École Doctorale ED488 Sciences, Ingénierie, Santé

Learning from Imbalanced Data: Application to Bank Fraud Detection

---

Apprentissage dans un Contexte Déséquilibré: Application à la Détection de Fraude Bancaire

Thèse préparée par **Guillaume Metzler**
au sein de l'**Université Jean Monnet de Saint-Étienne**
pour obtenir le grade de :

Docteur de l'Université de Lyon
Spécialité : **Informatique**

Univ Lyon, UJM-Saint-Etienne, CNRS, Institut d'Optique Graduate School,
Laboratoire Hubert Curien UMR 5516, F-42023, SAINT-ETIENNE, France.
Blitz Business Services, 38090, VILLEFONTAINE, France.

Thèse soutenue publiquement le **25 septembre 2019** devant le jury composé de :

Marianne CLAUSEL	Professeure, Université de Lorraine	Rapporteure
Marc TOMMASI	Professeur, Université de Lille	Rapporteur
Yves GRANDVALET	Professeur, Université de Technologie de Compiègne	Examinateur
Élisa FROMONT	Professeure, Université de Rennes I	Co-encadrante
Amaury HABRARD	Professeur, Université de Saint-Étienne	Co-encadrant
Marc SEBBAN	Professeur, Université de Saint-Étienne	Directeur
Xavier BADICHE	Président Directeur Général de Blitz Business Services	Membre invité
Brahim BELKASMI	Ingénieur R&D, Blitz Business Services	Membre invité

Remerciements

Je tiens tout d'abord à remercier Marc Sebban mon directeur de thèse ainsi que Elisa Fromont et Amaury Habrard pour avoir accepté de diriger cette thèse, ainsi que Xavier, de l'entreprise Blitz, pour le financement de cette thèse.
Je voulais également remercier les rapporteurs Marc Tommasi et Marianne Clausel pour leur travail de relecture du manuscrit et pour leurs retours. Je les remercie aussi, ainsi que Yves Grandvalet, d'avoir accepté d'évaluer mon travail en acceptant l'invitation pour faire partie de mon jury de thèse.

Cette thèse a marqué un tournant dans ma vie en terme de formation, ou comment "quitter" les mathématiques pour se diriger vers l'informatique théorique ! Les débuts furent difficiles, très difficiles mais vous avez toujours cru que cela pouvait aboutir et me motiver en ce sens. Vous avez finalement eu raison car la fin contraste totalement avec le début de thèse. Je ne peux que vous remercier pour cela, pour m'avoir poussé à ne jamais baisser les bras. Elisa, tu as étais une '"seconde mère" comme tu aimes à le dire à tes doctorants, tu m'as toujours remonté le moral et a toujours gardé le sourire avec moi et en toutes circonstances, tu fais preuve d'une joie et d'un optimisme qui pousse à ne pas se décourager et multiplier les efforts. Je remercie également Amaury avec qui les échanges techniques furent toujours aussi intéressants qu'enrichissants. Tu as toujours étais là pour me guider d'un point de vue technique et me diriger vers des branches du domaine que je ne connaissais. Finalement, Marc, comme Elisa qui fait toujours preuve d'optimisme même si je pense qu'il s'est demandé ce que l'on allait bien pouvoir faire de moi au cours des deux premières années de thèse. Au moment où j'écris ces quelques lignes, tu continues de croire en moi et en mes capacités en me proposant des débouchées dont ce fameux post-doc que je vais pouvoir effectuer au sein de votre équipe. Tout comme Amaury, tu fais preuve d'une culture très large dans le domaine et beaucoup d'idées émergent de ton esprit à chaque réunion permettant ainsi de développer de nouveaux axes de recherches. J'espère que nous aurons toujours l'occasion de travailler ensemble par la suite et ce même après mon départ du laboratoire. Et enfin ... bah on va finir par faire ce marathon ensemble ! Cette année c'est la bonne ! Je remercie également Rémi (Bob) pour les moments d'échanges que nous avons eu ensemble concernant les travaux de recherches, tu as su te montrer disponible pour moi. Merci également de m'avoir donné la possibilité de travailler avec Kevin, une collaboration qui a porté ses fruits et qui m'a permis de reprendre beaucoup de plaisir au travail de recherche.

Mais la thèse, ce n'est pas que de la recherche, c'est aussi un moment d'épanouissement où l'on rencontre beaucoup de gens comme ses collègues doctorants qui sont dans la même "misère" que toi mais avec qui tu partages d'excellents moments (surtout en conférence !). Mention spéciale à Rémi (Alice) et Kevin avec qui j'ai eu l'occasion de collaborer pour la rédaction de plusieurs articles et avec qui j'espère rester en contact en dépit du départ im-

minent de certains. Nous avons su nous montrer complémentaires, vous avez su combler mes lacunes et mon goût pour le “code” en effectuant cette partie et en me laissant m'occuper de la partie “théorique”. Finalement ce mariage a bien fonctionné et a montré à quel point le travail d'équipe porte ses fruits. Je n'oublie bien sûr pas les autres doctorants comme Jordan, avec qui j'ai pu échanger des difficultés de la thèse CIFRE et la conciliation entre l'entreprise et le laboratoire de recherche. Tanguy qui pose toujours des questions intéressantes et dont les réponses ne sont pas nécessairement triviales. Julien qui m'a marqué par sa discrétion mais aussi par son humour, qui, disons-le, est parfois particulier mais que nous avons le plaisir de voir pendant notre semaine passé au Japon avec Kevin.

Mais surtout ... que serait notre bureau de doctorants sans ... Léo ! The PhD Student que je ne pourrai jamais oublier, un humour totalement bizarre mais une patience d'ange et toujours d'une grande aide quand je suis confronté à des problèmes informatiques ... même triviaux. Je crois que tu ne cesseras de me surprendre à chacune de tes découvertes de la vie :) Je dois également te remercier pour tes précieuses relectures qui ont permis d'améliorer l'esthétisme du manuscrit mais aussi de corriger des fautes persistantes. Merci à toi aussi Valentina pour les moments d'échanges que nous avons pu avoir et j'espère que nous aurons l'occasion de nous croiser à San Francisco. Et la liste ne s'arrête pas là, je ne vous oublie pas : Thomas, Jules, Rémi (oui encore un autre) qui ont rendu les soirées doctorantes mémorables.

Il y a également nos fameux “stagiaires” que nous avons vu passer dans notre bureau. Ils sont su se démarquer par leur humour ... parfois ... très spécial disons-le, n'est-ce pas Paul ! Mais aussi par leur connaissance comme Fabien. Et j'en n'oublie bien d'autres ...

Mais cette thèse était également l'occasion de ... courir ! Je remercie tous mes collègues de course à pied du laboratoire : Ugo, Pierre-Louis, Rémi, Manu, Christophe, Brice, Thomas, J-P, Mathias, Damien, ...

Merci également à toute l'équipe de l'entreprise Blitz pour leur accueil. Je suis content d'avoir fait la connaissance de Benjamin et Jonathan pour que nous puissions courir ensemble les midis (d'ailleurs ... il y a toujours une Sainté-Lyon à faire les gars). Ils ont depuis quitté le nid et j'espère qu'ils s'épanouissent dans leur travail respectif. Marc ... alors comment dire ... tu sais que l'Alsace n'est plus allemande depuis longtemps quand même ! Tu étais un peu le “Paul de l'entreprise” qui racontait des anecdotes à propos de ses séances de Pilat assez drôles x). Je remercier également mon encadrant Brahim pour le temps qu'il a su m'accorder même si cela n'était pas toujours évident. Hichem et Brahim pour les nombreuses requêtes SQL et les différentes demandes concernant les bases de données. Simon et Xavier pour m'avoir recruté pour cette thèse. Merci aussi à Zakia pour l'organisation d'évènements et de repas pour les salariés de l'entreprise, cela a toujours permis de ressouder les liens et d'apporter de la bonne humeur dans l'équipe !

Je tiens à remercier mes parents qui ont toujours cru en moi. Ils ont su me soutenir

quelque soit le chemin emprunté et ont toujours fait en sorte que les moments passés avec eux soient reposants. Bon ok ... ils y avaient quelques pics de tensions par moment mais c'est normal.

Maman, tu as su toujours me pousser pour que j'aille jusqu'au bout pour mes études ... aujourd'hui je crois que c'est le cas, je ne peux pas aller plus loin. Cela n'était pas forcément des plus agréables mais tu avais raison de le faire.

Papa, tu étais plutôt la force tranquille et tu m'as aidé dans les différentes étapes de la vie qui m'ont permis d'en arriver là. Merci beaucoup à tous les deux. Je remercie plus généralement toute ma famille pour m'avoir soutenu tout au long de mon parcours.

Enfin, je tiens à remercie la personne qui m'a supporté pendant ces quatre années de thèse et ce, malgré les épreuves : Mathieu. Cela fait maintenant presque cinq ans que nous partageons notre vie ensemble et tu n'as jamais cessé de croire que je parviendrai au bout de cette thèse. Elle nous a fait connaître beaucoup de difficultés, elle a empiété sur nos vacances mais tu as su accepté cela afin que je puisse réussir cette dernière étape de mes études. Tu t'es montré patient quand il s'agissait de supporter l'ascenseur émotionnel dans lequel je me trouvais, les soirées et éventuelles nuits au cours desquelles je t'ai laissé seul. Tu as été, tu es et tu resteras un vrai pilier pour moi et c'est un vrai bonheur que de partager mon quotidien avec toi. Merci du fond du coeur.

Table of Contents

List of Figures

List of Tables

Introduction

Machine Learning is a subfield of artificial intelligence at the frontier of computer science and applied mathematics (statistics and as optimization). This discipline also partially overlaps *Data Science* as it is based on collecting data which are analyzed and studied in order to extract the substantial information that can be used for application at hand. Depending on that application, the nature of the data can be multiple: it can consists of images, videos, raw data, categorical data, trees, graphs, times series, etc.

Once the data are collected, and often completed and cleaned, they can be used for several machine learning tasks such as *regression* when it comes, e.g. to predict the price of a share or the price of a house according to its characteristics. They can also be exploited for *classification* tasks when, e.g. one aims to discriminate a spam from a ham when receiving an email, identify if a transaction is fraudulent or genuine, detect anomalies in a medical examination such as a blood test. In both cases, the data are labeled using a variable y. When we have a genuine transaction or a ham email, the example is usually labeled -1 (also called negative example) while it is labeled 1 (positive example) when it is an object of interest like a spam or a fraudulent transaction. When such a labeled information is used in a Machine Learning algorithm, we talk about supervised learning, and unsupervised learning otherwise.

Driven by the application of fraud detection, we will focus, in this document, on binary classification tasks. Our objective will be to learn a classifier h, also called an hypothesis, using a collection of labeled data $\{\mathbf{x}_i, y_i\}_{i=1}^m$ to classify new instances, where $\mathbf{x}_i$ is a set of descriptors.

This CIFRE thesis has been carried out in the context of a collaboration with the Blitz SA company working in the field of bank fraud detection. When a customer pays for his purchases in a store, he/she can use different payment medium, including checks. Blitz offers its clients (mainly companies of the mass distribution) a way to secure checks transactions by detecting so-called *fraudulent payments* which can be of two types: the use of a false check (a check not produced by the bank) or an unpaid check, i.e. the use of check for which there is not enough money on the bank account of the customer. Blitz only provides an advice and the final decision is always made by the client.

The difficulty of this task is two fold: the first one is the huge amount of transactions the company has to deal with per year as well as the response time of the algorithm which must be of the order of ten milliseconds. Second, a fraud constitutes an extremely rare event: it

represents less than 0.4% of the transactions while they cover 1% of the mass distribution turnover.

This thesis is therefore part of the so-called imbalanced learning field, in other words, where the class of interest is much less represented than the negative class. In such a context, most classification algorithms, mainly based on the minimization of the error rate, lead to the trivial solution of predicting all examples as negative. Thus, due to the imbalance in the data, a way to achieve high performances (i.e. 99.6% in our context) is to predict all instances as genuine, that is obviously totally inacceptable because one misses all the fraudulent transactions.

More generally, learning in an unbalanced context is an important issue for the Machine Learning community because of its wide range of applications (e.g. in bank, insurance and tax fraud, in health, etc.) but also because of the intrinsic methodological difficulty of this type of problem. There are many techniques to address this type of problem: learning a good representation of the data in which the classification task is easier; assigning costs to different classes to provide algorithms with a way to focus on the minority class; using sampling methods to reduce the imbalance as done in Blitz.

