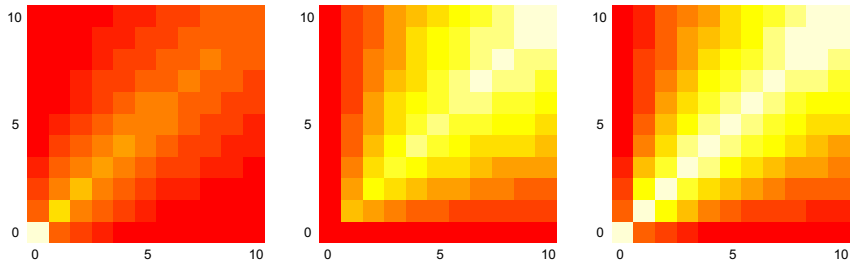


FIGURE 4.1: From left to right: Epistemic, aleatoric, and the total of epistemic aleatoric uncertainty as a function of the numbers of positive (x-axis) and negative (y-axis) examples in a region (Parzen window) of the instance space (lighter colors indicate higher values).

With α and β the number of positive and negative instances, respectively, within a Parzen window $R(\boldsymbol{t}, \delta)$, the likelihood is then given by

$$L_{\boldsymbol{t}}(\theta \,|\, \mathbf{D}) = \begin{pmatrix} \alpha+\beta \\ \beta \end{pmatrix} \theta^{\alpha}(1-\theta)^{\beta} \,, \tag{4.9}$$

and the maximum likelihood estimate is

$$\theta^{*} = \frac{\alpha}{\alpha+\beta} \,. \tag{4.10}$$

Therefore, the degrees of support for the positive and negative classes are

$$\pi(1|\boldsymbol{t}) = \sup_{\theta\in[0,1]} \min \left(\frac{\theta^{\alpha}(1-\theta)^{\beta}}{\left(\frac{\alpha}{\alpha+\beta}\right)^{\alpha}\left(\frac{\beta}{\alpha+\beta}\right)^{\beta}}, 2\theta-1 \right) \,, \tag{4.11}$$

$$\pi(0|\boldsymbol{t}) = \sup_{\theta\in[0,1]} \min \left(\frac{\theta^{\alpha}(1-\theta)^{\beta}}{\left(\frac{\alpha}{\alpha+\beta}\right)^{\alpha}\left(\frac{\beta}{\alpha+\beta}\right)^{\beta}}, 1-2\theta \right) \,. \tag{4.12}$$

Solving (4.11) and (4.12) comes down to maximizing a scalar function over a bounded domain, for which standard solvers can be used. We applied Brent's method[2] (which is a variant of the golden section method) to find a local minimum in the interval $\theta \in [0,1]$. From (4.11–4.12), the epistemic and aleatoric uncertainties associated with the region R can be derived according to (4.55) and (4.56), respectively. For different combinations of α and β, these uncertainty degrees can be pre-computed (cf. Figure 4.2).

How to determine the width δ of the Parzen window? This value is difficult to assess, and an appropriate choice strongly depends properties of the data and the dimensionality of the instance space. Intuitively, it is even difficult to say in which range this value should lie. Therefore, instead of fixing δ, we fixed an absolute number K of neighbors in the training data, which is intuitively more meaningful and easier to interpret. A corresponding value of δ is then determined in such a way that the average number of nearest neighbours of instances $\boldsymbol{x}_n$ in the training data $\mathbf{D}$ is just

[2]For an implementation in Python, see https://docs.scipy.org/doc/scipy-0.19.1/reference/generated/scipy.optimize.minimize_scalar.html

K (see Algorithm 16). In other words, δ is determined indirectly via K.

Since K is an average, individual instances may have more or less neighbors in their Parzen windows. In particular, a Parzen window may also be empty. In this case, we set $u_e(\boldsymbol{t}) = 1$ by definition, i.e., we consider this as a case of full epistemic uncertainty. Likewise, the uncertainty is considered to be maximal for all other sampling techniques. If the accuracy of the Parzen classifier needs to be determined, we assume that it yields an incorrect prediction.

Algorithm 16: Determining the width δ.

Input: **D**-normalized data, K-number
Output: the local width δ_K

foreach $\boldsymbol{x}_n \in \mathbf{D}$ **do**
  foreach $\boldsymbol{x}_m \neq \boldsymbol{x}_n$ **do**
    compute $d\big(\boldsymbol{x}_n, \boldsymbol{x}_m\big)$;
  form $1 \times (n-1)$ vector $\mathbf{d}_n = \big(d\big(\boldsymbol{x}_n, \boldsymbol{x}_m\big) \,|\, n \neq m\big)$;
  sort $\mathbf{d}_n$ by increasing order and determine the K-th element $\mathbf{d}_n^K$;
return $\delta_K = \frac{\sum_{n=1}^{|\mathbf{D}|} \mathbf{d}_n^K}{|\mathbf{D}|}$;

4.1.3 Estimation for logistic regression

Recall that logistic regression assumes posterior probabilities to depend on feature vectors $\boldsymbol{x} = (x^1, \ldots, x^P) \in \mathbb{R}^P$ in the following way:

$$\theta(\boldsymbol{x}) = p_\theta(1 \,|\, \boldsymbol{x}) = \frac{\exp\left(\theta^0 + \sum_{p=1}^P \theta^p \, x^p\right)}{1 + \exp\left(\theta^0 + \sum_{p=1}^P \theta^p \, x^p\right)} \tag{4.13}$$

This means that learning the model comes down to estimating a parameter vector $\theta = (\theta^0, \ldots, \theta^P)$, which is commonly done through likelihood maximization [62]. To avoid numerical issues (e.g, having to deal with the exponential function for large θ) when maximizing the target function, we employ L_2-regularization. The corresponding version of the log-likelihood function (4.14) is strictly concave [75]:

$$\begin{aligned} l(\theta \,|\, \mathbf{D}) = \log L(\theta \,|\, \mathbf{D}) = &\sum_{n=1}^N y_n \left(\theta^0 + \sum_{p=1}^P \theta^p x_n^p\right) \\ &- \sum_{n=1}^N \ln\left(1 + \exp\left(\theta_0 + \sum_{p=1}^P \theta^p x_n^p\right)\right) - \frac{\gamma}{2} \sum_{p=0}^P (\theta^p)^2, \end{aligned} \tag{4.14}$$

where the regularization parameter γ will be fixed to 1.

We now focus on determining the degree of support (4.4) for the positive class, and then summarize the results for the negative class (which can be determined in a similar manner). Associating each hypothesis $\theta \in \Theta$ with a vector $\theta \in \mathbb{R}^{P+1}$, the degree of support (4.4) can be rewritten as follows:

$$\pi(1 \,|\, \boldsymbol{t}) = \sup_{\theta \in \mathbb{R}^{d+1}} \min\big[\pi_\Theta(\theta \,|\, \mathbf{D}), 2\theta(\boldsymbol{t}) - 1\big] \tag{4.15}$$

It is easy to see that the target function to be maximized in (4.15) is not necessarily concave. Therefore, we propose the following approach.

Let us first note that whenever $\theta(\boldsymbol{t}) < 0.5$, we have

$$2\theta(\boldsymbol{t}) - 1 \leq 0 \text{ and } \min\left[\pi_\Theta(\theta \,|\, \mathbf{D}), 2\theta(\boldsymbol{t}) - 1\right] \leq 0.$$

Thus the optimal value of the target function (4.15) can only be achieved for some hypotheses θ such that $\theta(\boldsymbol{t}) \in [0.5, 1]$.

For a given value $\alpha \in [0.5, 1)$, the set of hypotheses θ such that $\theta(\boldsymbol{t}) = \alpha$ corresponds to the convex set

$$\theta^\alpha = \left\{ \theta \,\middle|\, \theta^0 + \sum_{p=1}^{P} \theta^p x^p = \ln\left(\frac{\alpha}{1-\alpha}\right) \right\}. \tag{4.16}$$

The optimal value $\pi^*_\alpha(1 \,|\, \boldsymbol{t})$ that can be achieved within the region (4.16) can be determined as follows:

$$\pi^*_\alpha(1 \,|\, \boldsymbol{t}) = \sup_{\theta \in \theta^\alpha} \min\left[\pi_\Theta(\theta \,|\, \mathbf{D}), 2\alpha - 1\right] = \min\left[\sup_{\theta \in \theta^\alpha} \pi_\Theta(\theta \,|\, \mathbf{D}), 2\alpha - 1\right]. \tag{4.17}$$

Thus, to find this value, we maximize the concave log-likelihood over a convex set:

$$\theta^*_\alpha = \arg\sup_{\theta \in \theta^\alpha} l(\theta \,|\, \mathbf{D}) \tag{4.18}$$

As the log-likelihood function (4.14) is concave and has second-order derivatives, we tackle the problem with a Newton-CG algorithm [68]. Furthermore, the optimization problem (4.18) can be solved using sequential least squares programming[3] [72]. Since regions defined in (4.16) are parallel hyperplanes, the solution of the optimization problem (4.15) can then be obtained by solving the following problem:

$$\sup_{\alpha \in [0.5,1)} \pi^*_\alpha(1|\boldsymbol{x}) = \sup_{\alpha \in [0.5,1)} \min\left[\pi_\Theta(\theta^*_\alpha \,|\, \mathbf{D}), 2\alpha - 1\right] \tag{4.19}$$

Following a similar procedure, we can estimate the degree of support for the negative class (4.5) as follows:

$$\sup_{\alpha \in (0,0.5]} \pi^*_\alpha(0|\boldsymbol{x}) = \sup_{\alpha \in (0,0.5]} \min\left[\pi_\Theta(\theta^*_\alpha \,|\, \mathbf{D}), 1 - 2\alpha\right] \tag{4.20}$$

Note that limit cases $\alpha = 1$ and $\alpha = 0$ cannot be solved, since the region (4.16) is then not well-defined (as $\ln(\infty)$ and $\ln(0)$ do not exist). For the purpose of practical implementation, we handle (4.19) by discretizing the interval over α. That is, we optimize the target function for a given number of values $\alpha \in [0.5, 1)$ and consider the solution corresponding to the α with the highest optimal value of the target function $\pi^*_\alpha(1 \,|\, \boldsymbol{t})$ as the maximum estimator. Similarly, (4.20) can be handled over the domain $(0, 0.5]$.

In practice, we evaluate (4.19) and (4.20) on uniform discretizations of cardinality 50 of $[0.5, 1)$ and $(0, 0.5]$, respectively. We can further increase efficiency by avoiding computations for values of α for which we know that $2\alpha - 1$ and $1 - 2\alpha$ are lower than the current highest support value given to class 1 and 0, respectively. See Algorithm 17 for a pseudo-code description of the whole procedure.

[3] For an implementation in Python, see https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.minimize.html

Algorithm 17: Degrees of support for logistic regression

Input: Q, $\mathbf{D}$, θ^*, $\boldsymbol{t}$- initial pool, training data, classifier, unlabelled instance
Output: $\pi(1\,|\,\boldsymbol{t})$, $\pi(0\,|\,\boldsymbol{t})$ - degrees of support

initialize subsets Q_1, Q_0 of cardinality Q;
$\pi(1\,|\,\boldsymbol{t}) = \max(2\theta^*(\boldsymbol{t}) - 1, 0)$, $\pi(0\,|\,\boldsymbol{t}) = \max(1 - 2\theta^*(\boldsymbol{x}), 0)$;
for $q = 1, \ldots, Q$ **do**
  $\alpha_1 = \max(Q_1)$; $\alpha_0 = \min(Q_0)$;
  if $2\alpha_1 - 1 > \pi(1\,|\,\boldsymbol{t})$ **then**
    solve (4.18) for $\boldsymbol{t}$, α_1 and return θ;
    $\pi(1\,|\,\boldsymbol{t}) = \max(\pi(1\,|\,\boldsymbol{t}), \min(\pi_\Theta(\theta\,|\,\mathbf{D}), 2\alpha_1 - 1))$;
  if $1 - 2\alpha_0 > \pi(0\,|\,\boldsymbol{t})$ **then**
    solve (4.18) for $\boldsymbol{t}$, α_0 and return θ;
    $\pi(0\,|\,\boldsymbol{t}) = \max(\pi(0\,|\,\boldsymbol{t}), \min(\pi_\Theta(\theta\,|\,\mathbf{D}), 1 - 2\alpha_0))$;
  $Q_1 = Q_1 \setminus \{\alpha_1\}$, $Q_0 = Q_0 \setminus \{\alpha_0\}$;
Return $\pi(1\,|\,\boldsymbol{t})$, $\pi(0\,|\,\boldsymbol{t})$;

4.1.4 Estimation for Naive Bayes

Let us first remind that we have been working on a training data set $\mathbf{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, where $\boldsymbol{x}_n = (x_n^1, \ldots, x_n^P)$ s.t $x_n^p \in \mathcal{X}^p = \{a_1^p, \ldots, a_{T_p}^p\}$, for $p = 1, \ldots, P$. For each feature $\mathcal{X}^p$, denoting by

$$\begin{aligned} \theta_1^{p,t_p} &= p_\theta(\mathcal{X}^p = a_{t_p}^p | Y = 1) \\ \theta_0^{p,t_p} &= p_\theta(\mathcal{X}^p = a_{t_p}^p | Y = 0), \end{aligned} \tag{4.21}$$

we then have that

$$\begin{aligned} &\sum_{t_p=1}^{T_p} \theta_1^{p,t_p} = \sum_{t_p=1}^{T_p} \theta_0^{p,t_p} = 1, \forall p = 1, \ldots, P, \\ &\theta_1^{p,t_p}, \theta_0^{p,t_p} \in [0,1], \forall t_p = 1 \ldots T_p, p = 1, \ldots, P. \end{aligned} \tag{4.22}$$

Furthermore, denoting by $\theta_1 := p_\theta(Y = 1)$ and $\theta_0 := p_\theta(Y = 0)$, we have that

$$\begin{aligned} &\theta_1 + \theta_0 = 1 \\ &\theta_1, \theta_0 \in [0,1]. \end{aligned} \tag{4.23}$$

Given a training data set $\mathbf{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, following information can be computed directly for all pairs of indices (p, t_p):

$$\begin{aligned} s_1 = |\{n | y_n = 1\}|, s_0 = |\{n | y_n = 0\}| \\ s_1^{p,t_p} = |\{n | x_n^p = a_{t_p}^p y_n = 1\}|, s_0^{p,t_p} = |\{n | x_n^p = a_{t_p}^p y_n = 0\}|. \end{aligned} \tag{4.24}$$

The underlying hypothesis space $\Theta \subseteq \mathbb{R}^f$, where $f = 2\sum_{p=1}^P T_p + 2$, and its individual θ can be rewritten as follows:

$$\theta := (\theta_1, \theta_0, \theta_1^{p,t_p}, \theta_0^{p,t_p} | t_p = 1 \ldots T_p, p = 1, \ldots, P), \tag{4.25}$$

$$\Theta := \{\theta | \theta \text{ satisfies (4.22) and (4.23)}\}. \tag{4.26}$$

The log-likelihood function is defined for binary Naive Bayes classifier as follows [14]

$$\begin{aligned} l(\theta \,|\, \mathbf{D}) :&= \sum_{n=1}^{N} \ln \left(p_\theta(y_n) \prod_{p=1}^{P} p_\theta(X^p = x_n^p | Y = y_n) \right) \\ &= s_1 \ln(\theta_1) + s_0 \ln(\theta_0) + \sum_{p=1}^{P} \left[s_1^{p,t_p} \ln(\theta_1^{p,t_p}) + s_0^{p,t_p} \ln(\theta_0^{p,t_p}) \right]. \end{aligned} \tag{4.27}$$

with its maximum estimate θ^*

$$\begin{aligned} \left((\theta_1)^*, (\theta_0)^*\right) &= \left(\frac{s_1}{N}, \frac{s_0}{N} \right), \\ \left((\theta_1^{p,t_p})^*, (\theta_0^{p,t_p})^*\right) &= \left(\frac{s_1^{p,t_p}}{s_1}, \frac{s_0^{p,t_p}}{s_0} \right). \end{aligned} \tag{4.28}$$

Thus, the corresponding regularized log-likelihood estimate θ^* are

$$\left((\theta_1^{p,t_p})^r, (\theta_0^{p,t_p})^r\right) = (1-\gamma) \left(\frac{s_1^{p,t_p}}{s_1}, \frac{s_0^{p,t_p}}{s_0} \right) + \frac{\gamma}{2}, \tag{4.29}$$

where $\gamma \in [0, 1]$ is the regularization parameter and will be fixed to be 0.001. Let us note that we employ the regularization form of the log-likelihood estimate here to avoid unexpected effect caused by the zero count, which is common when works with Naive Bayes, especially when dealing with small data sets with a large number of features.

We now focus on determining the degree of support for the positive class (4.4) and then summarize the results for the negative class (4.5) (which can be determined in a similar manner). Given an unlabelled instance $\boldsymbol{t} = (t^1, \ldots, t^P)$, denoting by

$$\theta_{\boldsymbol{t}}^{p,1} := p_\theta(X^p = t^p | Y = 1) = \theta_1^{p,t^p}, p = 1, \ldots, P, \tag{4.30}$$

$$\theta_{\boldsymbol{t}}^{p,0} := p_\theta(X^p = t^p | Y = 0) = \theta_0^{p,t^p}, p = 1, \ldots, P. \tag{4.31}$$

Then, the degree of support (4.4) can be rewritten explicitly as follows

$$\pi(1|\mathbf{t}) = \sup_{\theta \in \Theta} \min \left[\pi_\Theta(\theta \,|\, \mathbf{D}), \max\left(2\theta(\boldsymbol{t}) - 1, 0\right) \right] \tag{4.32}$$

$$\text{where, } \theta(\boldsymbol{t}) = \frac{\theta_1 \prod_{p=1}^{P} \theta_{\boldsymbol{t}}^{p,1}}{\theta_1 \prod_{p=1}^{P} \theta_{\boldsymbol{t}}^{p,1} + \theta_0 \prod_{p=1}^{P} \theta_{\boldsymbol{t}}^{p,0}} \tag{4.33}$$

Let us notice that the target function to be maximized in (4.32) is not necessarily concave, which can lead to difficulties when maximizing the function. We propose the following approach inspired by ε-contamination model. Given θ^*, the maximum likelihood estimate computed using (4.28), for a given number $\varepsilon \in [0, 1]$, we define a contour region Θ_ε as follows

$$\Theta_\varepsilon = \{\theta | \theta \in \Theta \cap [(1-\varepsilon)\theta^*, (1-\varepsilon)\theta^* + \varepsilon]\}. \tag{4.34}$$

The intuitive idea we interest here is to enlarge Θ_ε from a singleton $\{\theta^*\}$ to the entire hypothesis space Θ by increasing ε from 0 to 1. Following this direction, we can, as pointed out in the following, simultaneously increase $\theta(\boldsymbol{t})$ and decrease $\pi_\Theta(\theta \,|\, \mathbf{D})$ (in general). Thus, starting from the highest value of $\pi_\Theta(\theta \,|\, \mathbf{D})$, we will converge to the value of $\pi_\Theta(\theta \,|\, \mathbf{D})$ where $\theta(\boldsymbol{t})$ and $\pi_\Theta(\theta \,|\, \mathbf{D})$ are identical (or closed in practice), which

is the optimal estimate for the solution of (4.32).

For a contour region Θ_ε, the highest value of $\theta(\boldsymbol{t})$ attained at

$$\begin{aligned}\theta_\varepsilon^{\boldsymbol{t}} &= \arg\max_{\theta\in\Theta_\varepsilon}\theta(\boldsymbol{t}) \\ &= \big((1-\varepsilon)(\theta_1)^* + \epsilon, (1-\varepsilon)(\theta_0)^*, (1-\varepsilon)(\theta_{\boldsymbol{t}}^{p,1})^r + \varepsilon, (1-\varepsilon)(\theta_{\boldsymbol{t}}^{p,0})^r, p = 1,\ldots,P\big).\end{aligned} \tag{4.35}$$

The formula given in (4.35) comes from the combination of the monotonicity of $\theta(\boldsymbol{t})$ and the property that guarantees the feasibility of $\theta_\varepsilon^{\boldsymbol{t}}$ s.t,

$$(1-\varepsilon)(\theta_1)^* + \varepsilon + (1-\epsilon)(\theta_0)^* = (1-\varepsilon)\big((\theta_1)^* + (\theta_0)^*\big) + \varepsilon = 1, \forall\varepsilon\in[0,1].$$

When assessing the probability $\theta(\boldsymbol{t})$, the regularization $\big((\theta_{\boldsymbol{t}}^{p,1})^r, (\theta_{\boldsymbol{t}}^{p,0})^r\big)$ is employed (instead of $\big((\theta_{\boldsymbol{t}}^{p,1})^*, (\theta_{\boldsymbol{t}}^{p,0})^*\big)$) to overcome the effect of zero count. Furthermore, the monotonicity of $\theta(\boldsymbol{t})$ ensures that we can increase $\theta(\boldsymbol{t})$ (and consequently $2\theta(\boldsymbol{t})-1$) by increase ε. In other words, for $\varepsilon \leq \varepsilon^{'}$, we have that $\theta_\varepsilon^{\boldsymbol{t}}(\boldsymbol{t}) \leq \theta_{\varepsilon^{'}}^{\boldsymbol{t}}(\boldsymbol{t})$.

It is worth noticing that $\theta_\varepsilon^{\boldsymbol{t}}$ only contains $2 + 2 * d$ variables which is relatively smaller than f, the total number of variables within θ. Thus, the highest value $\theta(\boldsymbol{t})$ over Θ_ε is associated to a region $\theta_\varepsilon^{\boldsymbol{t}} \cap \Theta_\varepsilon$. That is to fix all the variables given in (4.35) while letting others freely as long as the condition of belonging to Θ_ε still satisfied.

The highest value that $\overline{\pi}_\Theta(\theta \,|\, \mathbf{D})$ can attain within Θ_ε when fixing $\theta(\boldsymbol{t})$ to be $\theta_\varepsilon^{\boldsymbol{t}}(\boldsymbol{t})$ can be determined as follows:

$$\overline{\pi}_\Theta(\theta_\varepsilon^{\boldsymbol{t}}) = \arg\max_{\theta\in\Theta_\varepsilon\cap\theta_\varepsilon^{\boldsymbol{t}}} \pi_\Theta(\theta \,|\, \mathbf{D}). \tag{4.36}$$

Thus, the highest degree of support $\pi(1|\boldsymbol{t})$ can be given to $\boldsymbol{t}$ over the hypothesis region Θ_ε can be approximated as

$$\pi_\varepsilon(1|\boldsymbol{t}) = \min\big(2\theta_\varepsilon^{\boldsymbol{t}}(\boldsymbol{t}) - 1, \overline{\pi}_\Theta(\theta_\varepsilon^{\boldsymbol{t}} \,|\, \mathbf{D})\big) \tag{4.37}$$

To this end, we obtain an estimate of $\pi(1|\mathbf{t})$ as follows

$$\pi(1|\mathbf{t}) = \arg\max_{\varepsilon\in[0,1]} \pi_\varepsilon(1|\mathbf{t}). \tag{4.38}$$

Let us notice that at the beginning, we always have that $\overline{\pi}_\Theta(\theta_0^{\boldsymbol{t}} \,|\, \mathbf{D}) \geq 2\theta_0^{\boldsymbol{t}}(\boldsymbol{t}) - 1$. This observation is quite interesting for making an early stopping criteria (as the optimization problem (4.36) could be expensive due to large number of variables) that is to continually increase ε (from 0 to 1) as long as $\overline{\pi}_\Theta(\theta_\varepsilon^{\boldsymbol{t}} \,|\, \mathbf{D}) \geq 2\theta_\varepsilon^{\boldsymbol{t}}(\boldsymbol{t}) - 1$ and stop as soon as the side is reversed, i.e, when seeing $\varepsilon^{'}$ s.t $\overline{\pi}_\Theta(\theta_{\varepsilon^{'}}^{\boldsymbol{t}} \,|\, \mathbf{D}) \leq 2\theta_{\varepsilon^{'}}^{\boldsymbol{t}}(\boldsymbol{t}) - 1$. The intuitive idea of this criteria is that when seeing a reverse, we are quite sure that we just already jumped over the crossing point, i,e, the optimal solution $\theta^{\boldsymbol{t}}$ s.t

$$\pi(1|\boldsymbol{t}) = \pi_\Theta(\theta^{\boldsymbol{t}} \,|\, \mathbf{D}) = 2\theta^{\boldsymbol{t}}(\boldsymbol{t}) - 1,$$

Thus, simply approximating $\pi(1|\boldsymbol{t}) = \pi_\varepsilon(1|\boldsymbol{t})$ could give a close estimate. Readers interested in more accurate approximations are recommended to do a further search within the region $[\varepsilon, \varepsilon']$.

Following a similar manner, we find an estimate of $\pi(0|\mathbf{t})$ s.t

$$\pi(0|\mathbf{t}) = \arg\max_{\epsilon\in[0,1]} \pi_\varepsilon(0|\mathbf{t}). \tag{4.39}$$

In practice, we evaluate Eqs. (4.38) and (4.39) on uniform discretizations of cardinality 200 of $[0, 1]$.

4.2 Active learning

In this proposal, we advocate a distinction between two different types of uncertainty, referred to as *epistemic* and *aleatoric*, in the context of active learning. We conjecture that, in uncertainty sampling, the usefulness of an instance is better reflected by its epistemic than by its aleatoric uncertainty. This leads us to suggest the principle of *epistemic uncertainty sampling*, which we instantiate by means of a concrete approach for measuring epistemic and aleatoric uncertainty.

4.2.1 Related methods

In this section, we recall the setting of uncertainty sampling and present two recent approaches that are related to our work in that they also distinguish different sources of uncertainty.

Uncertainty sampling

As usual in active learning, we assume to be given a labelled set of training data $\mathbf{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N}$ and a pool of unlabeled instances $\mathbf{U} = \{(\boldsymbol{t}_t, ?)\}_{t=1}^{T}$ that can be queried by the learner. Instances are represented as features vectors $\boldsymbol{x}_n = \left(x_n^1, \ldots, x_n^P\right) \in \mathcal{X} = \mathbb{R}^P$. In this proposal, we only consider the case of binary classification, where labels y_n are taken from $\mathcal{Y} = \{0, 1\}$, leaving the more general case of multi-class classification for future work.

In uncertainty sampling, instances are queried in a greedy fashion. Given the current model θ that has been trained on $\mathbf{D}$, each instance $\boldsymbol{t}$ in the current pool $\mathbf{U}$ is assigned a *utility* score $s(\theta, \boldsymbol{t})$, and the next instance to be queried is the one with the highest score [55, 80, 81, 85]. The chosen instance is labelled (by an oracle or expert) and added to the training data $\mathbf{D}$, on which the model is then re-trained. The active learning process for a given budget B (i.e, the number of unlabelled instances to be queried) is summarized in Algorithm 18.

Algorithm 18: Uncertainty sampling

Input: $\mathbf{U}$, $\mathbf{D}$, θ- initial pool, training data, classifier, and B-budget
Output: $\mathbf{U}$, $\mathbf{D}$, θ - updated pool, training data, classifier

initialize $b = 0$;
while $b < B$ **do**
  foreach $\boldsymbol{t} \in \mathbf{U}$ **do**
    compute $s(\theta, \boldsymbol{t})$
  query the label of the optimal instance $\boldsymbol{t}^*$ with respect to $s(\theta, \boldsymbol{t})$
  $\mathbf{D} = \mathbf{D} \cup \{\boldsymbol{t}^*, y^*\}$;
  $\mathbf{U} = \mathbf{U} \setminus \{\boldsymbol{t}^*, y^*\}$;
  train θ from $\mathbf{D}$;
  $b = b + 1$;
Return $\mathbf{U}$, $\mathbf{D}$, θ;

Assuming a probabilistic model producing predictions in the form of probability distributions $p_\theta(\cdot \,|\, \boldsymbol{t})$ on $\mathcal{Y}$, the utility score is typically defined in terms of a measure of uncertainty. Thus, instances on which the current model is highly uncertain are supposed to be maximally informative [80, 81, 85]. Popular examples of such measures include

– the entropy:

$$s(\theta, \boldsymbol{t}) = -\sum_{y \in \mathcal{Y}} p_\theta(y \,|\, \boldsymbol{t}) \log p_\theta(y \,|\, \boldsymbol{t}) \,, \tag{4.40}$$

– the least confidence:

$$s(\theta, \boldsymbol{t}) = 1 - \max_{y \in \mathcal{Y}} p_\theta(y \,|\, \boldsymbol{t}) \,, \tag{4.41}$$

– the smallest margin:

$$s(\theta, \boldsymbol{t}) = p_\theta(y_m \,|\, \boldsymbol{t}) - p_\theta(y_n \,|\, \boldsymbol{t}) \,, \tag{4.42}$$

where $y_m = \arg\max_{y \in \mathcal{Y}} p_\theta(y \,|\, \boldsymbol{t})$ and $y_n = \arg\max_{y \in \mathcal{Y} \setminus y_m} p_\theta(y \,|\, \boldsymbol{t})$.

While the first two measures ought to be maximized, the last one has to be minimized. In the case of binary classification, i.e, $\mathcal{Y} = \{0, 1\}$, all these measures rank unlabelled instances in the same order and look for instances with small difference between $p_\theta(0 \,|\, \boldsymbol{t})$ and $p_\theta(1 \,|\, \boldsymbol{t})$.