Our objective in this thesis is to propose new strategies to tackle the issue of learning from highly imbalanced data and apply them on fraud detection problem. We aim to propose new algorithms which perform well in this setting and which improve the model used by the Blitz company. We also dedicate a part of this thesis to theoretical contributions of learning from imbalanced data by proposing new ways to optimize performance measures that are suited for this context.

Our contributions are essentially divided into two parts. The first one focuses on the development of so-called geometric algorithms, which determine a better representation for discriminating transactions, by learning new features with which the classification is easier to do. The second one uses theoretically founded costs assign to each class in order to improve well suited performance measures for this context using a theoretical analysis. We also provide a concrete application of that type of methods in order to improve the check fraud detection process of the Blitz company by improving the benefits of its retailers.

Context of the thesis. This thesis was carried out in both in the Data Intelligence team of the Laboratoire Hubert Curien UMR CNRS 5516, affiliated to the University Jean Monnet in Saint-Etienne and the University of Lyon, and in the Blitz Business Services company in Villefontaine, France. This work has been conducted under a CIFRE contract funded by the ANRT (National Agency of Research and Technology).

About the Blitz Business Services company. The Blitz Business Services company is located in Villefontaine, France. Its main activity is the check fraud detection and its costumers are the mass distribution. It is also working on the optimization of the passage at the payment office or the granting of payment facilities, such as the agreement of a payment in several installments on online sales sites for example. Finally, Blitz develops an activity

which consists of optimizing the use of reminder letters and calls when a report has been drawn up for a person who does not have a valid ticket in public transports.

Contributions

This manuscript presents different contributions from both theoretical and practical aspects to address the imbalanced learning issues. The main body of this thesis contains the contributions of this thesis. Extensions and extended experiments are given in Appendix for the sake readability.

This first part is composed of two chapters:

Chapter 1. In the first chapter, we introduce the field of Statistical Learning starting from the learning of a model to its capacity of performing well on unseen data. It presents the definition and notations useful for the rest of the document. We end this chapter by introducing the problem of learning from imbalanced data and by illustrating the behavior some Machine Learning algorithms when they face imbalanced data.

Chapter 2. The second chapter focuses on the state of the art techniques used to tackle the imbalanced learning issue. It introduces generalities on both anomaly and fraud detection in the highly imbalanced learning scenarios. It is also dedicated to the presentation of relevant performance metrics used in such a context.

In the second part of this thesis, we present two contributions based on geometrical approaches to deal with imbalanced data. Both of them aim to learn the influence regions around the positive examples, using (i) a metric learning approach in Chapter 3 and (ii) an adjusted distance to positives in Chapter 4.

Chapter 3. Based on the assumption that frauds are locally closed to each other[1], our first contribution proposed a novel strategy based on Support Vector Data Description, a SVM variant which aims to encompass a set of examples. We propose to modify the standard formulation in order to create the largest area around each positive instances. The surface of this area is optimized using a metric learning strategy and leads to ellipsoids in the feature space for which size and orientation are controlled by a regularization term. By solving the dual formulation of our optimization problem, we show that, compared to most metric learning techniques, we have the semi definite positiveness of the learned metric for free. We also derive generalization guarantees on the proposed method using the uniform stability framework and show its effectiveness, in terms of F-Measure, compared to standard machine learning algorithms combined with sampling methods. On the private dataset of Blitz, we show that it possible to control either the recall and the precision of the model.

[1]Even though frauds are rare, a given fraudster usually acts quickly with the same strategy, leading to a few positive examples in the same region of the feature space.

Chapter 4. The positive examples of the training set have an important role when it comes to detect new positives at test time. When we are working with distance-based algorithms, such as the k-Nearest Neighbor algorithm, the distance to these positives is key. Instead of applying a linear transformation of the data as it is usually done, we show that modifying the distance of a new query only to positive examples is enough to significantly improve the k-Nearest Neighbor algorithm. Our method has also the ability to control to false positive rate and the false negative one. Furthermore, we show, on several datasets, that the performance of the proposed algorithm can also be improved when it is combined with sampling strategies and achieves at least better results than a metric learning approach or other modification of the k-NN.

The third and last part of this thesis focuses on two contributions on cost-sensitive methods. In Chapter 5, we optimize the F-measure and provide bounds on its optimality. Chapter 6 uses this approach for a practical application: the optimization of the retailers benefits.

Chapter 5. The use of specific measure such as the F-measure has been shown to be more relevant in the context of imbalanced learning. But its optimization remains a challenging task due to its non linearity and non convexity. However, due to one property of the F-measure, we are able to make link between its optimization and cost-sensitive learning, i.e. learning by assigning different costs to each class. Using this fact, we propose new bounds on the difference of F-measures between two classifiers. Thus, we are able to give bounds on the optimal reachable F-measure. We also provide a geometric interpretation of the derived bounds in the form of asymmetric cones. This last point is used to derive an iterative algorithm which chooses the right costs to assign to each class in order to optimize the F-measure. We show that the presented method is effective or even better than its competitors and only requires a small number of iterations. From a theoretical aspect, we also show that the proposed bounds are much tighter than the ones proposed in the literature

Chapter 6. Cost-sensitive approaches are relevant to address the imbalanced classification issue. In this chapter, we propose to combine these approaches to tree-based algorithms for the real application of the Blitz Company: the optimization of the retailers benefits. The amount of money of a transaction is a relevant information for the classification task and more attention should be paid on transactions with a large amount of money, something that previous models did not take into account. Furthermore, a miss-classification on that type of transactions has heavier consequences than the one with a small amount. We propose a cost matrix and, thus, a loss function which aims to optimize the retailers benefits. We also propose different tree-based methods to optimize such a loss. The experiments made in this chapter show that it is possible to drastically increase the retailers benefits by the use of a cost-sensitive approach and that the proposed gradient tree boosting algorithm achieves better results than random forest algorithms.

Appendix A We provide an extension to the multi-class setting of the work presented in Chapter 5. This extension includes the proofs and extended experiments.

Notations

All the notations used in this thesis are summarized in Table 1.

Notation	Description
m	Number of training instances
d	Dimension of the feature space
x	Scalar
$\mathbf{x}$	Vector or a set of vectors
x_i	Entry i in vector $\mathbf{x}$
$\mathbf{x}_i$	Vector i in the set of Vectors $\mathbf{x}$
$\mathbf{X}$	Matrix
$\mathbf{X}_{ij}$	Entry in row i and column j of matrix $\mathbf{X}$
y	Label
$\mathcal{X}, \mathcal{Y}, \mathcal{H}$	Input space, Output space, Hypothesis space
$\mathcal{D} = \mathcal{X} \times \mathcal{Y}$	Distribution of the data
$\mathbb{R}, \mathbb{R}_+$	Sets of real and non negative real numbers
$\mathbb{S}^+$	Set of Positive Semi Definite matrices
S	Sample
$\mathcal{R}, \mathcal{R}_S$	True Risk, Empirical Risk over S
$f(\cdot)$	Function
$\|\cdot\|$	Norm
$\|\cdot\|$	Absolute value
$[\cdot]_+$	Hinge loss function
$\langle \cdot, \cdot \rangle$	Dot product between vectors
$\mathbb{E}[\cdot]$	Expectation
$\Pr[\cdot]$	Probability
$\ell(\cdot)$	Loss function
$\mathcal{A}$	Algorithm

Table 1: Notations.

Part I

Background

Chapter 1

Preliminaries

Abstract

In Machine Learning, the data and the loss function used to train a given model are fundamental. In this chapter, we introduce Machine Learning by presenting the different types of problems we meet throughout this thesis. First, we introduce the definition of Risk, the quantity we aim to minimize in most of the optimization problems. We will see that it is possible to provide theoretical guarantees on the performance of the learned model using statistical tools such as concentration inequalities.

The second part of this introduction is dedicated to the loss functions and algorithms. We detail the presentation of the latter as we will meet them all along this thesis. This chapter ends by studying the behavior of some of them when one class becomes over-represented compared to the other one, i.e. when the dataset becomes imbalanced.

1.1 Statistical Learning Theory

In this first section, we introduce the concept of empirical risk minimization, the key point of the supervised learning algorithms in Machine Learning.

Let us consider $\mathcal{X}$ the input space of our data, also called the feature space and $\mathcal{Y}$ the output space.[1] The input space, i.e. the features used to describe the data can be of different types: it can consist of continuous or discrete descriptors. In this thesis, we will focus on continuous features, i.e. $\mathcal{X} \subset \mathbb{R}^d$. The output space can also vary from a task to another. It can be discrete, real, or even a structured data. The nature of the output gives important information on the nature of the problem and leads the user to choose the appropriate tools.

- When $\mathcal{Y} \subset \mathbb{R}$ or $\mathcal{Y} = [0, 1]$, i.e. when the output is a real value, the aim is to learn either a score, a probability or to estimate a quantity as it is usually done in *regression tasks*.

[1] Note that there may be no output space, i.e. $\mathcal{Y} = \emptyset$ as in *unsupervised learning*.

- When $\mathcal{Y} = \{-1, +1\}$ or $\{0, 1\}$, i.e. the output is binary, we aim to classify the data in two categories, also called *classes*. This type of tasks is called *binary classification*. Note that the output can take more than two values, i.e. $\mathcal{Y} = \{1, \dots, q\}$ where q is the number of classes. In this case, we talk about *multi-class classification*.

To solve these different tasks, we use an algorithm $\mathcal{A}$ which aims at learning a function $h : \mathcal{X} \to \mathcal{Y}$ called *hypothesis*. From a practical point of view, the joint distribution of the data $\mathcal{D} = \mathcal{X} \times \mathcal{Y}$ is unknown and we only have access to a collection of *sample points*, i.e. a set of observations $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$, drawn i.i.d. from $\mathcal{D}$. The goal is then to learn a function h using the sample S which is "good enough" on the given training sample but which is also able to perform well on a new sample $(\mathbf{x}, y)$.

In the next section, we explain how the function h is learned and the meaning of performance of an algorithm.

1.1.1 Empirical Risk Minimization

In order to solve a given task, the algorithm $\mathcal{A}$ needs a criterion to optimize, also called a *loss function* usually denoted by $\ell(h(\cdot), \cdot) : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$. In a regression task, it can be based on the difference between the estimated value, $h(\mathbf{x})$, and the true value y, for example $\ell(h(\mathbf{x}), y) = |h(\mathbf{x}) - y|$ or $(h(\mathbf{x}) - y)^2$. In a classification task, the most natural way is to consider the number of errors made by the classifier h. The considered loss function is then defined by $\ell(h(\mathbf{x}), y) = \mathbb{1}_{\{h(\mathbf{x}) \neq y\}}$ and is called *0-1* loss. However, because of its non convexity and non differentiability, minimizing this loss is known to be NP-hard. That is why, we usually change the *0-1* loss by a surrogate function (see Section 1.2).

Whatever h, we need to minimize its expected value over the distribution $\mathcal{D}$, leading to the *True Risk*.

Definition 1.1 (True Risk)**.** *Let $\ell : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ a loss function and $\mathcal{D}$ the distribution of the data. The True Risk $\mathcal{R}$ of an hypothesis h is defined by:*

$$\mathcal{R}^{\ell}(h) = \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} [\ell(h(\mathbf{x}), y)].$$

However, we can not compute this quantity in practice because the joint distribution $\mathcal{D}$ of the data is unknown. We rather minimize the *Empirical Risk*, the average error over the sample S.

Definition 1.2 (Empirical Risk)**.** *Let $S = \{(\mathbf{x}_i, y)\}_{i=1}^m$ a collection of m examples drawn i.i.d. from $\mathcal{D}$ and $\ell : \mathcal{X} \times \mathcal{Y} :\to \mathbb{R}$. The Empirical Risk risk of an hypothesis h is defined by:*

$$\mathcal{R}_S^{\ell}(h) = \frac{1}{m} \sum_{i=1}^{m} \ell(h(\mathbf{x}_i), y_i).$$

We aim to minimize the empirical risk with the hope that the sample S is representative of the unknown distribution. At the first glance, minimizing the empirical risk might be then equivalent to minimizing the true/generalization risk. But this is not sufficient in general, and the risk is to tend to an overfitting phenomenon.

It shows that it is important to fix some constraints on the hypothesis we can learn to avoid such a situation. We present below a non exhaustive list of ways to constrain the learned hypothesis.