Evidence-based uncertainty sampling

In their evidence-based uncertainty sampling approach [85], the authors propose to differentiate between *conflicting-evidence uncertainty* and *insufficient-evidence uncertainty*. The corresponding measures are specifically tailored for the Naive Bayes classifier as a learning algorithm.

In the spirit of the Naive Bayes predictor, evidence-based uncertainty sampling first looks at the influence of individual features t^p in the feature representation $\boldsymbol{t} = (t^1, \ldots, t^P)$ of instances. More specifically, given the current model θ, denote by $p_\theta(t^p \,|\, 0)$ and $p_\theta(t^p \,|\, 1)$ the class-conditional probabilities on the values of the p^{th} feature. For a given instance $\boldsymbol{t}$, the set of features is partitioned into those that provide evidence for the positive and for the negative class, respectively:

$$P_\theta(\boldsymbol{t}) := \left\{ t^p \;\middle|\; \frac{p_\theta(t^p \,|\, 1)}{p_\theta(t^p \,|\, 0)} > 1 \right\}, \tag{4.43}$$

$$N_\theta(\boldsymbol{t}) := \left\{ t^p \;\middle|\; \frac{p_\theta(t^p \,|\, 0)}{p_\theta(t^p \,|\, 1)} > 1 \right\}. \tag{4.44}$$

Then, the total evidence for the positive and the negative class is determined as follows:

$$E_1(\boldsymbol{t}) = \prod_{t^p \in P_\theta(\boldsymbol{t})} \frac{p_\theta(t^p \,|\, 1)}{p_\theta(t^p \,|\, 0)} \,, \tag{4.45}$$

$$E_0(\boldsymbol{t}) = \prod_{t^p \in N_\theta(\boldsymbol{x})} \frac{p_\theta(t^p \,|\, 0)}{p_\theta(t^p \,|\, 1)} \,. \tag{4.46}$$

The conflicting evidence-based approach simply queries the instance with the highest conflicting evidence, while the insufficient evidence-based approach looks for the one with the highest insufficient evidence:

$$\boldsymbol{t}^*_{conf} = \arg\max_{\boldsymbol{t}\in\mathbf{S}} \big(E_1(\boldsymbol{t}) \times E_0(\boldsymbol{t})\big)\,, \tag{4.47}$$

$$\boldsymbol{t}^*_{insu} = \arg\min_{\boldsymbol{t}\in\mathbf{S}} \big(E_1(\boldsymbol{t}) \times E_0(\boldsymbol{t})\big)\,. \tag{4.48}$$

Note that the selection is restricted to the set $\mathbf{S}$ of top high uncertain instances, i.e., those instances $\boldsymbol{t}$ in the pool $\mathbf{U}$ having a high score $s(\theta, \boldsymbol{t})$ according to standard uncertainty sampling. This ensures that the evidences for the two classes, $E_0(\boldsymbol{t})$ and $E_1(\boldsymbol{t})$, are close to each other. Then, a high conflicting evidence (4.47) captures the case where the evidences are close and both of large magnitude. In other words, it refers to the situation where a model is highly uncertain about an instance, and has strong but conflicting evidence for both classes. On the other hand, a high insufficient-evidence uncertainty (4.48) refers to the case where a model is highly uncertain about an instance because of not having enough evidence for either class.

Note, however, that this line of reasoning neglects the influence of the prior class probabilities, which is especially relevant in the case of imbalanced class distributions. In such cases, evidence-based uncertainty may strongly deviate from standard uncertainty, i.e., the entropy of the posterior distribution. For instance, $E_0(\boldsymbol{t})$ and $E_1(\boldsymbol{t})$ could both be very large, and $p_\theta(\boldsymbol{t}\,|\,0) \approx p_\theta(\boldsymbol{t}\,|\,1)$, although $p_\theta(0\,|\,\boldsymbol{t})$ is very different from $p_\theta(1\,|\,\boldsymbol{t})$ due to unequal prior odds, and hence the entropy small. Likewise, the entropy of the posterior can be large although both evidence-based uncertainties are small.

Credal uncertainty sampling

Credal uncertainty sampling [3] is another approach that seeks to differentiate between the reducible and irreducible part of the uncertainty. Denote by $C \subseteq \Theta$ a *credal set* of models, i.e., a set of plausible candidate models. We say that a class y dominates another class y' if y is more probable than y' for each distribution in the credal set, that is

$$s(y, y', \boldsymbol{t}) := \inf_{\theta\in C} \frac{p_\theta(y\,|\,\boldsymbol{t})}{p_\theta(y'\,|\,\boldsymbol{t})} > 1\,. \tag{4.49}$$

The credal uncertainty sampling approach simply looks for the instance $\boldsymbol{t}$ with the highest uncertainty, i.e, the least evidence for the dominance of one of the classes. In the case of binary classification with $\mathcal{Y} = \{0, 1\}$, this is expressed by the score

$$s(\boldsymbol{t}) = -\max\big(s(1, 0, \boldsymbol{t}), s(0, 1, \boldsymbol{t})\big)\,. \tag{4.50}$$

Practically, the computations are based on the interval-valued probabilities, denoted by $[\underline{p}_C(y\,|\,\boldsymbol{t}), \overline{p}_C(y\,|\,\boldsymbol{t})]$, assigned to each class $y \in \mathcal{Y}$, where

$$\underline{p}_C(y\,|\,\boldsymbol{t}) := \inf_{\theta\in C} p_\theta(y\,|\,\boldsymbol{t})\,, \quad \overline{p}_C(y\,|\,\boldsymbol{t}) := \sup_{\theta\in C} p_\theta(y\,|\,\boldsymbol{t})\,. \tag{4.51}$$

Such interval-valued probabilities can be produced within the framework of the Naive credal classifier [2, 3, 23, 103]. In the case of binary classification, where $p_\theta(0\,|\,\boldsymbol{t}) =$

$1 - p_\theta(1 \,|\, \boldsymbol{t})$, the score $s(1, 0, \boldsymbol{t})$ can be rewritten as follows:

$$s(1, 0, \boldsymbol{t}) = \inf_{\theta \in C} \frac{p_\theta(1 \,|\, \boldsymbol{t})}{p_\theta(0 \,|\, \boldsymbol{t})} = \inf_{\theta \in C} \frac{p_\theta(1 \,|\, \boldsymbol{t})}{1 - p_\theta(1 \,|\, \boldsymbol{t})} = \frac{\underline{p}_C(1 \,|\, \boldsymbol{t})}{1 - \underline{p}_C(1 \,|\, \boldsymbol{t})} \tag{4.52}$$

Likewise,

$$s(0, 1, \boldsymbol{t}) = \inf_{\theta \in C} \frac{p_\theta(0 \,|\, \boldsymbol{t})}{p_\theta(1 \,|\, \boldsymbol{t})} = \inf_{\theta \in C} \frac{1 - p_\theta(1 \,|\, \boldsymbol{t})}{p_\theta(1 \,|\, \boldsymbol{t})} = \frac{1 - \overline{p}_C(1 \,|\, \boldsymbol{t})}{\overline{p}_C(1 \,|\, \boldsymbol{t})} \,. \tag{4.53}$$

Finally, the uncertainty score (4.50) can simply be expressed as follows:

$$s(\boldsymbol{t}) = -\max\left(\frac{\underline{p}_C(1 \,|\, \boldsymbol{t})}{1 - \underline{p}_C(1 \,|\, \boldsymbol{t})}, \frac{1 - \overline{p}_C(1 \,|\, \boldsymbol{t})}{\overline{p}_C(1 \,|\, \boldsymbol{t})}\right) \tag{4.54}$$

4.2.2 Principle of our method

Let us remind that, aleatoric uncertainty is due to the influences on the data-generating process that are inherently random, whereas the epistemic uncertainty is caused by a lack of knowledge. Or, stated differently, u_e and u_a, respectively defined in (4.6) and (4.7), measure the *reducible* and the *irreducible* part of the total uncertainty, respectively. It thus appears reasonable to assume that epistemic uncertainty is more relevant for active learning: while it makes sense to query additional class labels in regions where uncertainty can be reduced, doing so in regions of high aleatoric uncertainty appears to be less reasonable.

This leads us to the principle of *epistemic uncertainty sampling*, which prescribes the selection

$$\boldsymbol{t}^* = \arg\max_{\boldsymbol{t} \in \mathbf{U}} u_e(\boldsymbol{t}) \,. \tag{4.55}$$

For comparison, we will also consider an analogous selection rule based on the aleatoric uncertainty, i.e.,

$$\boldsymbol{t}^* = \arg\max_{\boldsymbol{t} \in \mathbf{U}} u_a(\boldsymbol{t}) \,, \tag{4.56}$$

as well the toal uncertainty:

$$\boldsymbol{t}^* = \arg\max_{\boldsymbol{t} \in \mathbf{U}} (u_e(\boldsymbol{t}) + u_a(\boldsymbol{t})) \,. \tag{4.57}$$

Note that the latter is closest to standard uncertainty sampling, where the entire uncertainty is quantified in a single measure.

Let us remind that the above approach is completely generic and can in principle be instantiated with any hypothesis space Θ. The uncertainty measures (4.6–4.7) can be derived very easily from the support degrees (4.2–4.3). The computation of the latter may become difficult, however, as it requires the solution of an optimization problem, the properties of which depend on the choice of Θ (as studied in Sections 4.1.2-4.1.4).

Comparison with the evidence-based uncertainty sampling

Although the concepts of *conflicting evidence* and *insufficient evidence* of Sharma & Bilgic [85] appear to be quite related, respectively, to aleatoric and epistemic uncertainty, the correspondence becomes much less obvious (and in fact largely disappears)

upon a closer inspection. Besides, a direct comparison is complicated due to various technical issues with their evidence-based approach. In particular, we will subsequently ignore the preselection of top high uncertain instances (i.e., the set $\mathbf{S}$) in evidence-based uncertainty sampling, so as to separate the effect of their measures from standard entropy.

As a first important observation, note that the evidences $E_1(\boldsymbol{t})$ and $E_0(\boldsymbol{t})$ solely depend on the *relation* of the class-conditional probabilities $p_\theta(t^p \,|\, 1)$ and $p_\theta(t^p \,|\, 0)$, which hides the number of training examples they have been estimated from, and hence their confidence. The latter, however, has an important influence on whether we qualify something as aleatorically or epistemically uncertain. As an illustration, consider a simple example with two binary attributes, the first with domain $\{a_1, a_2\}$ and the second with domain $\{b_1, b_2\}$. Denote by $n_{i,j} = (n^+_{i,j}, n^-_{i,j})$ the number of positive and negative example observed for $(t^1, t^2) = (a_i, b_j)$. Here are three scenarios:

	b_1	b_2
a_1	$(1,1)$	$(1,1)$
a_2	$(1,1)$	$(1,1)$

	b_1	b_2
a_1	$(100,100)$	$(100,100)$
a_2	$(100,100)$	$(100,100)$

	b_1	b_2
a_1	$(1,1)$	$(10,1)$
a_2	$(1,10)$	$(1,1)$

In the first two scenarios, the insufficient evidence would be high, because all class-conditional probabilities are equal. In our approach, however, the first scenario would largely be a case of epistemic uncertainty, due to the few number of training examples, whereas the second would be aleatoric, because the equal posteriors[4] are sufficiently *confirmed*.

Similar remarks apply to conflicting evidence. In the third scenario, the latter would be high for (a_1, b_1), because $p_\theta(a_1 \,|\, 1) \gg p_\theta(a_1 \,|\, 0)$ and $p_\theta(b_1 \,|\, 0) \gg p_\theta(b_1 \,|\, 1)$. The same holds for (a_2, b_2), whereas the uncertainties for (a_1, b_2) and (a_2, b_1) would be low. Note, however, that in all these cases, exactly the same conditional probability estimates $p_\theta(t^p \,|\, 1)$ and $p_\theta(t^p \,|\, 0)$ are involved.

We would argue that the epistemic uncertainty should directly refer to these probabilities, because they constitute the parameter θ of the model. Thus, to reduce the epistemic uncertainty (about the right model θ), one should look for those examples that will mostly improve the estimation of these probabilities. Aleatoric uncertainty may occur in cases of posteriors close to $1/2$, in which case the conflicting evidence may indeed be high (although, as already mentioned, the latter ignores the class priors). Yet, we would not necessarily call such cases a *conflict*, because the predictions are completely in agreement with the underlying model (Naive Bayes), which assumes class-conditional independence of attributes, i.e., an independent combination of evidences on different attributes.

Comparison with the credal uncertainty sampling

Credal uncertainty sampling seems to be closer to our approach, at least in terms of the underlying principle. In both approaches, the model uncertainty is captured in terms of a set of plausible candidate models from the underlying hypothesis space, and this (epistemic) uncertainty about the right model is translated into uncertainty about the prediction for a given $\boldsymbol{t}$. In the credal uncertainty sampling, the candidate set is given by the credal set C, which corresponds to the distribution $\pi_\Theta(\theta \,|\, \mathbf{D})$ in our approach–as a difference, we thus note that ours is a *graded set*, to which a candidate θ belongs with a certain degree of membership (the relative likelihood), whereas a credal set is a standard set in which a model is either included or not. Using machine learning

[4] The class priors are ignored here.

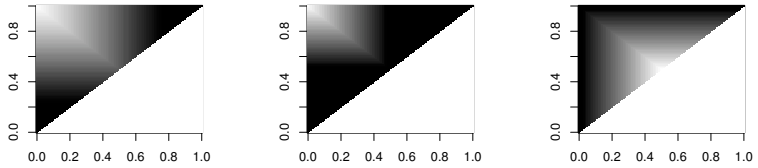


FIGURE 4.2: From left to right: Exponential rescaling of the credal uncertainty measure, epistemic uncertainty and aleatoric uncertainty for interval probabilities with lower probability (x-axis) and upper probability (y-axis). Lighter colors indicate higher values.

terminology, C plays the role of a *version space* [63], whereas $\pi_\Theta(\theta \,|\, \mathbf{D})$ represents a kind of generalized (graded) version space.

More specifically, the wider the interval $[\underline{p}_C(1 \,|\, \boldsymbol{t}), \overline{p}_C(1 \,|\, \boldsymbol{t})]$ in (4.54), the larger the score $s(\boldsymbol{t})$, with the maximum being obtained for the case $[0, 1]$ of complete ignorance. This is well in agreement with our degree of epistemic uncertainty. In the limit, when $[\underline{p}_C(1 \,|\, \boldsymbol{t}), \overline{p}_C(1 \,|\, \boldsymbol{t})]$ reduces to a precise probability $p_\theta(1 \,|\, \boldsymbol{t})$, i.e., the epistemic uncertainty disappears, (4.54) is maximal for $p_\theta(1 \,|\, \boldsymbol{t}) = 1/2$ and minimal for $p_\theta(1 \,|\, \boldsymbol{t})$ close to 0 or 1. Again, this behavior is in agreement with our conception of aleatoric uncertainty. More generally, comparing two intervals of the same length, (4.54) will be larger for the one that is closer to the middle point $1/2$. Thus, it seems that the credal uncertainty score (4.54) combines both epistemic and aleatoric uncertainty in a single measure.

Yet, upon closer examination, its similarity to our measure of epistemic uncertainty is much higher than the similarity to aleatoric uncertainty. Note that, for our approach, the special case of a credal set C can me imitated with the measure $\pi_\Theta(\theta \,|\, \mathbf{D}) = 1$ if $\theta \in C$ and $\pi_\Theta(h)(\theta \,|\, \mathbf{D}) = 0$ if $\theta \notin C$. Then, (4.2) and (4.3) become

$$\pi(1 \,|\, \boldsymbol{t}) = \sup_{\theta \in C} \max[\, 2\, p_\theta(1 \,|\, \boldsymbol{t}) - 1, 0\,] = \max[\, 2\, \overline{p}_C(1 \,|\, \boldsymbol{t}) - 1, 0\,],$$
$$\pi(0 \,|\, \boldsymbol{t}) = \sup_{\theta \in C} \max[\, 2\, p_\theta(0 \,|\, \boldsymbol{t}) - 1, 0\,] = \max[\, 1 - 2\, \underline{p}_C(1 \,|\, \boldsymbol{t}), 0\,],$$

and u_e and u_a can be derived from these values as before. Figure 4.2 shows a graphical illustration of the credal uncertainty score[5] (4.54) as a function of the probability bounds $\underline{p}_C$ and $\overline{p}_C$, and the same illustration is given for epistemic uncertainty u_e and aleatoric uncertainty u_a. From the visual impression, it is clear that the credibility score closely resembles u_e, while behaving quite differently than u_a. This impression is corroborated by a simple correlation analysis, in which we ranked the intervals

$$[\underline{p}_C, \overline{p}_C] \in \left\{ I_{a,b} = \left[\frac{a}{100}, \frac{b}{100} \right] \,\middle|\, a, b \in \{0, 1, \ldots, 100\},\, a \leq b \right\},$$

i.e., a quantization of the class of all probability intervals, according to the different measures, and then computed the Kendall rank correlation. While the ranking according to (4.54) is strongly correlated with the ranking for u_e (Kendall is around 0.86), it is almost uncorrelated with u_a.

[5] The score s is not well scaled, and may assume very large negative values. For better visibility, we therefore plotted the monotone transformation $\exp(s)$.

#	name	# instances	# features	attributes
1	parkinsons	197	22	real
2	vertebral-column	310	6	real
3	ionosphere	351	34	real
4	climate-model	540	18	real
5	breast-cancer	569	30	real
6	blood-transfusion	748	5	real
7	banknote-authentication	1372	4	real

TABLE 4.1: Data set used in the experiments

In summary, the credal uncertainty score appears to be quite similar to our measure of epistemic uncertainty. As potential advantages of our approach, let us mention the following points. First, our degree is normalized and bounded, and thus easier to interpret. Second, it is complemented by a degree of aleatoric uncertainty—the two degrees are carefully distinguished and have a clear semantics. Third, handling candidate models in a graded manner, and modulating their influence according to their plausibility, appears to be more reasonable than creating an artificial separation into plausible and non-plausible models (i.e., the credal set and its complement).

4.2.3 Experimental evaluation

Some experiments are conducted to illustrate the performance of our uncertainty measures in active learning. Our main concern here is how fast different uncertainty sampling approaches improve the performance of classifiers and restrict ourselves to three classical models that are the local model, the logistic regression and the Naive Bayes classifier.

Local method

Data sets and experimental setting

We perform experiments on data sets from the UCI repository whose descriptions are given in Table 4.1. We follow a 5×5-fold cross-validation procedure: each data set is randomly split into 5 folds. Each fold is in turn considered as the test set, while the other folds are used for learning. The latter is randomly split into a training data set and a pool set. The proportions of (training, pool, test) sets are $(20\%, 60\%, 20\%)$. The whole procedure is repeated 5 times, and accuracies are averaged. The budget of the active learner is fixed to be the length of the pool, and the performance of the classifiers is monitored over the entire learning process.

After each query, we update the data sets and, correspondingly, the classifiers. The improvements of the classifiers are compared for four different uncertainty measures, i.e., uncertainty sampling (following the strategy presented in Algorithm 18) based on four measures for selecting unlabelled instances: standard uncertainty (4.41), epistemic uncertainty (4.6), aleatoric uncertainty (4.7), total of epistemic and aleatoric uncertainty (4.57).

We also evaluate how quickly the querying procedure will be able to fill the low density regions. To this end, we measure the maximal distance between testing instances and their nearest neighbours.

Experimental results

Experiments were conducted for two values of the width ϵ, corresponding to neighborhood sizes $K = 4$ and $K = 8$. These can be considered as a small and large width of the Parzen window. Since the results are very similar, we only present those for the case $K = 8$.

As it can be seen in Figure 4.3, the results are nicely in agreement with our expectations: the epistemic uncertainty sampling performs the best and the aleatoric uncertainty sampling the worst. Moreover, the standard uncertainty sampling is in-between the two, very similar to total uncertainty (aleatoric plus epistemic). This supports our conjecture that, from an active learning point of view, the epistemic uncertainty is the more useful information. Even if the improvements compared to standard uncertainty sampling are not huge, they are still visible and quite consistent.

Figure 4.4 also shows that the epistemic uncertainty sampling achieves the best coverage of the instance space, measured in terms of the maximal distance between testing instances and their nearest neighbours. As expected, the aleatoric uncertainty is again the worst, and standard uncertainty sampling is in-between.

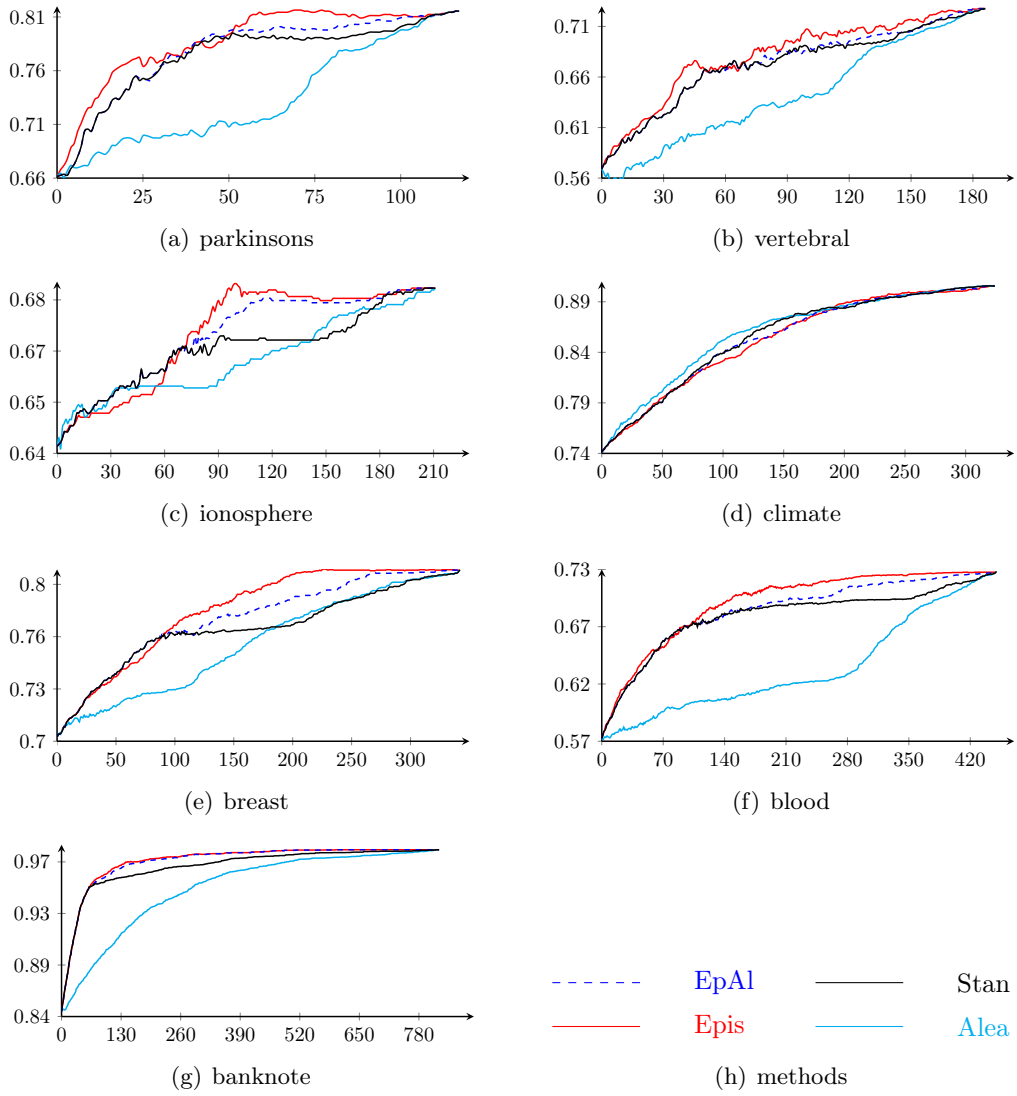
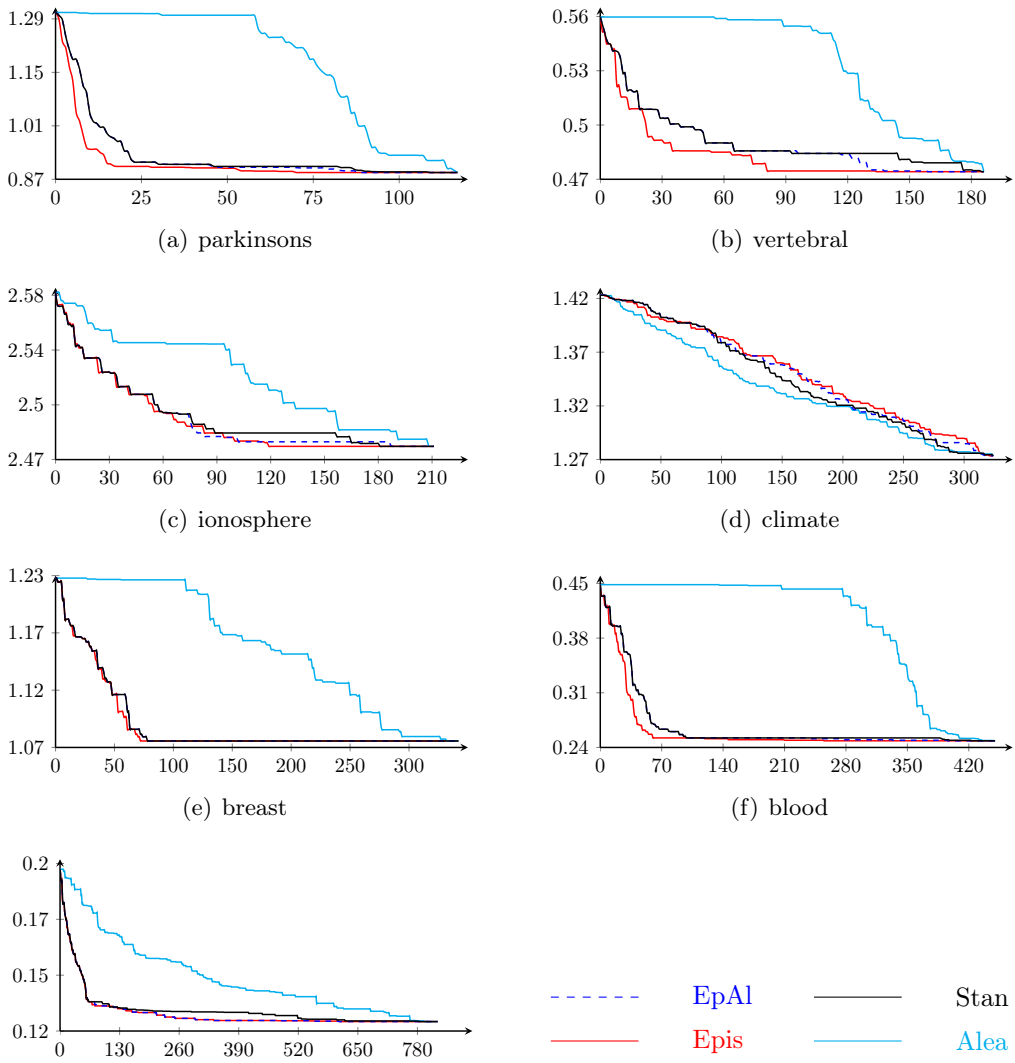


FIGURE 4.3: Average accuracies (y-axis) over 5×5-folds for the Parzen window classifier (K = 8) as a function of the number of examples queried from the pool (x-axis).



(a) parkinsons (b) vertebral

(c) ionosphere (d) climate

(e) breast (f) blood

(g) banknote (h) methods

FIGURE 4.4: Average maxmin distances (y-axis) over 5×5-folds for the Parzen window classifier (K = 8) as a function of the number of examples queried from the pool (x-axis).

Logistic regression

Data sets and experimental setting

We perform experiments on the same UCI data sets as before (cf. Table 4.1). To avoid the relatively strong bias imposed by the linear model assumption, we start with a very small amount of initial training data, thereby making improvements in the beginning more visible. We conduct a 10×3-fold cross validation procedure: each data set is split into 3 folds. Each fold is in turn considered as the learning set, while other folds are used for testing. The learning set is randomly split into a training data set and a pool set. The proportions of (training, pool, test) set are (1%, 32%, 67%). The whole procedure is repeated 10 times, and the accuracies are averaged. Similar to the case of the local learning, we fix the budget to be the length of the pool.

In addition to accuracy, we monitor the convergence of the ML estimate $\hat{\theta}$ toward the *best* model θ^*. Since the latter is not known, we use the parameter that would have been learned on the entire data as a surrogate. More specifically, we measure the convergence in terms of the Euclidean distance $\|\hat{\theta} - \theta^*\|$, and average over a sufficiently large number of repetitions to smooth the curves.

As before, the uncertainty sampling (Algorithm 18) is instantiated with four measures for selecting unlabelled instances: standard uncertainty (4.41), epistemic uncertainty (4.6), aleatoric uncertainty (4.7), and the sum of epistemic and aleatoric uncertainty (4.57). This time, we also include the conflicting-evidence (Conf) and insufficient-evidence (Insu) measures by Sharma & Bilgic [85][6]. Let us remind that these measures are tailored for Naive Bayes as a classifier. Yet, in contrast to the case of local learning, a comparison is now meaningful, because both linear regression and Naive Bayes construct a linear decision boundary.