Empirical risk minimization. The most straightforward solution is to fix in advance the hypothesis space $\mathcal{H}$, called a *Set of Hypotheses* or *Class Function*. Then, given this set of hypotheses $\mathcal{H}$, a sample S and a loss function ℓ, we are looking for the hypothesis $h_S^\star$ as the solution of:

$$h_S^\star = \underset{h \in \mathcal{H}}{\operatorname{argmin}} \, \mathcal{R}_S^\ell(h).$$

If the size of $\mathcal{H}$ is small enough, we can avoid the situation previously described. However, without any side information on the data, it is hard to choose the appropriate set of hypotheses for the task at hand.

Structural risk minimization. [*See Section 4.1 of Vapnik (1995)*] The idea here is to choose a growing sequence of sets of hypotheses $\{\mathcal{H}_n\}_{n \in \mathbb{N}}$, i.e. the complexity or the number of hypotheses is growing with n, and to choose the hypothesis $h^\star$ solution of the following minimization problem:

$$h_S^\star = \underset{h \in \mathcal{H}_n, \, n \in \mathbb{N}}{\operatorname{argmin}} \, \mathcal{R}_S^\ell(h) + pen(n, d).$$

Compared to the previous formulation, a *penalty term* is added to the empirical risk. This term measures the complexity/size of the set of hypotheses $\mathcal{H}_n$ and is a growing function of n (note that it also depends on the dimension of the data). The aim is then to learn the best hypothesis in the smallest set $\mathcal{H}_n$. With such a formulation, we are able to find a good trade-off between the empirical risk minimization and the complexity of the learned model, i.e. find a hypothesis that is able to generalize well. But choosing a sequence of sets of hypotheses remains hard in practice.

Regularized risk minimization. The preferred solution in practice, because easier to implement, is to choose a set of hypotheses $\mathcal{H}$ sufficiently large enough and to add a constraint on the parameters θ of the learned model. The added constraint, $\lambda \|\theta\|$, is called a *regularization term* and is associated to a *regularization constant* λ.

$$h_S^\star = \underset{h \in \mathcal{H}}{\operatorname{argmin}} \, \mathcal{R}_S^\ell(h) + \lambda \|\theta\| .$$

With the parameter λ, we are able to indirectly control the size or complexity of the set

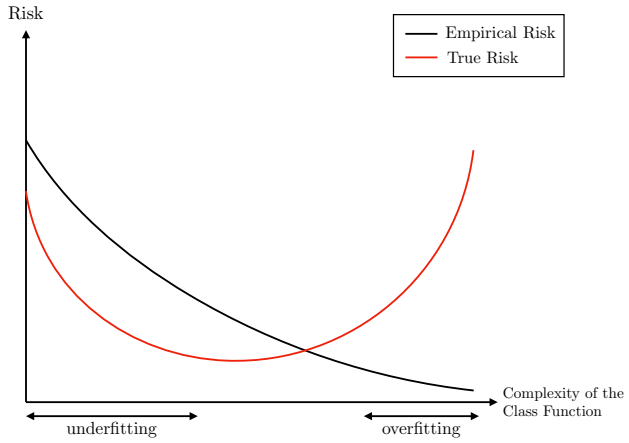

Figure 1.1: Evolution of the Empirical and True Risks according to the complexity of the set of hypotheses.

of hypotheses by constraining the norm of the parameters of the model for instance[2].

In this last expression, the smaller the value of λ is, the more importance we give on fitting well the data.

In the same way, the greater λ, the more importance we give on the complexity of the learned model. The errors made by the algorithm become insignificant compared to the control of the magnitude of the parameters θ. The risk is that, when $\lambda \to \infty$ we lead to an *under-fitting* phenomenon without a good generalization capacity.

The aim is to learn a model with the appropriate value of λ, i.e. to find a good trade-off between the minimization of the empirical risk and the complexity of the hypothesis as depicted in Figure 1.1. To *tune* the value of λ, we typically use a *k-fold cross-validation* on the training set: we separate the dataset into $k-$folds of regular size. We learn a hypothesis/model using $k-1$ folds and test its performance on the remaining one. The process is repeated $k-$times and we keep the hypothesis/model which achieves the lowest empirical risk in average.

Even if we can empirically assess the performance of a model on a test set, it is also important to provide *generalization guarantees* in order to bound the performance of the learned algorithm regarding its behavior at training time. The next section is dedicated to this point and presents generalization bounds based on the complexity of the set of hypotheses or on some properties on the loss function.

[2]In the rest of the thesis such a regularization constant will be called *hyper-parameter* instead of parameter to distinguish them from the parameters of the learned hypothesis h.

1.1.2 Generalization Bounds

A generalization bound has the following general form and is often referred to a *Probably Approximately Correct* (PAC) bound (Valiant, 1984):

$$Pr\left(|\mathcal{R}(\cdot) - \mathcal{R}_S(\cdot)| \geq \varepsilon\right) \leq \delta, \tag{1.1}$$

where $\varepsilon \geq 0$ and $\delta \in [0, 1]$. δ is an upper bound on the probability that the true risk deviates from at least ε from its empirical value. Looking at this bound, an immediate consequence is that the lower the value of ε, the greater (i.e. the closer to 1) the value of δ. Furthermore, the closer from 0 both δ and ε are, the more reliable our estimation is. Such a bound is usually derived using concentration inequalities such as Hoeffding (Hoeffding, 1963) or McDiarmid (McDiarmid, 1989) inequalities. The bound (1.1) can be rewritten as a probabilistic *bound of convergence* from the empirical estimate to its mean (Vapnik and Chervonenkis, 1982; Valiant, 1984)

$$\left|\mathcal{R}^\ell(\cdot) - \mathcal{R}^\ell_S(\cdot)\right| \leq \varepsilon(\delta, m),$$

where δ is the rate of confidence on the given bound and ε is a function of the rate of confidence and a decreasing function on the number of training examples. Such a bound holds with probability at least $1 - \delta$ and the aim is to build a function $\varepsilon(\cdot)$ with a high rate of convergence.
In the following, we will see how we can build such generalization bounds and that their convergence rate is often $\sim \mathcal{O}(\ln(m)/\sqrt{m})$ or even $\mathcal{O}(1/\sqrt{m})$. Some of the presented frameworks will be used in our contribution in Chapter 3.

Uniform Deviation

As stated before, generalization bounds are based on the convergence of empirical quantities to their means (Vapnik and Chervonenkis, 1971) and resort on the law of large numbers. Intuitively, the larger our training set is, the closer the empirical risk will be to the expected value, i.e.:

$$\lim_{m\to\infty} \frac{1}{m} \sum_{i=1}^{m} \ell(y_i, h(\mathbf{x}_i)) = \mathbb{E}[\ell(y, h(\mathbf{x}))].$$

The bound is said to be *uniform* because it holds for any hypothesis $h \in \mathcal{H}$.
To get such a bound when the size of $\mathcal{H}$ is finite, we need to estimate the probability of the following event:

$$\left\{\sup_{h\in\mathcal{H}} \left|\mathcal{R}^\ell(h) - \mathcal{R}^\ell_S(h)\right| \geq \varepsilon\right\}.$$

The probability of such an event can be estimated using the Hoeffding inequality and the *Union bound*. It leads to the following result.

Theorem 1.1. *[Uniform generalization bound] Let $\mathcal{H}$ be a set of hypotheses of finite size, S a training sample of size m drawn i.i.d. from $\mathcal{D}$, ℓ a loss function which takes its values in $[0,1]$ and $\delta > 0$. Then, for any $h \in \mathcal{H}$, with probability at least $1-\delta$, we have:*

$$\mathcal{R}^{\ell}(h) \leq \mathcal{R}^{\ell}_{S}(h) + \sqrt{\frac{\ln|\mathcal{H}| + \ln(2/\delta)}{2m}}.$$

When the space $\mathcal{H}$ is not finite, e.g. in the case of a set of linear classifiers in $\mathbb{R}^d$ where the parameters which define a linear separator belong to $\mathbb{R}^{d+1}$, we need another method to measure the size or complexity of the set of hypotheses. A way to measure the complexity is the VC-dimension, noted $VC(\mathcal{H})$ and introduced by Vapnik and Chervonenkis (1971).

Definition 1.3. *Let $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^m$ denotes a set of m examples and let $\mathcal{H}$ be a set of hypotheses. The VC-dimension of $\mathcal{H}$, $VC(\mathcal{H})$ is the largest value of m such that, for any labelization of the set $\mathbf{X}$, there is $h \in \mathcal{H}$ which correctly classifies all the instances.*

Roughly speaking, $VC(\mathcal{H})$ measures the risk of overtfitting the data whatever the labels assigned. It allows to bound the generalization performance given a set of hypotheses without any information on the used algorithm.

Theorem 1.2. *[Uniform generalization bound with VC-dimension] Let $\mathcal{H}$ be a set of hypotheses of VC-dimension $VC(\mathcal{H})$, $S = \{\mathbf{x}_i, y_i\}_{i=1}^m$ a training sample of size m drawn i.i.d. from $\mathcal{D}$ and ℓ a loss function which takes its values in $[0,1]$ Then, for any hypothesis $h \in \mathcal{H}$ and for all $\delta > 0$, with probability at least $1-\delta$, we have*

$$\mathcal{R}^{\ell}(h) \leq \mathcal{R}^{\ell}_{S}(h) + \sqrt{\frac{VC(\mathcal{H})\left(\ln\left(\frac{2m}{VC(\mathcal{H})}\right)+1\right) + \ln(4/\delta)}{m}}. \tag{1.2}$$

Note that the bound uniformly converges to zero when the size of the training set grows. The convergence is faster when the VC-dimension of $\mathcal{H}$ is small. However, the rate of convergence of this bound is $\mathcal{O}(\ln(m)/\sqrt{m})$, so the rate of convergence is lower than previously stated in Theorem 1.1. Thus, one drawback of such a complexity measure is to decrease the rate of convergence. Furthermore the VC-dimension remains hard to compute in general except for simple families of hypotheses, such that linear classifiers, or decision trees (Asian et al., 2009).

To overcome these limitations, we can resort to the *Rademacher complexity*.

Rademacher complexity

The Rademacher complexity (Bartlett and Mendelson, 2003; Koltchinskii and Panchenko, 2000) has been also introduced to measure the complexity of a set of hypotheses. Informally, it measures how the set of hypotheses is able to fit noise in the dataset.

Definition 1.4. *[(Empirical) Rademacher complexity] Let $\mathcal{H}$ be a family of functions and $S = \{\mathbf{x}_i\}_{i=1}^m$ a fixed sample of size m. Then, the empirical Rademacher complexity of $\mathcal{H}$ with respect to S is defined as:*

$$\mathfrak{R}_S(\mathcal{H}) = \mathop{\mathbb{E}}_{\boldsymbol{\sigma}} \left[\sup_{h \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^{m} \sigma_i h(\mathbf{x}_i) \right],$$

where $\sigma = (\sigma_1, \ldots, \sigma_m)$ is a vector of Rademacher random variables, i.e. random variables taking values in $\{-1, +1\}$ both with probability $1/2$.
The Rademacher complexity is the expectation of the above quantity on the distribution $\mathcal{D}$ of the data:

$$\mathfrak{R}_m(\mathcal{H}) = \mathop{\mathbb{E}}_{S \sim \mathcal{D}^m} [\mathfrak{R}_S(\mathcal{H})].$$

Note that a similar measure of complexity has been introduced by Bartlett and Mendelson (2003) using Gaussian variables instead of Rademacher ones.
Using Definition 1.4, we can derive the following generalization bound.

Theorem 1.3. *Let $\mathcal{H}$ be a family of hypotheses, ℓ a loss function taking its values in $[0, 1]$ and S a training set of examples of size m. Then, for all $\delta > 0$ and for all $h \in \mathcal{H}$, with probability at least $1 - \delta$, we have:*

$$\mathcal{R}^\ell(h) \leq \mathcal{R}_S^\ell(h) + 2\mathfrak{R}_m(\mathcal{H}) + \sqrt{\frac{\ln(1/\delta)}{2m}}, \tag{1.3}$$

$$\mathcal{R}^\ell(h) \leq \mathcal{R}_S^\ell(h) + 2\mathfrak{R}_S(\mathcal{H}) + 3\sqrt{\frac{\ln(1/\delta)}{2m}}. \tag{1.4}$$

Even if the first bound remains impossible to compute, the second can be evaluated because it only depends on the dataset. Compared to the bound based on the VC-dimension, we can note that the Rademacher complexity accounts the training examples, and it can lead to a higher convergence rate in $\mathcal{O}(1/\sqrt{m})$.

Despite the validity of the previous bounds, (1.2) to (1.4), for all hypotheses, both remain impossible to compute in some cases and require a huge number of examples to be relevant. Indeed, they do not take into account the learning algorithm $\mathcal{A}$ and its main properties and thus lead to pessimistic bounds. The two following results present generalization guarantees that take into account the learning algorithm $\mathcal{A}$.

Uniform Stability

This framework is more recent (Bousquet and Elisseeff, 2002) and is not directly based on a measure of complexity of the set of hypotheses. It resorts on the concept of *stability*. Roughly speaking, an algorithm is *stable* if its output does not change significantly under a small modification of the training sample. More precisely, we focus on *uniform stability*: we are looking for the greatest modification in the loss function under a small modification of the training sample. A more formal definition is given below.

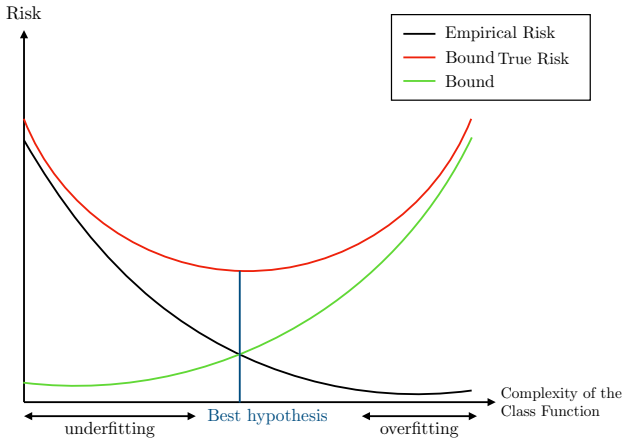

Figure 1.2: Bound on the true risk with with respect to the complexity of the set of hypothesis.

Definition 1.5. *[Definition 6 (Bousquet and Elisseeff, 2002)] A learning algorithm $\mathcal{A}$ has a uniform stability in $\frac{\beta}{m}$ with respect to a loss function ℓ and parameter set θ, with β a positive constant if:*

$$\forall S,\ \forall i,\ 1 \leq i \leq m,\ \sup_{\mathbf{x}} |\ell(\theta_S, \mathbf{x}) - \ell(\theta_{S^i}, \mathbf{x})| \leq \frac{\beta}{m},$$

where S is a learning sample of size m, θ_S the model parameters learned from S, θ_{S^i} the model parameters learned from the sample S^i obtained by replacing the i^{th} example $\mathbf{x}_i$ from S by another example $\mathbf{x}'_i$ independent from S and drawn from $\mathcal{D}$. $\ell(\theta_S, \mathbf{x})$ is the loss suffered at $\mathbf{x}$ [3] *.*

The constant β depends on the properties of the loss function but also on the regularization term of the minimization problem. The property of uniform stability has been shown to hold for a wide range of minimization problems (Bousquet and Elisseeff, 2002). Using the *convexity* of the loss function and the McDiarmid inequality we get the following generalization bound with a rate of convergence in $\mathcal{O}(1/\sqrt{m})$.

Theorem 1.4. *[Theorem 12 (Bousquet and Elisseeff, 2002)] Let $\delta > 0$ and $m > 1$. For any algorithm with uniform stability β/m, using a loss function ℓ bounded by K, with probability at least $1 - \delta$ over the random draw of S we have:*

$$\mathcal{R}^{\ell}(\theta_S) \leq \mathcal{R}^{\ell}_S(\theta_S) + \frac{2\beta}{m} + (4\beta + K)\sqrt{\frac{\ln 1/\delta}{2m}},$$

where $\mathcal{R}(\cdot)$ is the true risk and $\mathcal{R}_S(\cdot)$ its empirical estimate over S.

[3] Note that the authors also provide a definition of stability in which an example is removed. For the sake of convenience, we rather use this definition throughout this thesis. Indeed, we keep the same number of training instances from a set to another.

The previous bound has a global convergence rate in $\mathcal{O}(1/\sqrt{m})$ and depends on two parameters: β the constant of uniform stability of the algorithm $\mathcal{A}$ and K an upper bound on the loss function ℓ.

This generalization bound is more informative in practice and both terms are easy to compute. Moreover, compared to the two previous frameworks, it is consistent with the practice in Machine Learning, i.e. it takes into account the regularization term which is often used in the minimization problems. We will use this framework in Chapter 3.

Algorithmic Robustness

In the previous framework, the authors derive generalization guarantees driven by the fact that a small change in the training set implies a small modification of the loss function value. The robustness framework states that if a test sample is "similar", in a sense that should be defined, to a training one then the test error is close to the training error (Xu and Mannor, 2010, 2012). The definition of robustness of an algorithm $\mathcal{A}$ is itself defined using the notion of *Covering numbers* (A. N. Kolmogorov, 1959).

Definition 1.6. *[Covering number (van der Vaart and Wellner, 1996)] Let $(\mathcal{S}, \rho)$ be a metric space and $\mathcal{U}$ a subset of $\mathcal{S}$. We say that $\hat{\mathcal{U}} \subset \mathcal{U}$ is an $\varepsilon-$cover of $\mathcal{U}$, if for all $u \in \mathcal{U}$, there is $\hat{u} \in \hat{\mathcal{U}}$ such that $\rho(u, \hat{u}) \leq \varepsilon$. Then, the $\varepsilon-$covering number of T is defined by:*

$$\mathcal{N}(\varepsilon, \mathcal{U}, \rho) = \min\{|\hat{\mathcal{U}}| : \hat{\mathcal{U}} \text{ is an } \varepsilon - \text{cover of } \mathcal{U}\}.$$

This definition aims to define a partition of a space $\mathcal{Z}$, on which two examples are said to be similar if they lie on the same partition of the space, i.e. if they share the same characteristics (e.g. the labels in a context of classification). The notion of Covering number is particularly relevant in the context of Machine Learning. We consider that the space $\mathcal{Z}$ is bounded and closed and all the inputs consist of vectors of finite dimension. So the space $\mathcal{Z}$ is compact, thus, its $\varepsilon-$covering number is finite.

Using this definition, the authors proposed a definition of robustness of an algorithm.

Definition 1.7. *[Algorithmic robustness (Xu and Mannor, 2010)] Let S be a training sample of examples drawn i.i.d. from $\mathcal{D}$, ℓ a loss function. An algorithm $\mathcal{A}$ is said to be $(k, \varepsilon(S))-$robust if the space $\mathcal{Z}$ can be partitioned into $k \in \mathbb{N}$ disjoints set denoted $K_i, i = 1, \ldots, k$ such that, two elements that belong in the same set K_i have a distance lower than $\varepsilon(S)$ according to ℓ, i.e.*

$$\forall \mathbf{z} \in S,\ \mathbf{z}, \mathbf{z}' \in K_i \implies \left|\ell(h_S, \mathbf{z}) - \ell(h_S, \mathbf{z}')\right| \leq \varepsilon(S),$$

where h_S is an hypothesis learned from $\mathcal{A}$ with S.

In this definition we directly see that there is a link between the number of sets and the "size" of the sets used for the partition. Note that the authors (see Definition 8 of Xu and Mannor (2010)) also proposed a definition of *pseudo-robustness* in case that the previous condition does not hold for any training sample S.

The definition of robustness leads to another generalization bound which has the same rate of convergence as uniform stability.

Theorem 1.5. *[Theorem 3 (Xu and Mannor, 2010)] Let S be a set of m training examples drawn i.i.d. from $\mathcal{Z}$ and ℓ a loss function bounded by K. Let also $\mathcal{A}$ be an $(k, \varepsilon(S))-$robust algorithm and h_S the hypothesis learned from $\mathcal{A}$ with S. For all $\delta > 0$, with probability at least $1 - \delta$ we have:*

$$\mathcal{R}^{\ell}(h_S) \leq \mathcal{R}^{\ell}_S(h_S) + \varepsilon(S) + K\sqrt{\frac{2k\ln(2) + 2\ln(1/\delta)}{m}}.$$

Both robustness and uniform stability bounds rely on the deviations of the loss function value between samples. The main difference is that uniform stability measures this deviation by replacing (or removing) an example from the training set S while, in the definition of robustness, this deviation is measured between an example and another lying in the same set in the partition of $\mathcal{Z}$.

In Theorem 1.5, we see that a tradeoff has to be made between k the number of sets used to partition the space and $\varepsilon(S)$ the covering number. he bounds based on robustness can be applied to a larger class of functions than the one based on uniform stability. For instance, the convexity of the loss is not required here. However, having a sufficiently small $\varepsilon(S)$ implies to have a large value of k, thus a precise enough partition of the space. It implies that such bounds are looser than the ones based on stability and require more examples to be tight.

Now that the theoretical context has been presented, we present some fundamental algorithms and loss functions currently used in Machine Learning and which will be used in this thesis.

1.2 Learning Algorithms

In the previous section, the generalization bounds based on uniform deviation and Rademacher complexity have been presented for a loss ℓ function taking its values in $[0, 1]$, typically, the 0-1 loss, in the context of classification. However, as said before, such loss is hard to optimize from an algorithmic point of view due to its non-convexity and non-differentiability. So, we rather use some convex (and sometimes differentiable) relaxation, also called *surrogate losses*, to learn our hypothesis $h(\cdot, \cdot)$, where the two variables are respectively the parameters $\mathbf{w}$ of the model and an input $\mathbf{x}$. Both the 0-1 loss and some of its surrogates are presented in Figure 1.3.

Among these surrogates, the *hinge loss* is used when training a Support Vector Machine (Boser et al., 1992; Vapnik and Cortes, 1995). The hinge loss is used in the context of classification and is defined by:

$$\begin{aligned} \ell(h(\mathbf{x}), y) &= [1 - yh(\mathbf{w}, \mathbf{x})]_+ , \\ &= \max(0, 1 - yh(\mathbf{w}, \mathbf{x})). \end{aligned}$$

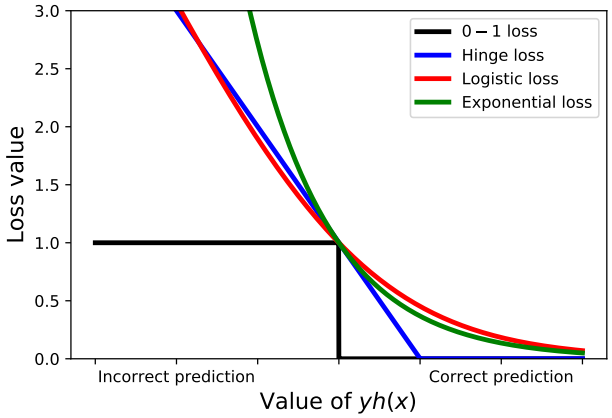

Figure 1.3: Examples of loss functions used for classification tasks: the hinge loss (linked to the SVM algorithm), the logisitic loss (used in for logistic regression) and the exponential loss (used in the context of boosting).

Another well known surrogate loss function is the exponential loss which is widely used in the context of boosting (Friedman et al., 2000):

$$\ell(h(\mathbf{x}), y) \quad = \quad \exp(-yh(\mathbf{x})).$$

Compared to the hinge loss, the exponential loss gives a more important weight on the errors and never equals 0.

Finally the logistic loss, used when training a logistic regression model (Mohri et al., 2012), is defined by:

$$\ell(h(\mathbf{x}), y) \quad = \quad \frac{1}{\ln(2)} \ln\left(1 + \exp(-yh(\mathbf{x}))\right).$$

Note that this definition holds when $y \in \{-1, 1\}$ and $h(\mathbf{x})$ belongs in $\mathbb{R}$ or $[-1, 1]$. There is another definition if we focus on the probability of belonging to a class or an other, i.e. if $y \in \{0, 1\}$ and $h(\mathbf{x}) \in [0, 1]$, such variant (Cox, 1958) is defined by:

$$\ell(h(\mathbf{x}), y) \quad = y \ln\left(1 + \exp(-h(\mathbf{x}))\right)^{-1} + (1 - y) \ln\left(1 - \frac{1}{1 + \exp(-h(\mathbf{x}))}\right).$$

Now we present the different algorithms which will be used in this thesis. The k-Nearest Neighbors algorithm is used in Chapters 3 and 4, a variant of the support vector machine is used in both Chapter 3 and Chapter 5. The logistic regression is used in the experiments of Chapter 5 and the decision-tree based algorithms are used in Chapter 6. Due to the context of this thesis which focus on fraud anomaly detection, all these algorithms are presented in the context of binary classification, i.e. when the output space $\mathcal{Y}$ is $\{-1, 1\}$.