Experimental results

As can be seen in Figure 4.5, the epistemic uncertainty sampling does again perform quite well in comparison to the others, except on the ionosphere data. Moreover, it achieves the overall best convergence to the best model, as shown in Figure 4.6. Furthermore, in Figure 4.6, it is clear that the improvements provided by difference uncertain measure are well-fitted to our expectation that epistemic and aleatoric uncertainty sampling provide, respectively, the best and the least improvement while the classical uncertainty sampling and the total of epistemic and aleatoric uncertainty provide something in between. Finally, no general pattern has been drawn for evidence-based uncertain measures.

Compared with the case of local learning, however, the improvements in comparison to standard uncertainty sampling are now smaller, and sometimes completely disappear. This is arguably due to the relatively strong bias imposed by the linear model assumption: Although we initialize with a comparatively small set of training data, the learning curves converge quite quickly (in the case of climate and blood, there is almost no improvement at all). In other words, the linear model is more or less fixed from the beginning, so that it becomes difficult for any sampling strategy to make a real difference.

[6]For better comparison, we use the measures in a *pure* form, that is, without using the *high uncertainty* criterion as a pre-filter. Thus, we seek to avoid mixing the effect of their measures with standard entropy.

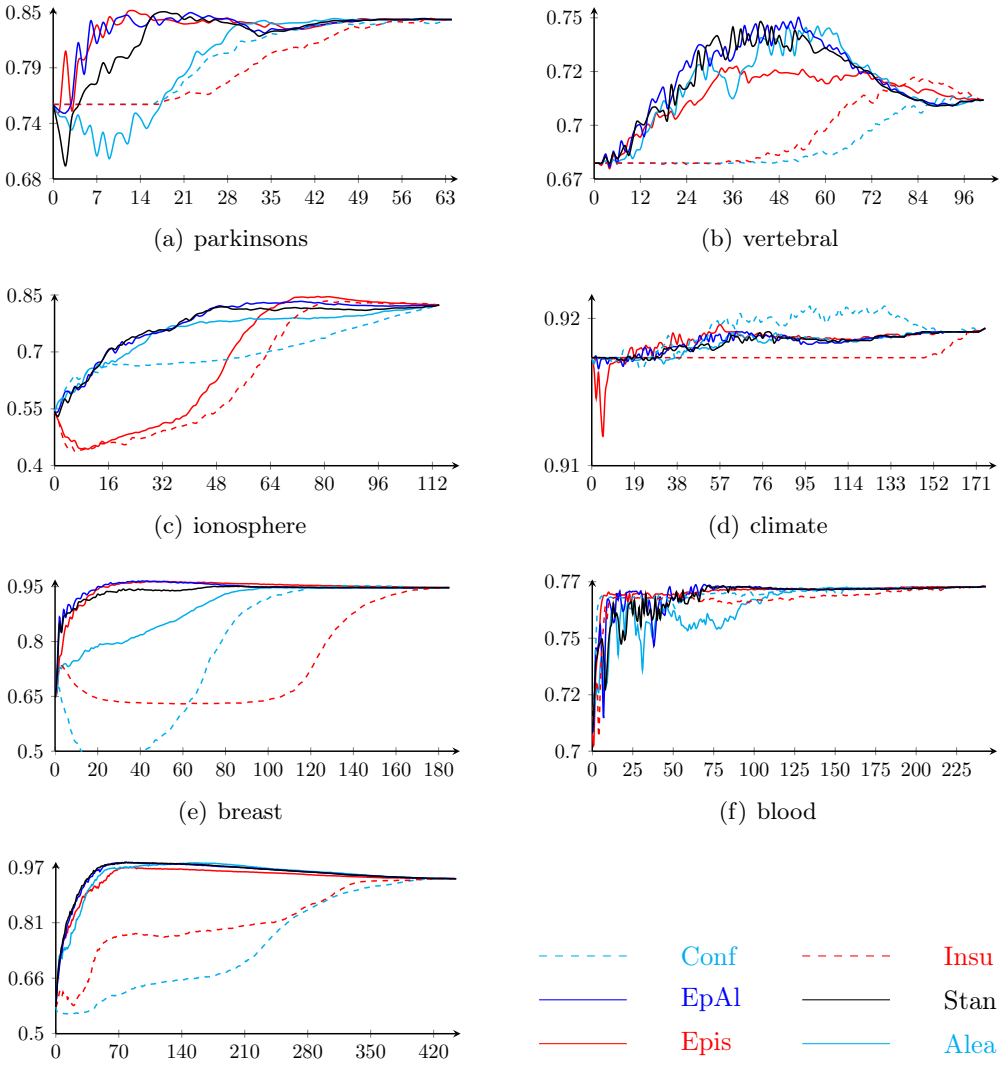


FIGURE 4.5: Average accuracies (y-axis) over 10×3-folds for logistic regression as a function of the number of examples queried from the pool (x-axis).

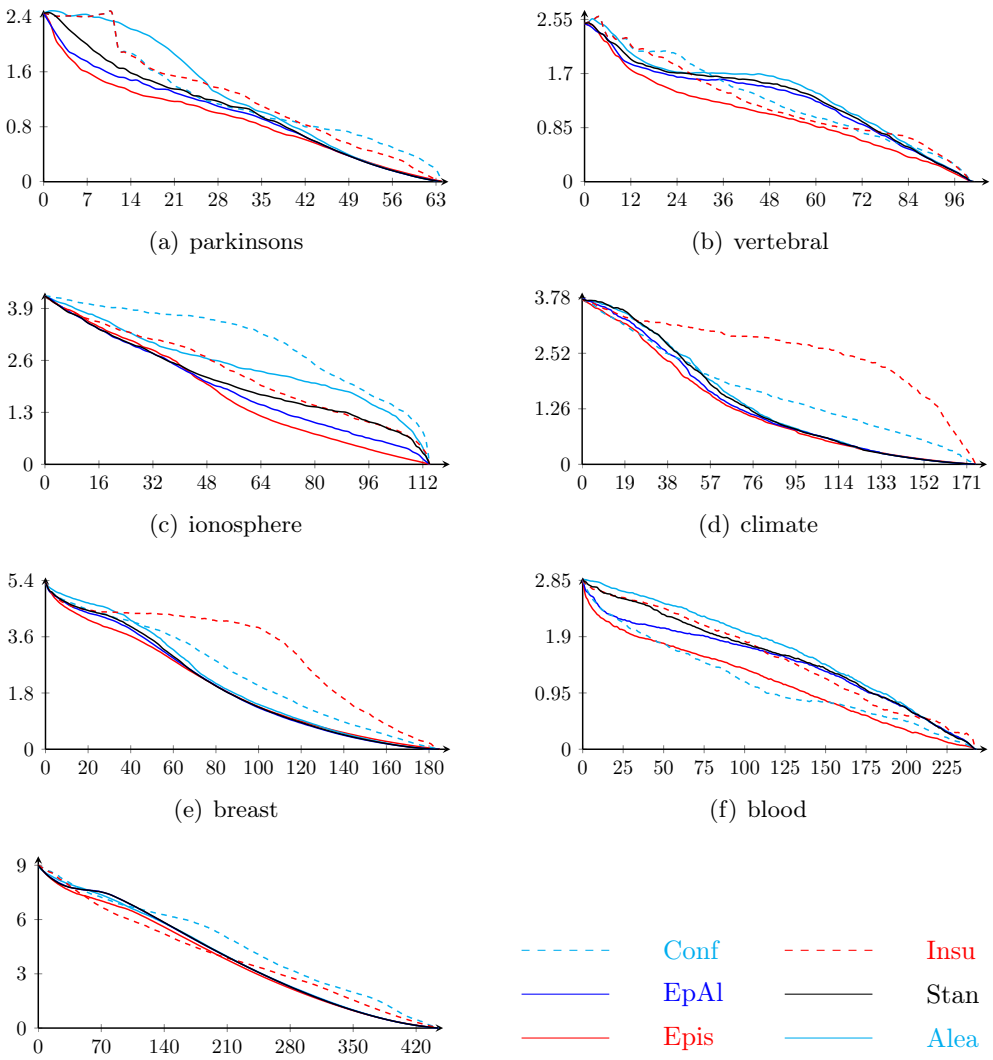


FIGURE 4.6: Average distances (y-axis) over 10×3-folds for logistic regression as a function of the number of examples queried from the pool (x-axis).

Naive Bayes

Data sets and experimental setting

We perform experiments on the first two small data sets described in the Table 4.1. Similarly to the case of logistic regression, we start with a very small amount of initial training data, thereby making improvements in the beginning more visible. We conduct a 10×3-fold cross validation procedure: Each data set is split into 3 folds. The learning set is randomly split into a training data set and a pool set. The proportions of (training, pool, test) set are (5%, 28%, 67%). The whole procedure is repeated 10 times, and the accuracies are averaged. We fix the budget to be the length of the pool. Let us note that we start from a slightly larger training data to reduce the effect of the zero frequency problem which could present unexpected effects when assessing the improvements provided by the methods.

In addition to accuracy, we monitor the convergence of the Kullback–Leibler divergence (KL divergence) of $\hat{\theta}$ toward the *best* model θ^*. Since the latter is not known, we use the parameter that would have been learned on the entire data as a surrogate. More specifically, we measure convergence in terms of $D_{KL}(\theta^* \parallel \hat{\theta})$, which is often called the information gain achieved if $\hat{\theta}$ is used instead of θ^*, such that:

$$D_{KL}(\theta^* \parallel \hat{\theta}) = -\sum_i \theta^*(i) \ln \left(\frac{\hat{\theta}(i)}{\theta^*(i)} \right).$$

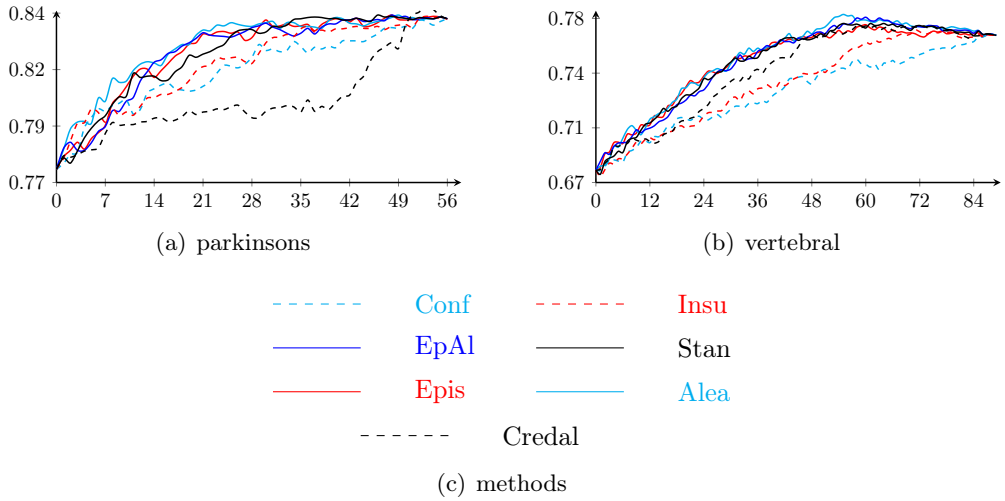
As before, uncertainty sampling (Algorithm 18) is instantiated with six measures for selecting unlabelled instances: standard uncertainty (4.41), epistemic uncertainty (4.6), aleatoric uncertainty (4.7), the sum of epistemic and aleatoric uncertainty (4.57), the conflicting-evidence (Conf) and insufficient-evidence (Insu) measures by Sharma & Bilgic [85]. In addition, we employ the credal uncertainty sampling (Credal), which shares similar purpose with our interests and is applicable for Naive Bayes, to select unlabelled instances.

Experimental results

As can be seen in Figure 4.7-4.8, there are very similar improvements provided by four methods: standard uncertainty (4.41), epistemic uncertainty (4.6), aleatoric uncertainty (4.7), the sum of epistemic and aleatoric uncertainty (4.57). The evidence-based methods and credal uncertainty sampling (Credal) appear less effective in this test.

We think that these behaviors could be due to the presence of zero frequencies. For instance, if we see zero frequencies when assessing an instance $\boldsymbol{t}$, (4.33) implies that the probabilities assigned for both classes are close to 0.5. On the other hand, the plausible hypotheses tend to assign for $\boldsymbol{t}$ the conditional probabilities around 0.5. Consequently, (4.19) and (4.20) suggest that the degrees of support for both classes are close to zero, i.e, a high degree of aleatoric uncertainty.

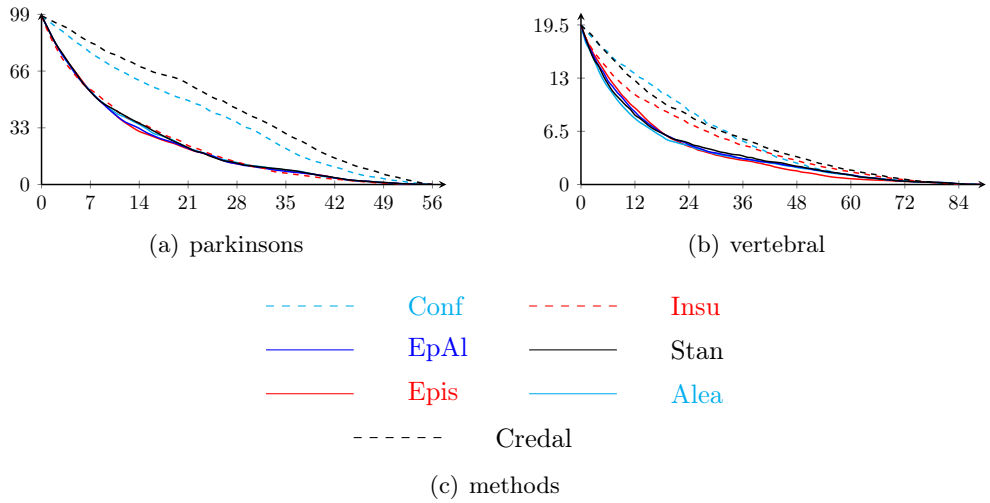
We thus derive a hypothesis that zero frequency problem have introduced another kind of uncertainty (lack of knowledge on the unobserved parameters θ_1^{p,t_p} and θ_0^{p,t_p}) that will be preferred by both the standard uncertainty (4.41) and aleatoric uncertainty (4.7). In contrast, the epistemic uncertainty (4.6) is interested in the lack of knowledge on some observed parameters. How to effectively investigate such situations is not obvious, we thus leave it as an open problem.



(a) parkinsons (b) vertebral

(c) methods

FIGURE 4.7: Average accuracies (y-axis) over 10 × 3-folds for Naive Bayes as a function of the number of examples queried from the pool (x-axis).



(a) parkinsons (b) vertebral

(c) methods

FIGURE 4.8: Average KL divergence (y-axis) over 10 × 3-folds for Naive Bayes as a function of the number of examples queried from the pool (x-axis).