1.2.1 k-Nearest Neighbors

The k-Nearest Neighbors algorithm (Cover and Hart, 1967) is a non parametric method which does not impose any assumption on the underlying distribution. To predict the label of a

new instance $\mathbf{x}'$, it computes the "distances" between the new query $\mathbf{x}'$ and the set of samples $\mathbf{x}_i \in S$. Then, using a selected k value, it looks for the k nearest neighbors of $\mathbf{x}'$ and predicts the label of $\mathbf{x}'$ using a majority vote. An example of the use of the k-NN algorithm is drawn on Figure 1.4. It shows the importance of the choice of the k value to predict the label of a new instance. We see on this example that the new point is closed to points of both classes, that is why its prediction varies according to k. It is "red" when k is equal to 3 or 5. However, if we consider $k = 9$, it is predicted "blue".

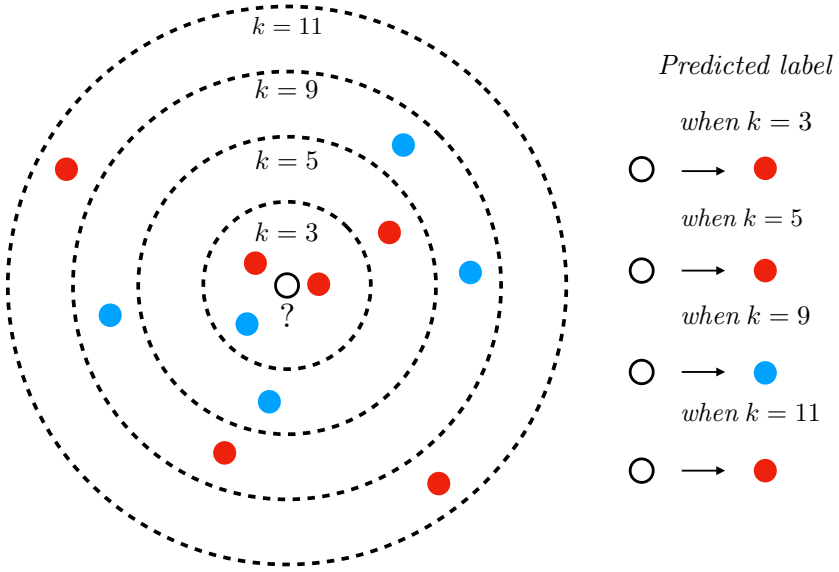

Figure 1.4: Illustration of the k-NN using the euclidean distance. We show that the predicted label depends on the value of k.

For small values of k, for instance when $k = 1$, the algorithm assigns to the new example $\mathbf{x}'$ the same label as the one of its closest data in the training set, i.e. similar examples shall share the same label. Such a rule is the simplest one and it has been shown by Cover and Hart (1967), when m is large enough, that the error rate is no more than twice the bayesian error, i.e. the smallest error we can achieve given the distribution of the data. Conversely, when we increase the value of k we tend to smooth our decision. When k is very large (typically when $k = m$), the label given to a new data is no more than the majority class in the dataset.

It raises one important question: *which value of k shall we take?* In practice, we usually use a cross validation procedure to choose the best k value.

The dimensionality d of the data and the choice of the distance are also important for such an algorithm. Indeed, when we are dealing with high-dimensional data, some features (or attributes) sometimes non informative can have a huge impact in the computation of the

distance. This leads to a classification based on non relevant features and poor performances. It is often referred as *the curse of the dimensionality*.

To overcome this issue, several refinement of the method exists, the most common and the one that will be used in Chapter 4 is a weighted k-Nearest Neighbors Dudani (1976). The idea is simple and consists in assigning weights to all training examples. A standard weight is the use of the inverse of the distance (Liu and Chawla, 2011). By this way, we give less importance to the examples that are far from the tested one. Another possibility is to learn a new representation of the data by projecting the data in a (better) latent space (He and Wang, 2008; Weinberger and Saul, 2009).

One drawback of this algorithm is that it needs to store the entire dataset and to compute all the distances to the training samples to predict the label of a new one. However, we can reduce the computation time and size using approaches based on the triangle inequality (Elkan, 2003), or on structured segmentations (Bentley, 1975) or by selecting the most relevant instances of the training set as in *Condensed Nearest Neighbor* (Hart, 1968).

Let us now detail the Support Vector Machine.

1.2.2 Support Vector Machine

The Support Vector Machine algorithm (SVM) is probably one of the most known and used classification algorithm in Machine Learning (Vapnik and Cortes, 1995) for binary classification.

The SVM algorithm outputs an hypothesis h which returns the label of an example $\mathbf{x}$. This hypothesis is an (affine) hyperplane which separates the space into two spaces. e.g. $\{-1, +1\}$ as follows:

$$h(\mathbf{x}) = sign[\langle \mathbf{w}, \mathbf{x} \rangle + b] = \begin{cases} -1 & \text{if} \quad \langle \mathbf{w}, \mathbf{x} \rangle + b < 0, \\ +1 & \text{if} \quad \langle \mathbf{w}, \mathbf{x} \rangle + b > 0. \end{cases}$$

When the two classes can be perfectly seperated, there are several hyperplanes, i.e. several values of $(\mathbf{w}, b)$, which can well separate the data. The idea developed by Boser et al. (1992) is to choose the one that has the largest margin. In Figure 1.5, the margin γ is defined by the distance between the hyperplanes of equation $\langle \mathbf{w}, \mathbf{x} \rangle + b = \pm\rho$. Note that we can rewrite these equations as $\langle \mathbf{w}, \mathbf{x} \rangle + b = \pm 1$ by a normalization of $(\mathbf{w}, b)$. Let us now consider the points $\mathbf{x}_-$ and $\mathbf{x}_+$ which lie on the two hyperplanes as depicted in Figure 1.5. Each of these vectors can be decomposed as $\mathbf{x} = \mathbf{x}^{\mathbf{w}} + \mathbf{x}^{\mathbf{w}^\perp}$, where $\mathbf{x}^{\mathbf{w}}$ is colinear to $\mathbf{w}$ and $\mathbf{x}^{\mathbf{w}^\perp}$ is orthogonal to $\mathbf{w}$. These remarks lead to the following relations:

$$\begin{aligned} h(\mathbf{x}_+) - h(\mathbf{x}_-) &= 2, \\ \langle \mathbf{w}, \mathbf{x}_+ \rangle + b - (\langle \mathbf{w}, \mathbf{x}_- \rangle + b) &= 2, \end{aligned}$$

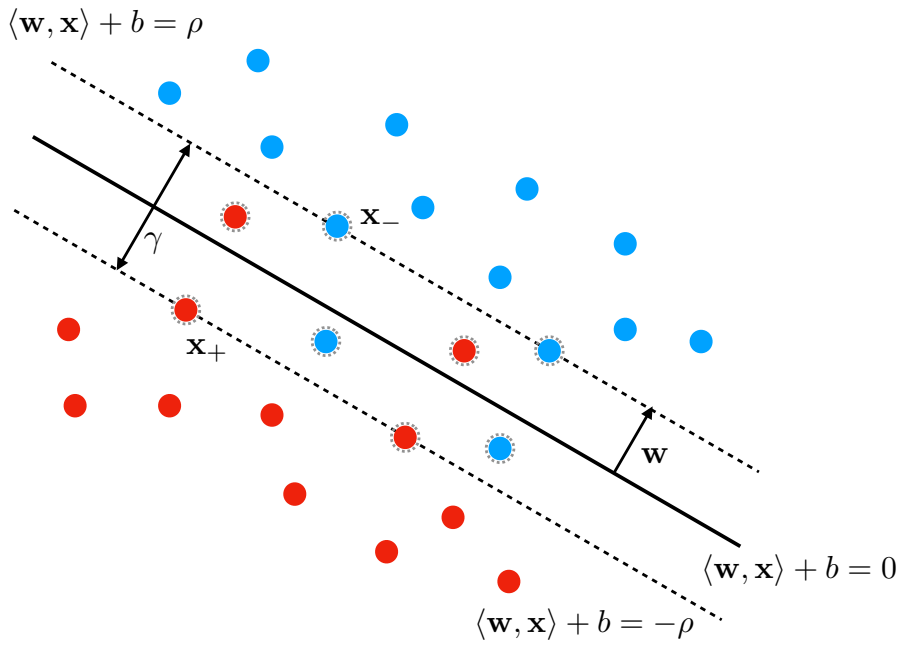

Figure 1.5: Illustration of the SVM algorithm when the problem is not linearly separable. The circled points are the support vectors. The dotted lines represent the hyperplanes of equation $\langle \mathbf{w}, \mathbf{x} \rangle + b = \pm\rho$ and the distance between the two hyperplanes is equal to γ.

$$\begin{aligned} \langle \mathbf{w}, \mathbf{x}_+^{\mathbf{w}} \rangle + \underbrace{\langle \mathbf{w}, \mathbf{x}_+^{\mathbf{w}^\perp} \rangle}_{=0} + b - (\langle \mathbf{w}, \mathbf{x}_-^{\mathbf{w}} \rangle + \underbrace{\langle \mathbf{w}, \mathbf{x}_-^{\mathbf{w}^\perp} \rangle}_{=0} + b) &= 2, \\ \langle \mathbf{w}, \mathbf{x}_+^{\mathbf{w}} \rangle - \langle \mathbf{w}, \mathbf{x}_-^{\mathbf{w}} \rangle &= 2, \\ 2\langle \mathbf{w}, \mathbf{x}_+^{\mathbf{w}} \rangle &= 2. \end{aligned}$$

Therefore, taking the norm on both sides and using the definition of γ leads to:

$$\gamma = 2 \left\| \mathbf{x}_+^{\mathbf{w}} \right\|_2 = \frac{2}{\|\mathbf{w}\|_2}.$$

Thus, maximizing the margin is equivalent to minimizing $\dfrac{2}{\|\mathbf{w}\|}$. The minimization problem is then:

$$\begin{aligned} \min_{(\mathbf{w},b)\in\mathbb{R}^{d+1}} \quad & \frac{1}{2}\|\mathbf{w}\| \\ s.t. \quad & y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1, \quad \text{for all } i = 1, ..., m. \end{aligned}$$

This version of SVM is called the *hard margin SVM*. However, in most of the real cases, the two classes are not linearly separable (see Figure 1.5), and some points violate the constraint. So we need to define a relaxation of the optimization problem in which the errors are taken into account. These errors take the form of a vector of *slack variables* $\boldsymbol{\xi} = (\xi_1, ..., \xi_m)$ and are

included in the optimization problem:

$$\begin{array}{ll} \min\limits_{\xi\in\mathbb{R}^m,\,(\mathbf{w},b)\in\mathbb{R}^{d+1}} & \frac{1}{2}\left\|w\right\|_2 + \frac{C}{m}\sum_{i=1}^m \xi_i \\ s.t. & y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) \geq 1-\xi_i, \quad \text{for all } i = 1, ..., m, \\ & \xi_i \geq 0, \qquad\qquad\qquad\quad \text{for all } i = 1, ..., m. \end{array} \tag{1.5}$$

In this formulation, we have to find a good compromise between the maximization of the margin γ and minimizing the number of errors by tuning the parameter C. If we take the constraints into account, the optimization problem 1.5 can be rewritten as follows:

$$\min_{(\mathbf{w},b)\in\mathbb{R}^{d+1}} \frac{1}{2}\left\|w\right\|_2 + \frac{C}{m}\sum_{i=1}^m \left[1 - y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle) + b\right]_+ .$$

However, such a problem is hard to optimize. When the dimension of the data is high, the complexity is in general $\mathcal{O}(d^3)$ (Chapelle, 2007). Thus, we usually solve what we call the *dual formulation* of such a problem to improve the speed of convergence of the optimization algorithm. We thus prefer to optimize the dual formulation given below:

$$\begin{array}{ll} \max\limits_{\boldsymbol{\alpha}} & -\frac{1}{2}\sum_{i=1}^m\sum_{j=1}^m \alpha_i\alpha_j y_i y_j \mathbf{x}_i\mathbf{x}_j + \sum_{i=1}^m \alpha_i, \\ s.t. & 0 \leq \alpha_i \leq \frac{C}{m} \qquad\qquad\qquad\qquad \text{for all } i = 1, ..., m, \\ & \sum_{i=1}^m y_i\alpha_i = 0. \end{array} \tag{1.6}$$

The vector $\boldsymbol{\alpha}$ is the vector of Lagrangian variables. The way we get the dual problem is fully described by Boyd and Vandenberghe (2004). The reader is also referred to Chapter 3 for a complete example. Note that the dual optimization problem is always a strictly concave problem with respect to the dual variables. Thus, there exists one and only one solution.