4.3 Cautious inference

This section presents a method for reliable prediction in multi-class classification, where the reliability refers to the possibility of partial abstention in cases of uncertainty. More specifically, we allow for predictions in the form of preorder relations on the set of classes, thereby generalizing the idea of set-valued predictions. Our approach relies on combining learning by pairwise comparison with the distinction made between reducible (a.k.a. *epistemic*) uncertainty caused by a lack of information and irreducible (a.k.a. *aleatoric*) uncertainty due to intrinsic randomness. The problem of combining uncertain pairwise predictions into a most plausible preorder is then formalized as an integer programming problem. This inference procedure is inspired by the belief functions-based approach proposed recently by Masson et al. [58].

4.3.1 Principle of our method

We are going to present our approach to reliable multi-class prediction, which is based on the idea of binary decomposition and a stepwise simplification (approximation) of the information contained in the set of pairwise comparisons between classes—first in terms of a preorder and then in terms of a set.

Learning by Pairwise Comparison

In the multi-class classification setting, we are dealing with a set of $M > 2$ classes $\mathcal{Y} = \{y_1, \ldots, y_M\}$. Suppose a set of training data $\mathbf{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$ to be given, and denote by $\mathbf{D}_m = \{\boldsymbol{x}_n \,|\, (\boldsymbol{x}_n, y_m) \in \mathbf{D}\}$ the observations from class y_m.

Learning by pairwise comparison (LPC) a.k.a. all-pairs is a decomposition technique that trains one (binary) classifier $\theta_{i,j}$ for each pair of classes (y_i, y_j), $1 \leq i < j \leq M$ [36]. The task of $\theta_{i,j}$, which is trained on $\mathbf{D}_{i,j} = \mathbf{D}_i \cup \mathbf{D}_j$, is to separate instances with label y_i from those having label y_j. Suppose we solve these problems with the approach described in the previous section, instead of using a standard binary classifier. Then, given a new query instance $\boldsymbol{t} \in \mathcal{X}$, we can produce predictions in the form of a quadruple

$$\mathbf{I}^{i,j}(\boldsymbol{t}) := \left(s_{y_i}^{i,j}(\boldsymbol{t}), s_{y_j}^{i,j}(\boldsymbol{t}), u_e^{i,j}(\boldsymbol{t}), u_a^{i,j}(\boldsymbol{t})\right), \tag{4.58}$$

one for each pair of classes (y_i, y_j). These predictions can also be summarized in three $[0,1]^{M \times M}$ relations, a (strict) preference relation P, an indifference relation A, and an incomparability relation E:

$$P = \left(s_{y_i}^{i,j}(\boldsymbol{t})\right)_{i,j},\ A = \left(u_a^{i,j}(\boldsymbol{t})\right)_{i,j},\ E = \left(s_e^{i,j}(\boldsymbol{t})\right)_{i,j}$$

Let us note that, in our approach, predictions are always derived per instance, i.e., for an individual query instance $\boldsymbol{t}$. Likewise, all subsequent inference steps are tailored for that instance. Keeping this in mind, we will henceforth simplify notations and often omit the dependence of scores and relations on $\boldsymbol{t}$.

Inferring a preorder

The structure (P, A, E) provides a rich source of information, which we seek to represent in a condensed form. To this end, we approximate this structure by a preorder R. This approximation may also serve the purpose of correction, since the relational structure (P, A, E) is not necessarily consistent; for example, since all binary classifiers are trained independently of each other, their predictions are not necessarily transitive.

Recall that a preorder is a binary relation $R \subseteq \Omega \times \Omega$ that is reflexive. In the following, we will also use the following notation:

$$\begin{aligned} y_i \succ_R y_j \text{ (or simply } y_i \succ y_j) \quad &\text{if} \quad r_{i,j} = 1,\, r_{j,i} = 0\,, \\ y_i \sim_R y_j \text{ (or simply } y_i \sim y_j) \quad &\text{if} \quad r_{i,j} = 1,\, r_{j,i} = 1\,, \\ y_i \perp_R y_j \text{ (or simply } y_i \perp y_j) \quad &\text{if} \quad r_{i,j} = 0,\, r_{j,i} = 0\,, \end{aligned}$$

where $r_{i,j} = 1$ if $(y_i, y_j) \in R$ and $r_{i,j} = 0$ if $(y_i, y_j) \notin R$. Note that the binary relations $\succ$, $\sim$, $\perp$ are in direct correspondence with the relations P, A, and E, respectively.

How compatible is a relation R with a structure (P, A, E)? Interpreting the scores (4.58) as probabilities, we could imagine that a relation R is produced by randomly "hardening" the soft (probabilistic) structure (P, A, E), namely by selecting one of the relations $y_i \succ y_j$, $y_j \succ y_i$, $y_i \sim y_j$, $y_i \perp y_j$ with probability $s_{y_i}^{i,j}$, $s_{y_j}^{i,j}$, $u_a^{i,j}$, and $u_e^{i,j}$, respectively. Then, making a simplifying assumption of independence, the probability of ending up with R is given as follows:

$$p(R) = \prod_{y_i \succ_R y_j} s_{y_i}^{i,j} \prod_{y_j \succ_R y_i} s_{y_j}^{i,j} \prod_{y_i \perp_R y_j} u_e^{i,j} \prod_{y_i \sim_R y_j} u_a^{i,j} \tag{4.59}$$

The most probable preorder R^* then corresponds to

$$R^* = \arg\max_{R \in \mathbf{R}} p(R)\,, \tag{4.60}$$

where $\mathbf{R}$ is the set of all preorders on $\mathcal{Y}$.

Let us now propose a practical procedure to determine R^*, which is based on representing the optimization problems (4.60) as a binary linear integer program. To this end, we introduce the following variables:

$$X_{i,j}^1 = r_{i,j}(1 - r_{j,i}), X_{i,j}^2 = r_{j,i}(1 - r_{i,j}), X_{i,j}^3 = (1 - r_{i,j})(1 - r_{j,i}), X_{i,j}^4 = r_{i,j} r_{j,i}.$$

Then, by adding the constraints $\sum_{l=1}^4 X_{i,j}^l = 1$ and $X_{i,j}^l \in \{0, 1\}$, we can rewrite the probability (4.59) as follows:

$$p(R) = \prod_{i<j} \left(s_{\lambda_i}^{i,j}\right)^{X_{i,j}^1} \left(s_{\lambda_j}^{i,j}\right)^{X_{i,j}^2} \left(u_e^{i,j}\right)^{X_{i,j}^3} \left(u_a^{i,j}\right)^{X_{i,j}^4} \tag{4.61}$$

Furthermore, the transitivity property

$$r_{i,k} + r_{k,j} - 1 \leq r_{i,j}, \quad \forall\, i \neq j \neq k. \tag{4.62}$$

can easily be encoded by noting that $r_{i,j} = X_{i,j}^1 + X_{i,j}^4$ and $r_{i,j} = X_{j,i}^2 + X_{j,i}^4$ if $i < j$ and $j < i$, respectively.

Altogether, the most probable preorder $R^* \in \mathbf{R}$ is determined by $X^* = (X_{i,j}^1, \ldots, X_{i,j}^4)_{i,j}$, which is the solution of the following optimization problem:

$$\max \sum_{i<j} X_{i,j}^1 \ln\left(s_{\lambda_i}^{i,j}\right) + X_{i,j}^2 \ln\left(s_{\lambda_j}^{i,j}\right) + X_{i,j}^3 \ln\left(u_e^{i,j}\right) + X_{i,j}^4 \ln\left(u_a^{i,j}\right) \tag{4.63}$$

$$\begin{aligned} \text{s.t.} \quad & \sum_{l=1}^4 X_{i,j}^l = 1,\, \forall\, 1 \leq i < j \leq M\,, \\ & X_{i,j}^1, X_{i,j}^2, X_{i,j}^3, X_{i,j}^4 \in \{0, 1\},\, \forall\, 1 \leq i < j \leq M\,, \\ & r_{i,k} + r_{k,j} - 1 \leq r_{i,j},\, \forall\, i \neq j \neq k\,. \end{aligned}$$

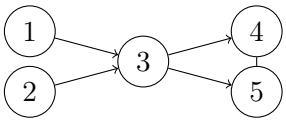


FIGURE 4.9: Preorder induced by Example 18 (strict preference symbolized by directed edge, indifference by undirected edge, incomparability by missing edge).

Note that if $u_e^{i,j} = 0$ for all pairs, then the solution will be a complete preorder (in which the binary relations are either $\sim$ or $\succ$) between class probabilities, which is consistent with our interpretation. Similarly, if $u_a^{i,j} = 0$ and $u_e^{i,j} = 0$ for all pairs, we would obtain a linear ordering, as in [12].

Obtaining credible sets from R^*

Consider the preorder $R^* = R^*(\boldsymbol{t})$ for an unlabelled query instance $\boldsymbol{t}$, and suppose we seek a set-valued prediction $\theta(\boldsymbol{t}) \subseteq \mathcal{Y}$. A reasonable way to obtain such a prediction is to collect all non-dominated classes, i.e., to exclude only those classes y_j for which $y_i \succ_{R^*} y_j$ for at least one competing class y_i. A class label of that kind can be seen as a potentially optimal prediction for $\boldsymbol{t}$. Adopting the above notation, the set-valued prediction can thus be determined as

$$\theta(\boldsymbol{t}) = \left\{ y_i \in \mathcal{Y} \mid \sum_{j<i} X_{j,i}^1 + \sum_{i<j} X_{i,j}^2 = 0 \right\}, \tag{4.64}$$

which means that it can immediately be derived from the solution of (4.63). Note that full uncertainty, i.e, $\theta(\boldsymbol{t}) = \mathcal{Y}$, only occur if all pairs (y_i, y_j) are incomparable or indifferent.

How to obtain a set-valued prediction from the pairwise information is illustrated in the following example.

Example 18. *Assume that we have the output space $\mathcal{Y} = \{y_1, \ldots, y_5\}$ and pairwise information (4.58) for an unlabelled instance $\boldsymbol{t}$ given by the following quadruples:*

$$\begin{aligned}
&\mathbf{I}^{1,2}(\boldsymbol{t}) = (0, 0.1, 0.6, 0.3), && \mathbf{I}^{1,3}(\boldsymbol{t}) = (0.6, 0, 0.1, 0.2),\\
&\mathbf{I}^{1,4}(\boldsymbol{t}) = (0.9, 0, 0.1, 0), && \mathbf{I}^{1,5}(\boldsymbol{t}) = (0.4, 0, 0.3, 0.3),\\
&\mathbf{I}^{2,3}(\boldsymbol{t}) = (0.6, 0, 0.2, 0.2), && \mathbf{I}^{2,4}(\boldsymbol{t}) = (0.7, 0, 0, 0.3),\\
&\mathbf{I}^{2,5}(\boldsymbol{t}) = (0.9, 0, 0, 0.1), && \mathbf{I}^{3,4}(\boldsymbol{t}) = (0.6, 0, 0.2, 0.2),\\
&\mathbf{I}^{3,5}(\boldsymbol{t}) = (0.9, 0, 0.1, 0), && \mathbf{I}^{4,5}(\boldsymbol{t}) = (0.05, 0.05, 0.4, 0.5).
\end{aligned}$$

Solving the optimization problem (4.63) gives the most probable preorder R^* pictured in Figure 4.9 with the corresponding value X^* s.t. $X_{1,2}^3 = X_{1,3}^1 = X_{2,3}^1 = X_{3,4}^1 = X_{3,5}^1 = X_{4,5}^4 = 1$. Finally, from (4.64) we get $\theta(\boldsymbol{t}) = \{1, 2\}$.

4.3.2 Experimental evaluation

This section presents some experimental results to assess the performance of our approach to reliable classification.

#	name	# instances	# features	# labels
a	iris	150	4	3
b	wine	178	13	3
c	forest	198	27	4
d	seeds	210	7	3
e	glass	214	9	6
f	ecoli	336	7	8
g	libras	360	91	15
h	dermatology	385	34	6
i	vehicle	846	18	4
j	vowel	990	10	11
k	yeast	1484	8	12
l	wine quality	1599	11	6
m	optdigits	1797	64	10
n	segment	2300	19	7
o	wall-following	5456	24	4

TABLE 4.2: Data sets used in the experiments

Data sets and experimental setting

We perform experiments on 15 data sets from the UCI repository (cf. Table 4.2), following a 10×10-fold cross-validation procedure. We compare the performance of our method (referred to as PREORDER) with two competitors. To make the results as comparable as possible, these methods are also implemented with pairwise learning using a logistic regression classifier as base learner. Thus, they only differ in how the pairwise information provided by the logistic regression is turned into a (reliable) multi-class prediction.

- VOTE: The first method is based on aggregating pairwise predictions via standard voting, which is a common approach in LPC. However, instead of simple weighted voting, we apply the more sophisticated aggregation technique proposed in [46], which shows better performance. Note that, by predicting the winner of the voting procedure, this approach always produces a precise prediction.

- NONDET: As a baseline for set-valued predictions, we use the approach of [22], which has been shown to exhibit competitive performance in comparison to other imprecise prediction methods [104]. Recall that this approach produces nondeterministic predictions from precise probabilistic assessments. This requires turning pairwise probability estimates into conditional probabilities $(p_\theta(y_1 \mid \boldsymbol{t}), \ldots, p_\theta(y_M \mid \boldsymbol{t}))$ on the classes, a problem known as pairwise coupling. To this end, we apply the δ_2 method, which performs best among those investigated in [96].

Evaluation metrics for assessing set-valued predictions have to balance correctness (the true class y is an element of the predicted set $Y := \theta(t)$) and precision (size of the predicted set) in an appropriate manner. For example, in [104], the authors argue that using the simple discounted accuracy ($1/|Y|$ if $y \in Y$ and 0 otherwise) is equivalent to saying that producing a set-valued prediction is the same as choosing within this set (uniformly) at random. This means that the discounted accuracy does not reward any cautiousness. Also, it can be shown that minimizing the expected discounted accuracy in expectation would never lead to imprecise predictions [102].