If the dual formulation leads to an easier problem to solve when the number of data is small, it does not solve the problem of non linear separability of the data. To tackle this issue, we use what is commonly called the *kernel trick*. Instead of using the standard inner product between two examples, we define a function $K(\cdot,\cdot)$ which takes two vectors as input and returns a real number. Such function $\mathbf{K}$ is called a kernel. We also denote by K its associated matrix, that is (i) symmetric and (ii) positive semi-definite, i.e.:

(i) $\forall\ (\mathbf{x}, \mathbf{x}') \in \mathbb{R}^d \times \mathbb{R}^d$, we have: $K(\mathbf{x}, \mathbf{x}') = K(\mathbf{x}', \mathbf{x})$,

(ii) $\forall\ (\mathbf{x}_i, \mathbf{x}_j) \in \mathbb{R}^d \times \mathbb{R}^d$ and $\forall\ \mathbf{c} \in \mathbb{R}^d$, we have: $c^T\mathbf{K}c = \sum_{i=1}^m\sum_{j=1}^m c_i c_j K(\mathbf{x}_i, \mathbf{x}_j) \geq 0$.

These properties on the function (or matrix) K play a key role and lead to the following result due to Mercer (Mercer, 1909).

Theorem 1.6. *[Mercer Theorem] Let $\mathcal{X}$ be a compact subset of $\mathbb{R}^d$ and let K be a continuous symmetric positive semi-definite function, i.e. a kernel. Then, there exists an orthonormal*

basis of functions $(\Phi_j)_{j\in\mathbb{N}}$ *and a sequence* $(\lambda_j)_{j\in\mathbb{N}}$, *where* $\lambda_j \geq 0$ *for all* j, *such that:*

$$K(\mathbf{x}, \mathbf{x}') = \sum_{j=1}^{\infty} \lambda_j \Phi_j(\mathbf{x}) \Phi_j(\mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle,$$

where $\Phi(\mathbf{x}) = (\sqrt{\lambda_1}\Phi_1(\mathbf{x}), ..., \sqrt{\lambda_j}\Phi_j(\mathbf{x}), ...)$ *is the representation of* $\mathbf{x}$ *in a new space.*

Thus, what we call the kernel trick is that there is no need to explicitly know the transformation Φ, the knowledge of K is enough. Furthermore, it gives the possibility to project the data in a higher dimensional space, called the latent space (possibly of infinite dimension) in which the classes are linearly separable.

Introducing the kernel K, in the optimization problem 1.6 leads to the following dual formulation:

$$\begin{array}{lll} \max\limits_{\boldsymbol{\alpha}} & -\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) + \sum_{i=1}^{m}\alpha_i, & \\ s.t. & 0 \leq \alpha_i \leq \dfrac{C}{m}, & \text{for all } i = 1, ..., m, \\ & \sum_{i=1}^{m}\alpha_i = 0, & \end{array}$$

There exists a large number of kernel functions (see Genton (2002) for a more complete list of kernels) among which the most used are:

- **Linear Kernel**: $K(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle$, which is the standard inner product.
- **Gaussian Kernel**: $K(\mathbf{x}, \mathbf{x}') = \exp -\dfrac{\|\mathbf{x} - \mathbf{x}'\|}{2\sigma^2}$, where σ is an hyper-parameter that needs to be tuned. It controls the importance given to the similarity between two instances. The bigger this value is the less importance we give to two similar examples and the more uniform is the role of each examples in the dataset.

However, regarding the dataset, there is no solution to determine which kernel is the most appropriate to solve the classification task.

The next method belongs to the category of the *Class Probability Estimator* algorithms. The objective is no more to predict a discrete label of a given example but its probability to be in a certain class.

1.2.3 Logistic Regression

The Logistic Regression model, also called the *logit model* has been introduced in the middle of the 20^{th} century (Cox, 1958) but the use of *logit* models dates back to the end of the 19^{th} century (Cramer, 2003).

These models are close to SVM in the way that they also aim to learn a linear separator between two classes. However, they differ since are used to estimate the probability that an

example belongs to a given class, for instance the positive class: $\eta = Pr(Y = 1 \mid X)$. More precisely, the logistic regression aims to compute the logarithm of the *odds*, i.e. the ratio of the probabilities. Then we estimate the log of this ratio using a linear model:

$$\ln\left(\frac{Pr(y=1 \mid x)}{Pr(y=-1 \mid x)}\right) = h(\mathbf{x}) = b + \langle \mathbf{x}, \mathbf{w} \rangle.$$

Thus, once the parameters of the model are learned, we can compute the probability of being in class 1:

$$Pr(y = 1 \mid x) = \frac{\exp(h(\mathbf{x}))}{1 + \exp(h(\mathbf{x}))}.$$

Such function is called a *logistic function* and takes its values in $[0, 1]$. An example $\mathbf{x}_i$ is (usually) predicted in class 1 if $Pr(y = 1 \mid x) > 0.5$, i.e. if $h(\mathbf{x}) > 0$. Given a task and an objective, we can choose to modify this threshold as we will see in the next chapter.

To estimate the parameters of the model, we maximize the likelihood of the data $\mathfrak{L}(\mathbf{w}, S)$, where S is a set of m examples.

$$\begin{aligned}
\mathfrak{L}(\mathbf{w}, S) &= \prod_{i=1}^{m} Pr(Y = y_i \mid X = x_i), \\
&= \prod_{i=1, y_i=1}^{m} Pr(Y = y_i \mid X = x_i) \times \prod_{i=1, y_i=-1}^{m} Pr(Y = y_i \mid X = x_i), \\
&= \prod_{i=1}^{m} \left(\frac{1}{1 + \exp(-h(\mathbf{x}_i))}\right)^{\frac{1}{2}(1+y_i)} \times \left(\frac{1}{1 + \exp(h(\mathbf{x}_i))}\right)^{\frac{1}{2}(1-y_i)}, \\
&= \prod_{i=1}^{m} \frac{1}{1 + \exp(-y_i h(\mathbf{x}_i))}.
\end{aligned}$$

Note that we usually prefer to minimize the negative log-likelihood of the data. By doing so, we find the logistic loss function introduced before. Therefore, the optimization problem becomes:

$$\min_{\mathbf{w}, b \in \mathbb{R}^{d+1}} \frac{1}{m} \sum_{i=1}^{m} \ln\left(1 + \exp\left(-y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b)\right)\right).$$

We divide the loss by a factor m in order to be consistent with the notion of empirical risk previously defined. In order to avoid over-fitting, a regularization term of the form $\lambda \|\mathbf{w}\|$ is sometimes used. Thus, the optimization problem can be rewritten:

$$\min_{\mathbf{w}, b \in \mathbb{R}^{d+1}} -\frac{1}{m} \sum_{i=1}^{m} \ln\left(1 + \exp\left(-y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b)\right)\right) + \lambda \|\mathbf{w}\|.$$

If SVMs and Logistic Regression models are similar geometrically speaking (they both learn a hyperplane) and present similar regularized empirical risk, a closer look in the loss functions shows that the Logistic Regression is sensitive to outliers in the data compared to SVMs and thus can lead to completely different solutions. Indeed, the loss function associated

to the Logistic Regression exponentially penalizes the errors.

The last algorithm we will present can be used for both regression and classification tasks. It can return the probability of being of a given class, like the Logistic Regression or directly assign the label as the SVM or the k-NN.

1.2.4 Decision Trees

Decision trees were introduced by Quinlan (1986) but the currently used version of Classification and Regression Trees algorithm was well introduced by Breiman et al. (1984).

Decision trees consist of a series of rules that are successively applied to the dataset in order to separate the data into two or more groups. Here, we will only focus on binary decision trees, i.e. when a decision rule separates the data set in exactly two different sets.

The nature of the tree depends on the output space $\mathcal{Y}$:

- when $\mathcal{Y} \subset \mathbb{R}$, we talk about regression tree,
- when $\mathcal{Y} = \{-1, 1\}$, we talk about classification tree.

An example of classification tree is provided in Figure 1.6 with a toy dataset, in which we aim to separate the two classes (male and female) using two descriptors (age and height) using a set of rules. The initial dataset is composed of 4 females and 3 males. The first decision rule: $Age \leq 25$ divides the initial dataset into two groups, one composed of two females and the other one of 3 males and 2 females. The group on the left is *pure* and only contains females, so we do not focus anymore on this one. Now, we apply a second decision rule: $Height \leq 170$ with which we are able to separate the remaining males and females.

Such an algorithm is able to (non linearly) separate a complex dataset when using a large number of decision rules. To see how the decision rules are chosen, we define a criterion to optimize.

For this purpose, we need two tools, a *metric* which evaluates the quality of a node and a measure of improvement after a split, called the *gain* (Safavian and Landgrebe, 1991; Rokach and Maimon, 2005).

The kind of used metrics depends on the type of tree we are dealing with. A list of such metrics are provided by Rokach and Maimon (2005) among which:

- the *Variance*, used for regression trees:

$$\frac{1}{n}\sum_{i=1}^{m_N}(y_i - \bar{y})^2,$$

  where m_N denotes the number of examples in the node N and $\bar{y}$ the average value of y_i in the node.

Toy dataset

Age	Height	Sex
20	175	F
32	180	M
40	175	M
28	172	M
22	165	F
40	169	F
70	170	F

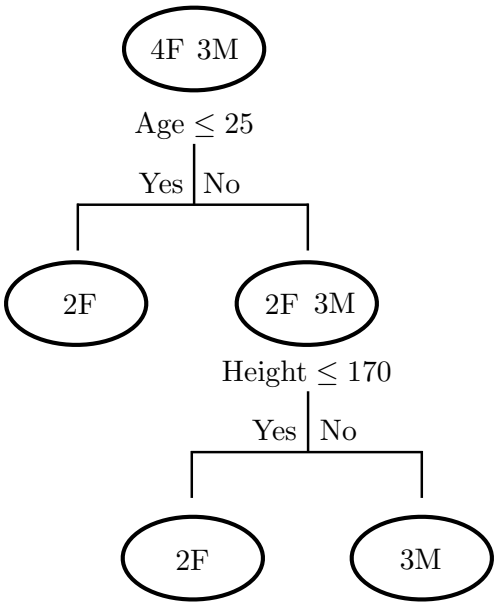

Figure 1.6: An example of classification tree.

- the *Gini impurity* used in classification tree. It measures the impurity of a node by computing the proportion of each classes present in the node. For instance, in binary classification [4], the Gini impurity G_N of a node N is defined by:

$$G_N = \sum_{j=1,-1} p_j(1-p_j) = 2p_1(1-p_1), \tag{1.7}$$

  where p_j denotes the proportion of examples being in class j.

In the binary setting, the Gini impurity is a real value which belongs in $[0, 0.5]$. A value of 0 means that the node is pure, i.e. it contains only examples from one class. A value of 0.5 means that the node contains the same number of examples from both classes.

Let us now illustrate this notion. In the previous example (Figure 1.6) the Gini impurity of the root is $G_{\text{root}} = 2 \times \frac{4}{7}\left(1 - \frac{4}{7}\right) = \frac{24}{49}$ while it is equal to 0 and $\frac{12}{25}$ respectively on each node after the first split. We have previously said that the node on the left was *pure* because it contains only examples from one class. As this node is pure, its Gini Impurity can not be improved.

The next step consists in choosing the optimal rule to split the dataset into two nodes. This rule is chosen in order to minimize the Gini impurity at the end of the tree. For this purpose, we define the *Gini gain* Γ as follows:

$$\Gamma = G_{\text{root}} - \left(\frac{|N_L|}{|N_L + N_R|}G_{N_L} + \frac{|N_R|}{|N_L + N_R|}G_{N_R}\right),$$

[4] The definition can be extended to L classes and the Gini impurety is then defined by $G_N = \sum_{j=1}^{L} p_j(1-p_j)$, where p_j is the proportion of examples of class j in the node N.