	VOTE	PREORDER		NONDET	
#	acc.	u_{80}	u_{65}	u_{80}	u_{65}
a	84.33(3, 1)	90.45(1)	83.29(2)	86.71(2)	76.88(3)
b	96.35(1, 1)	95.89(2)	93.18(2)	93.47(3)	88.92(3)
c	89.76(2, 1)	92.15(1)	88.82(2)	88.49(3)	81.57(3)
d	88.81(3, 1)	92.15(1)	88.16(2)	90.03(2)	83.60(3)
e	47.14(3, 3)	67.32(1)	57.24(1)	65.03(2)	52.98(2)
f	75.57(3, 1)	80.66(1)	75.25(2)	77.02(2)	68.89(3)
g	50.50(3, 3)	70.51(1)	63.91(1)	62.50(2)	53.02(2)
h	96.43(2, 2)	97.70(1)	96.46(1)	96.01(3)	93.38(3)
i	63.99(3, 1)	71.07(1)	62.17(2)	68.92(2)	57.17(3)
j	39.57(3, 2)	51.10(1)	42.57(1)	48.22(2)	37.27(3)
k	49.35(3, 2)	60.60(2)	50.04(1)	60.84(1)	49.22(3)
l	58.10(3, 3)	69.65(2)	59.92(1)	71.02(1)	59.16(2)
m	96.37(3, 2)	97.67(1)	96.81(1)	96.85(2)	95.46(3)
n	84.51(3, 3)	91.87(1)	89.16(1)	90.01(2)	85.49(2)
o	68.69(3, 3)	76.42(2)	70.79(1)	77.34(1)	70.39(2)
aver.	(u_{80}, u_{65})	u_{80}	u_{65}	u_{80}	u_{65}
rank	(2.73, 1.93)	1.27	1.40	2.00	2.67

TABLE 4.3: Average utility-discounted accuracies (%)

Here, we therefore adopt the *average utility-discounted accuracy* measure, which has been proposed and formally justified in [104]:

$$u(y, Y) = \begin{cases} 0 & \text{if } y \notin Y \\ \frac{\phi_1}{|Y|} - \frac{\phi_2}{|Y|^2} & \text{otherwise} \end{cases}$$

More specifically, we use the measures u_{65} with $(\phi_1, \phi_2) = (1.6, 0.6)$ and u_{80} with $(\phi_1, \phi_2) = (2.2, 1.2)$. Note that, in the case of precise decisions, both u_{65} and u_{80} reduce to standard accuracy.

Experimental Results

The average performances in terms of the utility-discounted accuracies are shown in Table 4.3, with ranks in parenthesis (note that we provide one set of ranks for u_{65}, and another one for u_{80}). Firstly, we notice that PREORDER yields the best average ranks over the 15 data sets, both for u_{80} and u_{65}. Furthermore, a Friedman test [24] on the ranks yields p-values of 0.0003138 and 0.002319 for u_{80} and u_{65}, respectively, thus strongly suggesting performance differences between the algorithms. The Nemenyi post-hoc test (see Table 4.4) further indicates that PREORDER is significantly better than VOTE regarding u_{80} and NONDET in the case of u_{65}. Since u_{80} rewards cautious predictions stronger than u_{65} does, it is not surprising that indeterminate classifiers do better in this case. Yet, even when considering u_{65}, PREORDER remains competitive with VOTE. This suggests that it tends to be more precise than NONDET, while still accurately recognizing those instances for which we have to be cautious.

Ideally, an imprecise classifier should abstain (i.e., provide set-valued predictions) on difficult cases, on which the precise classifier is likely to fail [101]. The goal of Figure

#	H_0	u_{80}	u_{65}
1	$V = P$	**0.00017**	0.3101
2	$V = N$	0.11017	0.1102
3	$P = N$	0.11017	**0.0015**

TABLE 4.4: Nemenyi post-hoc test: null hypothesis H_0 and p-value

4.10(a,b) is to verify this ability. Figure 4.10(a) displays, for each data set, the percentage of times the true class is in the prediction of PREORDER, given the prediction was imprecise, versus the accuracy of VOTE on those instances. Figure 4.10(b) does the same for NONDET. Both imprecise classifiers achieve high percentages (> 80) of correct partial predictions, while the corresponding percentages of VOTE vary in a wider range. Also, the accuracy of the latter significantly drops on those instances (for example, the average accuracy for data set g is 50% in Table 4.3, but drops to less than 30% in Figure 4.10(a)), confirming that the imprecise classifiers do indeed abstain on difficult cases. Finally, note that the points in Figure 4.10(a) are a bit more to the left than those in Figure 4.10(b), again suggesting that PREORDER is doing slightly better in recognizing difficult instances than NONDET.

For the two imprecise classifiers, we also compare the average proportion of partial predictions and the average (normalized) size of the predictions when at least one method produces a partial prediction. Figures 4.10(c) and 4.10(d) indicate that NONDET produces more partial predictions of (slightly) larger size.

4.4 Conclusion

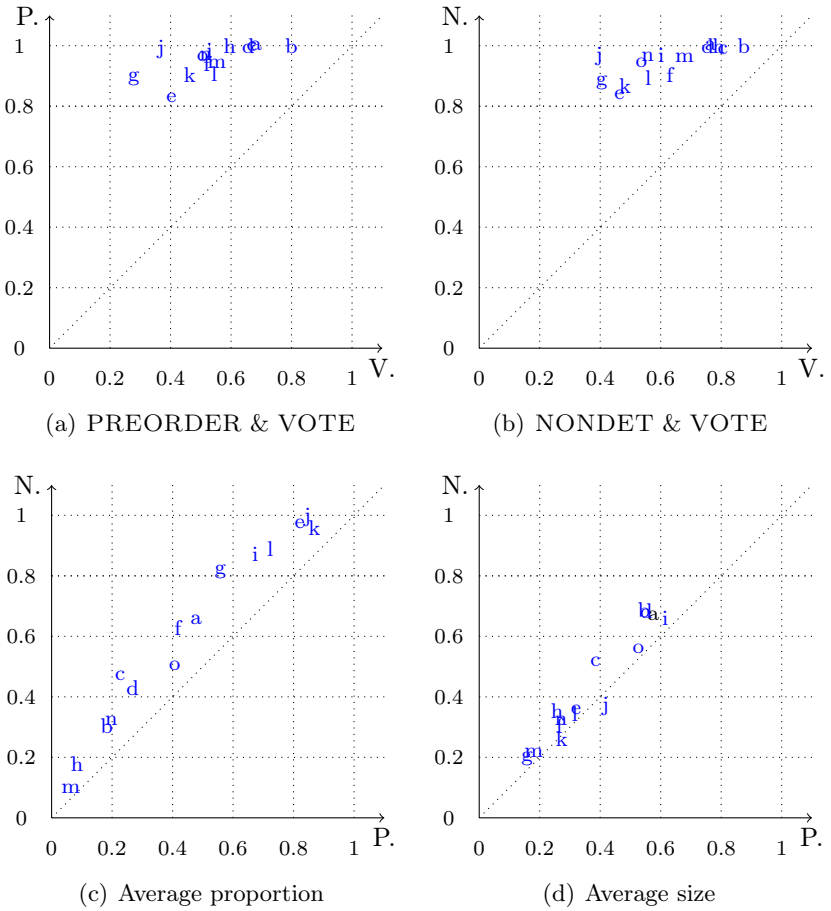
Yet the distinction between epistemic and aleatoric uncertainty has been increasingly studied, a lack of efficient techniques to estimate these degrees of the uncertainty seems to restrict its subsequent applications. We have proposed estimators for popular classifiers and used it to solve two machine learning problems: active learning and cautious inference. Our general conclusion is that the distinction between epistemic and aleatoric uncertainty can indeed provide advantages for subsequent machine learning applications.

4.4.1 Active learning

We reconsider the principle of uncertainty sampling in active learning from the perspective of uncertainty modeling. More specifically, it starts from the supposition that, when it comes to the question of which instances to select from a pool of candidates, a learner's predictive uncertainty due to *not knowing* should be more relevant than its uncertainty due to inherent randomness.

To corroborate this conjecture, we have proposed the *epistemic uncertainty sampling*, in which standard uncertainty measures such as the entropy are replaced by a novel measure of epistemic uncertainty. The latter is borrowed from a recent framework for uncertainty modeling, in which the epistemic uncertainty is distinguished from the aleatoric uncertainty [79]. In comparison to previous proposals based on similar ideas, our approach is arguably more principled. Moreover, it is completely generic and can be instantiated with any (probabilistic) classifier as a learning algorithm.

We interpret the experiments conducted with a simple local learning algorithm (Parzen window classifier) and logistic regression as evidence in favor of our conjecture. They clearly show that a separation of the total uncertainty (into epistemic and



(a) PREORDER & VOTE (b) NONDET & VOTE

(c) Average proportion (d) Average size

FIGURE 4.10: (a) Correctness of the PREORDER in the case of abstention versus accuracy of the VOTE. (b) Correctness of the NONDET in the case of abstention versus accuracy of the VOTE. (c) Proportion of partial predictions when at least one method produces a partial prediction. (d) Average normalized size of the predictions in such cases.

aleatoric) is effective, and that the epistemic part is the better criterion for selecting instances to be queried. As already said, investigating how to effectively implement our approach for the case of Naive Bayes requires significant extra efforts and will be left as an open problem.

4.4.2 Cautious inference

We have introduced an approach to cautious inference and reliable prediction in multi-class classification. The basic idea is to provide predictions in the form of preorder relations, which allow for representing preferences for some candidate classes over others, as well as indifference and incomparability between them; the two latter relations are in direct correspondence with two types of uncertainty, aleatoric and epistemic. This can be seen as a sophisticated way of partial abstention, which generalizes set-valued predictions and classification with reject option. Technically, our approach combines reliable binary classification with pairwise decomposition and approximate inference over preorders.

Our experiments on this type of problem are quite promising and suggest that our method is highly competitive to existing approaches to reliable prediction. Yet, by

using to the set of maximal elements, we only used preorder predictions for the purpose of set-valued classification. The preoder, however, provides very rich information about the preference for classes, which could be used for other purposes.

Chapter 5

Conclusion, perspectives and open problems

In this work, we have studied different aspects of imprecision treatment, focusing on two potential settings where imprecision due to imperfect data and imperfect knowledge, respectively. Considering the former setting, we have studied both the problem of making inference and the one of learning an optimal model from partially specified data. We have investigated different potential situations where one may have to deal with multiple optimal decisions (either labels or models) due to the presence of partial data and developed active learning techniques to tackle these situations. We have focused, in the later setting, on the situations where data are precisely specified, however, these are classes that can not be distinguished due to a lack of knowledge or due to a high uncertainty. In particular, we have advocated a distinction between epistemic and aleatoric uncertainty in machine learning problems.

The main conclusions from Chapter 2, in which we have (1) implemented the maximax approach for the case of partially featured data and (2) developed active learning approaches to reduce the imprecision in the inference step due to the presence of partial data, are following:

- We can employ the maximax approach to make inferences from partially specified data using tractable and scalable techniques. Furthermore, in complement to the promising results regarding the case of partially labelled data, our experiments indicate that, in the case of partially featured data, a simple imputation method could often work as well as the maximax approach, but for some data sets the maximax approach can bring a real advantage. This conclusion can motivate further research on broadening the applications of the maximax approach.

- The possible and necessary label sets have appeared to be efficient tools for quantifying the imprecision introduced to learners by partial data. Experimentally, our investigation have indicated that (1) there are situations where partial data can indeed affect the predictive ability of the maximax approach (e.g, when employing a small number K or there is a large amount of partial labels) and (2) by doing active learning, we can significantly improve the performance of the maximax approach.

- The perspectives we provided in the end of this Chapter could benefit future attempts on tackling both the problem of making inferences and the active learning in the generic setting of partially specified data.

The first conclusion from Chapter 3 is that, together with the active learning proposal presented in Chapter 2, we have addressed different settings of the active learning problem for partial data. This problem has been little explored in the literature, in

particular in the case of partially featured data. Furthermore, the improvements on all criteria suggest that the presence of partial data can introduce significant imprecision to the learning step. Considering the case of partially featured data, our racing algorithms have consistently outperformed other simple baselines. This means that doing active learning in this case is a promising direction while it is not necessarily the case for partially labelled data where even random strategies performs similar to others. Yet, our proposals have been developed upon noticeable intuitions. Developing more sophisticated approaches would be a worthy research direction.

From Chapter 4, we can conclude that a separation of the total uncertainty into epistemic and aleatoric part is effective.

- In the active learning problem, the epistemic part has appeared to be the better criterion for querying instances. Given this affirmation, we are now encouraged to elaborate on *epistemic uncertainty sampling* in more depth, and to develop it in more sophistication. This also includes an extension to other active learning strategies (e.g., expected model change).

- Considering the problem of making cautious inferences, the distinction *epistemic/aleatoric* uncertainty provides pairwise information from which we can learn predictions in the form of preorder relations. Such a preorder allows for representing preferences for some candidate classes over others, as well as indifference and incomparability between them. It thus suggests reasons for *why a class should be included into or discarded from the set-valued prediction*. This characteristic gives the ability to appropriately balance reliability and precision which is a crucial demand when doing cautious inferences. Thus, next research efforts should focus on exploiting more of the potential of preorder predictions, and to use such predictions in other contexts and problem settings. In active learning, for example, preorder predictions may provide very useful information for guiding the selection of queries. Since our approach applies as soon as a likelihood is defined, extending it to other kinds of likelihood such as evidential ones [25] would be another promising direction.

Bibliography

1. Ahlberg, E. *et al. Using conformal prediction to prioritize compound synthesis in drug discovery* in *The 6th Symposium on Conformal and Probabilistic Prediction with Applications (COPA)* (2017), 174–184.
2. Antonucci, A. & Cuzzolin, F. *Credal Sets Approximation by Lower Probabilities: Application to Credal Networks* in *Proceedings of the 13th international Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU)* (Springer, 2010), 716–725.
3. Antonucci, A., Corani, G. & Gabaglio, S. *Active Learning by the Naive Credal Classifier* in *Proceedings of the Sixth European Workshop on Probabilistic Graphical Models (PGM)* (2012), 3–10.
4. Balasubramanian, V., Ho, S.-S. & Vovk, V. *Conformal Prediction for Reliable Machine Learning: Theory, Adaptations and Applications* (Morgan Kaufmann, 2014).
5. Betzler, N. & Dorn, B. Towards a Dichotomy for the Possible Winner Problem in Elections based on Scoring Rules. *Journal of Computer and System Sciences* **76,** 812–836 (2010).
6. Birnbaum, A. On the Foundations of Statistical Inference. *Journal of the American Statistical Association* **57,** 269–306 (1962).
7. Bishop, C. M. *Pattern recognition and machine learning* (springer, 2006).
8. Bottou, L. & Vapnik, V. Local Learning Algorithms. *Neural Computation* **4,** 888–900 (1992).
9. Briggs, F., Fern, X. Z. & Raich, R. *Rank-loss support instance machines for MIML instance annotation* in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)* (2012), 534–542.
10. Burges, C. J. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery* **2,** 121–167 (1998).
11. Chapelle, O. *Active Learning for Parzen Window Classifier* in *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics (AISTATS)* **5** (2005), 49–56.
12. Cheng, W. & Hüllermeier, E. *Probability estimation for multi-class classification based on label ranking* in *Proceedings of the 2012 European conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)* (2012), 83–98.
13. Chow, C. On optimum recognition error and reject tradeoff. *IEEE Transactions on Information Theory* **16,** 41–46 (1970).
14. Collins, M. The Naive Bayes model, Maximum-likelihood Estimation, and the EM Algorithm. *Lecture Notes.* <http://web2.cs.columbia.edu/~mcollins/em.pdf> (2012).

15. Corani, G., Abellán, J., Masegosa, A., Moral, S. & Zaffalon, M. in *Introduction to Imprecise Probabilities* 230–257 (John Wiley & Sons, Ltd, 2014).