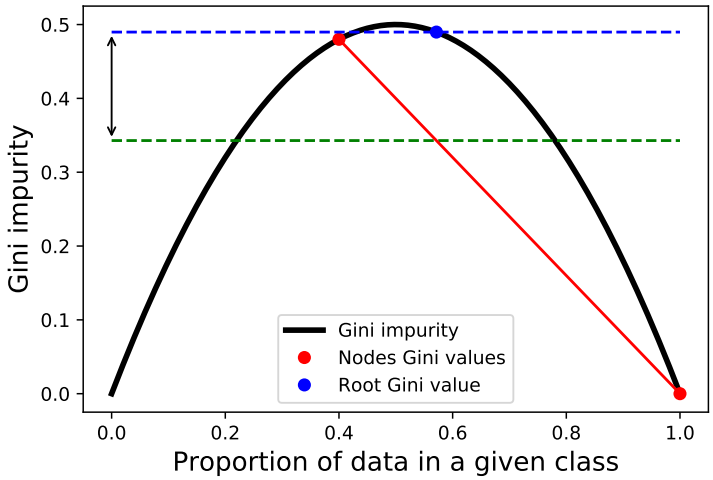

Figure 1.7: Illustration of the Gini Gain using the example presented in Figure 1.6. The dotted green line represent the weighted sum of the Gini impurity of both nodes. The arrow between the two dashed lines represents the Gini gain Γ.

where G_{N_L} and G_{N_R} denote the Gini impurity of the node on the left, respectively on the right.

Figure 1.7 illustrates the use of the Gini impurity as a metric to build our decision tree on the given example. The arrow between the two dashed lines represents the Gini gain Γ. On this figure, we also see that the Gini function is concave. This concavity ensures the positiveness of the gain by the Jensen Inequality (Jensen, 1906) so that each split leads to a lower classification error. Furthermore, at each step, we choose the feature and its corresponding value which maximizes the gain Γ. The decision rule is then applied and the node is separated into two different nodes until getting pure leaves.

In practice, it is always possible to lead to such perfect leaves. However, building such trees might tend to overfitting and bad performance in generalization. To overcome this issue, we usually use a pruning strategy which can be controlled by parameters:

- the *size/depth* of the tree,
- the size of a node: minimum number of examples required in the node to make a new split,
- the size of a leaf: minimum number of examples in both leaves after a split,
- a threshold on the gain: the minimum value of gain required to make a new split.

In the next section, we introduce the problematic of learning from imbalanced data.

1.2.5 Behaviour toward Imbalanced Dataset

The previously presented algorithms work well when the two classes are balanced. However, in some scenarios, the class of interest, such as a rare disease or fraudulent transactions, is less represented. The over-represented class is called the *majority class* and the class of interest, is called the *minority class*. We denote by p the number of examples in the minority class and n the number of examples in the majority class. Two different metrics are usually used to measure the imbalance in a dataset (or distribution): the *Imbalance Ratio (I.R.)* (García et al., 2012) which is the number of positives over the number of negatives. Sometimes we also use the rate of minority class examples to measure the disequilibrium in the data.

$$\textit{Imbalance Ratio} = \frac{n}{p} \quad \text{and} \quad \textit{Minority Rate} = \frac{p}{n+p}.$$

In the following, we focus on two of the previously presented algorithms: a linear SVM and the k-Nearest Neighbors algorithm. We aim to show how they perform on imbalanced data. For this purpose, we consider the Sonar dataset[5], on which a PCA was applied to get a representation in a 2-dimensional space. The true rate of minority examples in the dataset is close to 46%. Thus, to reach a rate of 50% we have simply removed examples from the majority class. After that, in order to decrease the rate of minority examples to respectively 35% and 20%, we remove examples from the minority class.

We represent the obtained classifiers on Figure 1.8. We note first that, for both algorithms, the classification error is decreasing at test time as the imbalance ratio increases. At a first glance, minimizing the classification error (or a surrogate loss function) seems to be well adapted to adress this issue. However if we focus on the left of Figure 1.8, we note that the linear SVM is predicting all the examples as negative, represented by the blue area, when the rate of minority examples is less than 35%. So even if the error rate is low, the algorithm is no more able to capture the minority class.

This comment also holds for the k-NN algorithm represented on the right of Figure 1.8. It struggles to detect examples from the minority class. As the number of minority examples decreases, the decision boundaries around the positives are drastically decreasing, so the probability of being predicted positive becomes lower and lower.

[5]This dataset is available from the UCI repository: `https://archive.ics.uci.edu/ml/index.php`

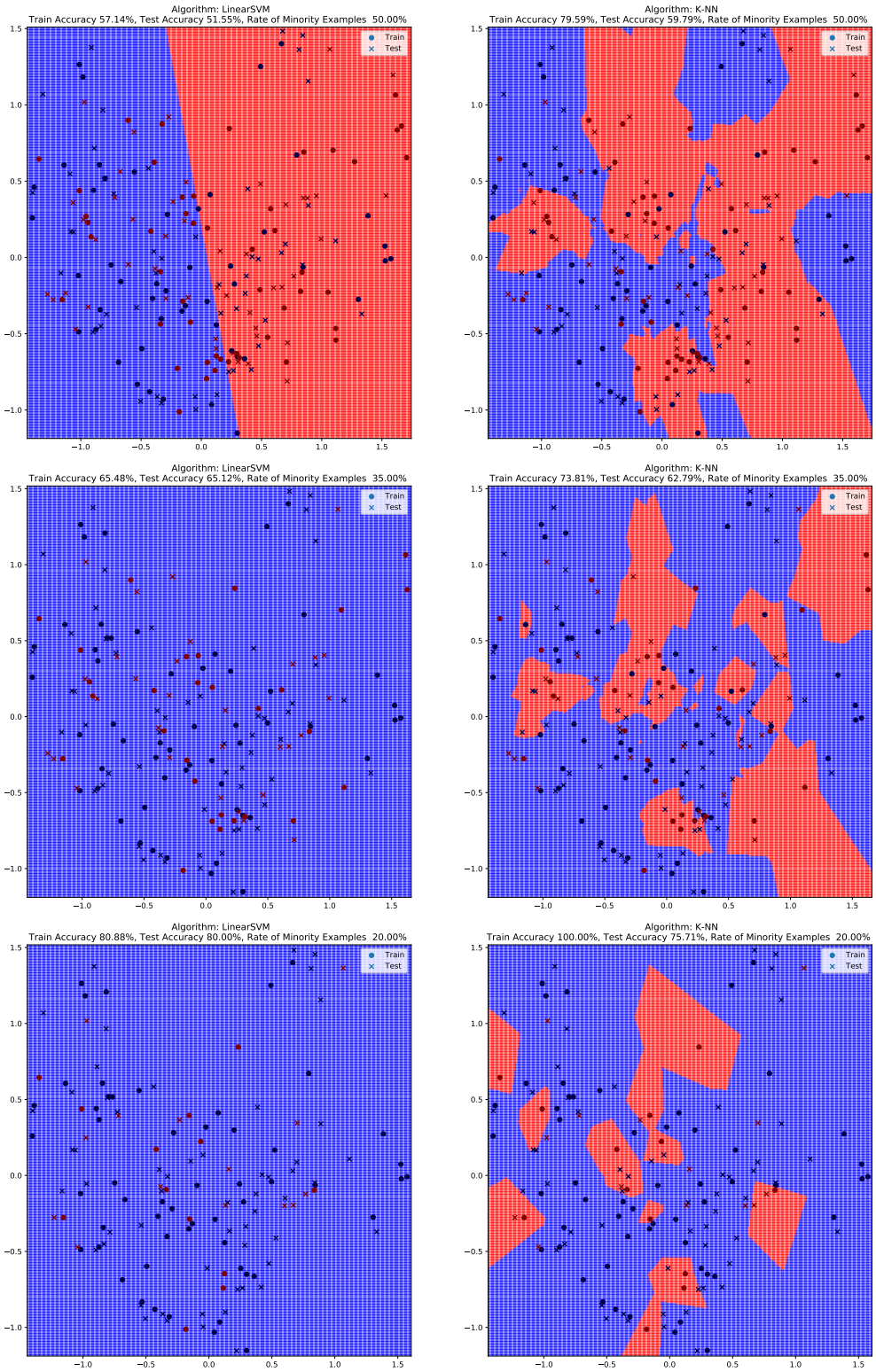

Figure 1.8: Illustration of the decision surfaces of two standard classification algorithms: an SVM on the left and the $k-$NN algorithm, with $k = 3$, on the right. We draw the decision areas with respect to the rate of minority examples in the dataset: 50%, 35% and 20% respectively. The minority class is represented in red and the majority one in blue.

Chapter 2

Learning from Imbalanced Data

Abstract

Learning from Imbalanced data is one of the most problematic issue when one aims to deal with fraud and anomaly detection. Learning a classifier, in such a context leads to neglect the minority class, which is the class of interest for the user. There are several complementary solutions usable at different stages of the learning process, to address this problem. They can be used during a pre-process with the main goal to balance the data with sampling strategies. They can be used during the optimization of the model by resorting to alternative loss functions, better suited than the classification error.

2.1 Introduction

In the previous chapter, we have introduced the problem of learning from imbalanced data with toy examples. Imbalance often occurs when we aim to detect *frauds* or *anomalies*, i.e. rare events in a collection of data.

In this first section, we define the problem of fraud and anomaly detection in the binary classification setting[1]. We also present several applications where this kind of problem occurs.

2.1.1 Anomaly versus Fraud Detection

In the literature (Aggarwal, 2017), an anomaly is often considered as an outlier, i.e. an instance which behavior deviates from the normality. From a statistical point of view, anomalies can be seen as events that rarely occur, i.e. they lie in the tail of the distribution of the data (Aggarwal, 2017; Phua et al., 2010) or far from the main clusters (Chawla and Gionis, 2013).

While anomalies can be considered as abnormal cases, it is often harder to detect a fraud in the due to their similarity to a genuine behavior. For instance, in bank fraud detection, it is harder to say if a transaction is a genuine one or not by just looking at the attributes that

[1]Note that the problem of imbalance also exists in a regression setting (Torgo et al., 2013, 2014; Branco et al., 2016).

describe the transaction. We need experts to understand why the transaction is fraudulent or even to be sure that a transaction is so. Indeed, a fraudster tries to behave as a normal customer and to hide himself behind any other normal behavior. This is the reason why frauds can occurs anywhere in the distribution of the data, unlike anomalies.

In bank fraud detection, the fraud can sometimes be easily detected when the fraudster uses a stolen credit card several times in a small window of time. In this case, the fraud can be described as a repeated and unusual event. But it is not always that simple.

As for anomalies, frauds are also rare events. But in some industrial contexts, such as bank fraud detection, the rate of frauds can be close to 0.1% as for the data handled by the Blitz company. The problem becomes much more complex than detecting anomalies.

Finally, unlike anomalies, frauds evolve with time. This is called *concept drift* (Wang et al., 2003; Dal Pozzolo et al., 2015). The fraudsters change their behavior to not be caught and to adapt to the ongoing detection technique Therefore, the learned model has to be able to detect a change in the behavior of the fraudsters.

To sum up, a fraud can be described as:

- an unusual event that occurs a very small amount of times,
- it can "uniformly" appears in the feature space,
- its characteristics are context dependent,
- it evolves with time.

2.1.2 A Wide Range of Applications in Fraud Detection

Imbalanced learning and fraud detection are present in many industrial problems.

A first example is the network security when dealing with intrusion detection (Kou et al., 2004). The principle is to detect unauthorized persons who are (or at least try to be) connected to a secured server in order to have access to confidential information. It can lead to a leak of sensitive data, theft of industrial processes or suppression of important information for a company. Such intrusions can be detected by monitoring the different computers which have access to a given server. In the same area, the detection of fraudulent emails is also a challenging task in fraud detection (Nizamani et al., 2014). The fraudsters tend to lure a person by sending an email that has a similar format than the one sent by the administrations.

Credit card fraud detection is maybe one of the most active domain, due to the amount of money this type of fraud represents (Abdallah et al., 2016). There are multiple credit card frauds: it can be the use of a stolen credit card, the use of a counterfeit credit card or simply the use of a card for which the identifier have been stolen (Bolton et al., 2001). Nowadays, fraud is also reaching e-commerce new ways of payment as mobile phones. In this thesis, we focus on check fraud detection which consists in using false checks to pay expensive products. The fraudsters modify the information of real checks to produce false ones by changing the series of numbers that identify the check.

	Predicted positive ($h(\mathbf{x}) = 1$)	Predicted negative ($h(\mathbf{x}) = 0$)
Actual positive ($y = 1$)	True Positive (TP)	False Negative (FN)
Actual negative ($y = 0$)	False Positive (TP)	True Negative (TN)

Table 2.1: Confusion matrix for a binary classification task. The label is denoted by y and the prediction by $h(\mathbf{x})$.