16. Cortes, C. & Vapnik, V. Support-vector networks. *Machine Learning* **20,** 273–297 (1995).

17. Cour, T., Sapp, B., Jordan, C. & Taskar, B. *Learning from Ambiguously Labeled Images* in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2009), 919–926.

18. Cour, T., Sapp, B. & Taskar, B. Learning from Partial Labels. *Journal of Machine Learning Research* **12,** 1501–1536 (2011).

19. Couso, I. & Dubois, D. A general framework for maximizing likelihood under incomplete data. *International Journal of Approximate Reasoning* **93,** 238–260 (2018).

20. Cover, T. & Hart, P. Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory* **13,** 21–27 (1967).

21. Cox, D. R. The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B (Methodological),* 215–242 (1958).

22. Coz, J. J. d., Díez, J. & Bahamonde, A. Learning Nondeterministic Classifiers. *Journal of Machine Learning Research* **10,** 2273–2293 (2009).

23. De Campos, L. M., Huete, J. F. & Moral, S. Probability Intervals: A Tool for Uncertain Reasoning. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* **2,** 167–196 (1994).

24. Demšar, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research* **7,** 1–30 (2006).

25. Denoeux, T. Likelihood-based belief function: justification and some extensions to low-quality data. *International Journal of Approximate Reasoning* **55,** 1535–1547 (2014).

26. Denœux, T. Maximum Likelihood Estimation from Fuzzy Data using The EM algorithm. *Fuzzy Sets and Systems* **183,** 72–91 (2011).

27. Denoeux, T. Maximum Likelihood Estimation from Uncertain Data in the Belief Function Framework. *IEEE Transactions on Knowledge and Data Engineering* **25,** 119–130 (2013).

28. Devetyarov, D. *et al.* Conformal predictors in early diagnostics of ovarian and breast cancers. *Progress in Artificial Intelligence* **1,** 245–257 (2012).

29. Dobra, A. & Fienberg, S. E. Bounds for cell entries in contingency tables given marginal totals and decomposable graphs. *Proceedings of the National Academy of Sciences* **97,** 11885–11892 (2000).

30. Efron, B. Censored data and the bootstrap. *Journal of the American Statistical Association* **76,** 312–319 (1981).

31. Eklund, M., Norinder, U., Boyer, S. & Carlsson, L. The application of conformal prediction to the drug discovery process. *Annals of Mathematics and Artificial Intelligence* **74,** 117–132 (2015).

32. Fisher, R. On the Mathematical Foundations of Theoretical Statistics. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character* **222,** 309–368 (1922).

33. Fitzpatrick, P. *Advanced calculus* (American Mathematical Soc., 2006).

34. Friedman, J., Hastie, T. & Tibshirani, R. *The Elements of Statistical Learning* (Springer Series in Statistics New York, 2001).

35. Fu, Y., Zhu, X. & Li, B. A Survey on Instance Selection for Active Learning. *Knowledge and Information Systems,* 1–35 (2013).

36. Fürnkranz, J. Round robin classification. *Journal of Machine Learning Research* **2,** 721–747 (2002).

37. Grabisch, M. & Nicolas, J.-M. Classification by fuzzy integral: Performance and tests. *Fuzzy Sets and Systems* **65,** 255–271 (1994).

38. Groenen, P. J., Winsberg, S., Rodriguez, O & Diday, E. I-Scal: Multidimensional Scaling of Interval Dissimilarities. *Computational Statistics & Data Analysis* **51,** 360–378 (2006).

39. Guillaume, R., Couso, I. & Dubois, D. *Maximum Likelihood with Coarse Data based on Robust Optimisation* in *Proceedings of the Tenth International Symposium on Imprecise Probability: Theories and Applications (ISIPTA)* (2017), 169–180.

40. Guyon, I, Vapnik, V, Boser, B, Bottou, L & Solla, S. *Structural risk minimization for character recognition* in *Proceedings of the 4th International Conference on Neural Information Processing Systems (NIPS)* (1991), 471–479.

41. Heitjan, D. F. Ignorability and coarse data: Some biomedical examples. *Biometrics,* 1099–1109 (1993).

42. Hora, S. C. Aleatory and Epistemic Uncertainty in Probability Elicitation with an Example from Hazardous Waste Management. *Reliability Engineering & System Safety* **54,** 217–223 (1996).

43. Hüllermeier, E. Learning from Imprecise and Fuzzy Observations: Data Disambiguation through Generalized Loss Minimization. *International Journal of Approximate Reasoning* **55,** 1519–1534 (2014).

44. Hüllermeier, E. & Beringer, J. Learning from Ambiguously Labeled Examples. *Intelligent Data Analysis* **10,** 419–439 (2006).

45. Hüllermeier, E. & Cheng, W. *Superset Learning Based on Generalized Loss Minimization* in *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML)* (2015), 260–275.

46. Hüllermeier, E. & Vanderlooy, S. Combining predictions in pairwise classification: An optimal adaptive voting strategy and its relation to weighted voting. *Pattern Recognition* **43,** 128–142 (2010).

47. James, G., Witten, D., Hastie, T. & Tibshirani, R. *An introduction to statistical learning* (Springer, 2013).

48. Joachims, T. *Transductive Inference for Text Classification using Support Vector Machines* in *Proceedings of the Sixteenth International Conference on Machine Learning (ICML 1999)* (1999), 200–209.

49. Joshi, A. J., Porikli, F. & Papanikolopoulos, N. *Coverage optimized active learning for k-NN classifiers* in *Proceedings of the 2012 IEEE International Conference on Robotics and Automation (ICRA)* (2012).

50. Kasabov, N. & Pang, S. *Transductive support vector machines and applications in bioinformatics for promoter recognition* in *Proceedings of the 2003 International Conference on Neural networks and Signal Processing (ICNNSP)* **1** (2003), 1–6.

51. Kendall, A. & Gal, Y. *What Uncertainties do We Need in Bayesian Deep Learning for Computer Vision?* in *Proceedings of the Thirty-first Annual Conference on Neural Information Processing Systems (NIPS)* (2017), 5580–5590.

52. Konczak, K. & Lang, J. *Voting Procedures with Incomplete Preferences* in *Proceedings of the IJCAI 2005 Multidisciplinary Workshop on Advances in Preference Handling* **20** (2005).

53. Kull, M. & Flach, P. *Reliability maps: a tool to enhance probability estimates and improve classification accuracy* in *Proceedings of the 2014 European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)* (2014), 18–33.

54. Lagacherie, P., Cazemier, D. R., Martin-Clouaire, R. & Wassenaar, T. A spatial approach using imprecise soil data for modelling crop yields over vast areas. *Agriculture, Ecosystems & Environment* **81,** 5–16 (2000).

55. Lewis, D. D. & Gale, W. A. *A Sequential Algorithm for Training Text Classifiers* in *Proceedings of the 17th annual International SIGIR Conference on Research and Development in Information Retrieval (SIGIR)* (Springer, 1994), 3–12.

56. Liu, L.-P. & Dietterich, T. G. *A Conditional Multinomial Mixture Model for superset label learning* in *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS)* (2012), 548–556.

57. Mamitsuka, N. A. H. *et al. Query Learning Strategies Using Boosting and Bagging* in *Proceedings of the Fifteenth International Conference on Machine Learning (ICML)* (1998), 1–9.

58. Masson, M.-H., Destercke, S. & Denoeux, T. Modelling and predicting partial orders from pairwise belief functions. *Soft Computing* **20,** 939–950 (2016).

59. McDonald, J., Stoddard, O. & Walton, D. On using interval response data in experimental economics. *Journal of Behavioral and Experimental Economics* **72,** 9–16 (2018).

60. Melville, P., Saar-Tsechansky, M., Provost, F. & Mooney, R. *Active Feature-Value Acquisition for Classifier Induction* in *Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM)* (2004), 483–486.

61. Melville, P., Saar-Tsechansky, M., Provost, F. & Mooney, R. *An Expected Utility Approach to Active Feature-Value Acquisition* in *Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM 2005)* (2005), 745–748.

62. Menard, S. *Applied Logistic Regression Analysis* (Sage, 2002).

63. Mitchell, T. M. *Version Spaces: A Candidate Elimination Approach to Rule Learning* in *Proceedings of the 5th International Joint Conference on Artificial Intelligence (IJCAI)* (1977), 305–310.

64. Moulin, H. *et al. Handbook of Computational Social Choice* (Cambridge University Press, 2016).

65. Myung, I. J. Tutorial on maximum likelihood estimation. *Journal of Mathematical Psychology* **47,** 90–100 (2003).

66. Ng, A. Y. & Jordan, M. I. *On Discriminative vs. Generative classifiers: A comparison of logistic regression and naive Bayes* in *Proceedings of the 15th International Conference on Neural Information Processing Systems (NIPS)* **2** (2002), 841–848.

67. Nigam, K. & McCallum, A. *Pool-based active learning for text classification* in *Proceeding of the 1998 Conference on Automated Learning and Discovery (CONALD)* (1998).

68. Nocedal, J. & Wright, S. *Numerical Optimization* (Springer New York, 2006).

69. Pang, S. & Kasabov, N. *Inductive vs transductive inference, global vs local models: SVM, TSVM, and SVMT for gene expression classification problems* in *Proceedings of 2004 IEEE International Joint Conference on Neural Networks (IJCNN)* **2** (IEEE, 2004), 1197–1202.

70. Papadopoulos, H., Gammerman, A. & Vovk, V. Reliable diagnosis of acute abdominal pain with conformal prediction. *Engineering Intelligent Systems* **17,** 127 (2009).

71. Patil, G. & Taillie, C. Multiple Indicators, Partially Ordered Sets, and Linear Extensions: Multi-criterion Ranking and Prioritization. *Environmental and Ecological Statistics* **11,** 199–228 (2004).

72. Philip, E & Elizabeth, W. Sequential Quadratic Programming Methods. *UCSD Department of Mathematics Technical Report NA-10-03* (2010).

73. Quinlan, J. R. Induction of decision trees. *Machine learning* **1,** 81–106 (1986).

74. Raman-Sundström, M. A pedagogical history of compactness. *The American Mathematical Monthly* **122,** 619–635 (2015).

75. Rennie, J. D. Regularized Logistic Regression is Strictly Convex. *Technical report, MIT* (2005).

76. Rodríguez, J. J. & Maudes, J. Boosting recombined weak classifiers. *Pattern Recognition Letters* **29,** 1049–1059 (2008).

77. Russell, S. J. & Norvig, P. *Artificial intelligence: A modern approach* (Pearson Education Asia Ltd., 2016).

78. Safavian, S. R. & Landgrebe, D. A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics* **21,** 660–674 (1991).

79. Senge, R. *et al.* Reliable Classification: Learning Classifiers that Distinguish Aleatoric and Epistemic Uncertainty. *Information Sciences* **255,** 16–29 (2014).

80. Settles, B. Active Learning Literature Survey. *Technical Report, University of Wisconsin, Madison* **52,** 11 (2010).

81. Settles, B. & Craven, M. *An Analysis of Active Learning Strategies for Sequence Labeling Tasks* in *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2008), 1070–1079.

82. Settles, B., Craven, M. & Ray, S. *Multiple-instance active learning* in *Proceedings of the 20th International Conference on Neural Information Processing Systems (NIPS)* (2007), 1289–1296.

83. Seung, H. S., Opper, M. & Sompolinsky, H. *Query by committee* in *Proceedings of the fifth Annual Workshop on Computational Learning theory* (1992), 287–294.

84. Shafer, G. & Vovk, V. A Tutorial on Conformal Prediction. *Journal of Machine Learning Research* **9,** 371–421 (2008).

85. Sharma, M. & Bilgic, M. Evidence-based Uncertainty Sampling for Active Learning. *Data Mining and Knowledge Discovery* **31,** 164–202 (2017).

86. Tehrani, A. F., Cheng, W., Dembczyński, K. & Hüllermeier, E. Learning monotone nonlinear models using the Choquet integral. *Machine Learning* **89,** 183–211 (2012).

87. Troffaes, M. C. Decision Making under Uncertainty using Imprecise Probabilities. *International Journal of Approximate Reasoning* **45,** 17–29 (2007).

88. Utkin, L. V. & Augustin, T. Decision making under incomplete data using the imprecise Dirichlet model. *International Journal of Approximate Reasoning* **44,** 322–338 (2007).

89. Vapnik, V. N. An overview of statistical learning theory. *IEEE Transactions on Neural Networks* **10,** 988–999 (1999).

90. Vapnik, V. N. *Estimation of dependences based on empirical data* (Springer-Verlag New York, 1982).

91. Vapnik, V. N. *Principles of risk minimization for learning theory* in *Proceedings of the 4th International Conference on Neural Information Processing Systems (NIPS)* (Morgan Kaufmann Publishers Inc., 1991), 831–838.

92. Vapnik, V. N. *Statistical Learning Theory* (Wiley, New York, 1998).

93. Walker, S. H. & Duncan, D. B. Estimation of the probability of an event as a function of several independent variables. *Biometrika* **54,** 167–179 (1967).

94. Walley, P. & Moral, S. Upper Probabilities based only on the Likelihood Function. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **61,** 831–847 (1999).

95. Wiencierz, A. & Cattaneo, M. *On the Validity of Minimin and Minimax Methods for Support Vector Regression with Interval Data* in *Proceedings of the 9th International Symposium on Imprecise Probability: Theories and Applications (ISIPTA)* (2015), 325–332.

96. Wu, T.-F., Lin, C.-J. & Weng, R. C. Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research* **5,** 975–1005 (2004).

97. Wu, X. *et al.* Top 10 algorithms in data mining. *Knowledge and Information Systems* **14,** 1–37 (2008).

98. Xia, L. & Conitzer, V. Determining Possible and Necessary Winners under Common Voting Rules given Partial Orders. *Journal of Artificial Intelligence Research* **41,** 25–67 (2011).

99. Xu, P., Davoine, F., Zha, H. & Denoeux, T. Evidential calibration of binary SVM classifiers. *International Journal of Approximate Reasoning* **72,** 55–70 (2016).

100. Yang, F. & Vozila, P. *Semi-supervised Chinese Word Segmentation using Partial-label Learning with Conditional Random Fields* in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2014), 90–98.

101. Yang, G., Destercke, S. & Masson, M.-H. *Nested Dichotomies with probability sets for multi-class classification* in *Proceedings of the Twenty-first European Conference on Artificial Intelligence (ECAI)* (2014), 363–368.

102. Yang, G., Destercke, S. & Masson, M.-H. The Costs of Indeterminacy: How to Determine Them? *IEEE Transactions on Cybernetics* **47,** 4316–4327 (2017).

103. Zaffalon, M. The Naive Credal Classifier. *Journal of Statistical Planning and Inference* **105,** 5–21 (2002).

104. Zaffalon, M., Corani, G. & Mauá, D. Evaluating credal classifiers by utility-discounted predictive accuracy. *International Journal of Approximate Reasoning* **53,** 1282–1301 (2012).