There are several other domains in which the fraud detection is useful such as health care insurance (Schiller, 2006) or automobile insurance (Artís et al., 2002). The problem also occurs in telecommunications (Shawe-Taylor et al., 1999) where a fraudster can subscribe to a service for which he has no intention to pay or by taking a legitimate account that will be used to make some calls which cannot be identified by the owner of the account and which leads to costs (Yusoff et al., 2013). Another type of fraud is the fake news. We refer the interested reader to Abdallah et al. (2016); Chandola et al. (2009); Aggarwal (2017); Kou et al. (2004) for a more complete survey on the the fraud detection applications.

2.2 Performance Measures

In this section, we present different metrics or performance measures that are suited in a binary imbalanced learning context. However, they can easily be extended to the multi-class setting as it is shown in Appendix A.

In imbalanced settings, we are not only interested in knowing whether the algorithm is making a mistake or not. We also need to focus on the particular classes on which the mistakes are made (see Table 2.1).

An example from the minority class predicted positive by the learned model is said to be a *True Positive (TP)*. If it is miss-classified it is called a *False Negative (FN)*. Similarly, for the majority class, a well classified example is called a *True Negative (TN)* and a *False Positive (FP)* otherwise.

The most common performance metric used to evaluate an algorithm is the *Accuracy*, also known as the complementary of the error rate. Using the notations of Table 2.1, they are defined as:

$$\text{Accuracy} = \frac{TP + TN}{m} \quad \text{and} \quad \text{Error rate} = \frac{FP + FN}{m}.$$

Let us consider a dataset in which we only have 1% of positive data. A simple way to achieve a high accuracy (or a low error rate) is to predict all the instances as negative. In such

a case, the performance of our classifier h is equal to 99%. At a first glance, we have learned a hypothesis h which is able to perform well on our dataset. However, it has completely missed all the interesting information and the class of interest for the user.

To overcome this issue, we need to use another performance measure which takes into account the capacity of the predicted model h to capture the minority class. The *Sensivity*, also known as *Recall* is defined as:

$$\text{Recall} = \frac{TP}{TP + FN}.$$

By definition, the Recall returns the percentage of examples in the class of interest the model is able to capture. The higher this value (i.e. the closer to 1), the more examples in the minority class are retrieved. By measuring how a model is able to detect the examples that belong to the minority class, the Recall seems to be a good candidate for a performance measure to use in imbalanced scenarios. In the context of the thesis, having a high recall very often involves refusing the transaction of most of the customers even if it is a genuine one. It can lead to huge marketing impacts by losing customers for which the payment has been refused. We also need to take into account how precise the model is in making decisions. The *Precision*, also called *Positive Predictive Value*, is defined as:

$$\text{Precision} = \frac{TP}{TP + FP}.$$

The Precision measures how a hypothesis h performs on the example predicted positive.

Other performance measures (Branco et al., 2016; Umberger et al., 2017; Konukoglu and Ganz, 2014) based on the classification made by h are given below:

$$\text{True Negative Rate} = \frac{TN}{TN + FP}, \quad \text{False Negative Rate} = \frac{FN}{FN + TP},$$

$$\text{False Positive Rate} = \frac{FP}{FP + TN}, \quad \text{Negative Predictive Value} = \frac{TN}{TN + FN}.$$

Such measures can be seen as complementary to the Precision and Recall. As shown before, the metric used depends on the user preference (Torgo and Ribeiro, 2007; Torgo and Lopes, 2011). For instance the *True Negative Rate* (also known as *specificity)* is the complementary of the *False Positive Rate*, since that the sum of these two quantities is equal to the number of negative examples in the dataset.

In check fraud detection, we have seen that it is important to take into account both the precision and the recall of the model of the model This is the goal of the F-measure introduced in the seventies (Rijsbergen, 1979).

2.2.1 F-measure

The F-measure is a good performance measure when a user wants to focus on the behavior of a model on a minority class. It is highly used in information retrieval (Sanderson, 1994)

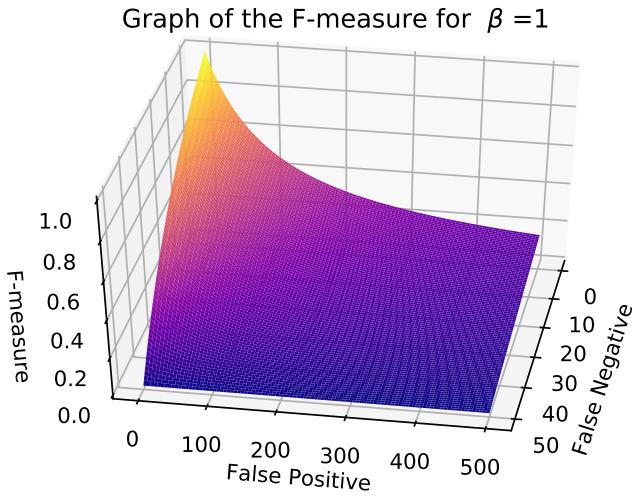

Figure 2.1: Illustration of the F-measure with 50 positive instances and 500 negative instances. The axes represent respectively: FN, FP and the value of the F-measure.

and obviously in Fraud and Anomaly Detection (Gee, 2014; Bahnsen et al., 2014; Bolton and Hand, 2002). It is defined as the harmonic mean of the Precision and the Recall and depends on a parameter β:

$$F_\beta = \frac{(1+\beta^2)\text{Precision} \times \text{Recall}}{\beta^2 \times \text{Precision} + \text{Recall}} = \frac{(1+\beta^2)TP}{(1+\beta^2)TP + \beta^2 FN + FP} = \frac{(1+\beta^2)(P-FN)}{(1+\beta^2)TP + \beta^2 FN + FP}.$$

An illustration of the F-measure is given in Figure 2.1. The F-measure is a flexible measure. By modifying the β value, the user is able to control the importance of either the Precision or the Recall. If $\beta = 1$ the same weight is given to both Precision and Recall. If the user wants to give more importance to the recall, he can choose a value of β greater than one. By choosing $\beta < 1$, he will give more importance to the precision.

Let us take an example of a sample of 1000 instances in which the minority class represents 1% of the data. So we have 10 positive examples for 990 negative ones. Let us also consider two hypothesis, h_1 and h_2, which have the following confusion matrices:

h_1

	Predicted positive	Predicted negative
Actual positive	TP = 3	FN7
Actual negative	FP = 0	TN = 990

h_2

	Predicted positive	Predicted negative
Actual positive	TP = 9	FN = 1
Actual negative	FP = 3	TN = 987

Both of these classifiers have an error rate less than 1%. The first classifier, h_1 has a Precision equal to 1 and a Recall equal to 0.3 while for the second classifier h_2, these two

values are respectively equal to 0.75 and 0.9. These two classifiers lead to completely different F-measures (with $\beta = 1$): 0.46 for the first hypothesis and 0.82 for the second one. Despite the high precision reached by the first classifier (equal to 1), the F-measure remains lower than the one achieved by the hypothesis h_2.

Compared to the Accuracy and despite how useful such a measure can be in imbalanced scenarios, it remains hard to optimize. The F-measure presents two main drawbacks: (i) it is non-linear and (ii) non convex, as depicted on Figure 2.1. Because of (i), it is hard to derive generalization guarantees for such a measure. Furthermore, we can not use standard gradient descent algorithms to optimize it. Because of (ii), an optimization algorithm can fall into local optima that might be far from the optimal solution. However, the literature is rich of studies and algorithms which aim to deal with such a task (Zhao et al., 2013; Busa-Fekete et al., 2015). A more complete state of the art is provided in Chapter 5 as the objective is to provide a new algorithm to optimize the F-measure.

2.2.2 Other Performance Measures

The *Class Weighted Accuracy*, noted CWA and proposed by Cohen et al. (2006) is similar to the F-measure. However, it does not take into account the Precision anymore but the Specificity instead, which is directly linked to the Precision of the model. It is expressed as a convex combination of both Sensitivity (i.e. the Recall) and Specificity , i.e. for any $\alpha \in [0, 1]$ it can be expressed as:

$$CWA = \alpha \times \text{Sensitivity} + (1 - \alpha) \times \text{Specificity}.$$

As for the β parameter of the F-measure, the user can also choose the value of the parameter α.

Another evaluation metric used in imbalanced scenarios is the G-mean measure. As for the Class Weighted Accuracy, its expression depends on both Sensitivity and Specificity. It is the square root of the product of these two quantities:

$$\text{G-mean} = \sqrt{\text{Specificity} \times \text{Sensitivity}}.$$

This measure is used in imbalanced scenarios because it takes into account the information on both classes but gives the same importance to each class.

A last popular evaluation metric used in imbalanced scenarios is the AU ROC (Metz, 1978; Hanley and McNeil, 1982; Cortes and Mohri, 2004), the“Area Under the Receiver Operating Curve”. This metric is used when the learned model returns a score, such as a confidence in a given prediction. The training examples can be ranked according to this score and a curve which plots the Recall according to the False Positive Rate is drawn using the different scores obtained by the model (see Figure 2.2). The closer to 1 the value of the AUC is, the better the model is.

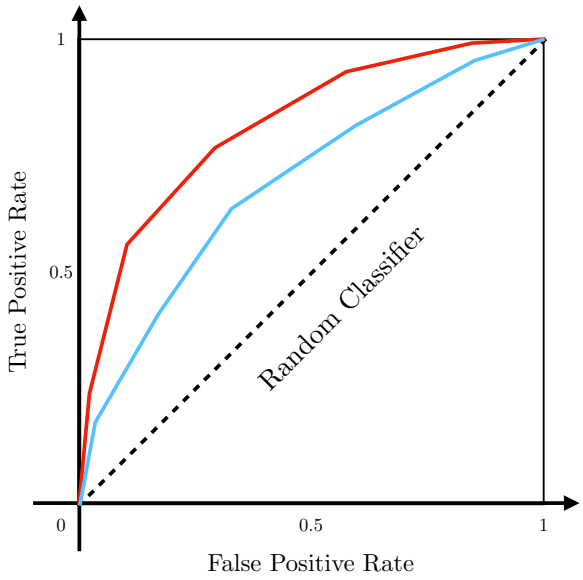

Figure 2.2: ROC and AU ROC for two different classifiers. The classifier associated to the red curve achieves a larger AUC than the blue one.

Compared to the previous measures, the AU ROC allows us to can visualize the performance of the model. Furthermore, it is usually used to choose the appropriate threshold for the given task. It also gives the possibility to plot several models on a single graph and choose the one best suited for a given purpose. The ideal model is the one that achieves a True Positive Rate equal to 1 while the False Positive Rate remains close to 0, i.e. when all the examples of the minority class have a greater score than the examples of the majority class.

There exist other measures used in imbalanced scenarios, such as the Average Precision, the H-measure or the Precision-Recall curve to cite a few (Frery et al., 2017; Ferri et al., 2009; Jeni et al., 2013; Branco et al., 2016; Behl et al., 2014).

A complementary solution to overcome the issues related to imbalance learning consists in balancing the dataset during a pre-process. We review in the next section the main strategies.

2.3 Pre-Processing

To avoid imbalanced situation, we can modify the distribution of the two classes using sampling methods (Garcia and He, 2008). This is maybe one of the most used solutions in such a context (Chawla et al., 2004). The main question is thus: *How do we have to modify the*

dataset to improve the performance of the future used algorithm?

The sampling of the data is performed either by replicating and removing some examples or creating new ones in an appropriate manner. The simplest procedure consists in doing this in a random way (Chawla et al., 2004), but several algorithms were also developed to focus on specific regions of the feature space (Dittman et al., 2014). Another method to "artificially" balance the two classes is to assign different costs to the two classes. In this case, we talk about *Cost Sensitive Learning* and it is related to what we will introduce in Section 2.3.3.

2.3.1 Undersampling Strategies

Assuming that information is redundant (Estabrooks et al., 2004), a first way to reduce the disequilibrium is to remove examples from the majority class. To get rid of this redundancy, one can randomly discard examples from the majority class. This method is called *Random Undersampling* (More, 2016). It presents one interesting advantage: by reducing the number of negative examples, the time required to learn a model also decreases. Other strategies called directed under-sampling were developed to control the data removed from the dataset. A first directed method has been introduced by Hart in 1968 for the Nearest Neighbor algorithm (Hart, 1968). It is called the *Condensed Nearest Neighbor* (CNN). Initially, this method was designed to reduce the storage of training examples for the k-NN algorithm. However, it can also be seen as an under-sampling strategy for the same reasons. The idea of this strategy is to take a subset of the training data which contains all the training examples from the minority class and a subpart of the instances from the majority class. Using this subset, we classify instances from the whole training set using a 1-NN rule and remove examples that are far from the decision boundaries i.e. not useful for the classification. However, this greedy strategy is not stable and depends on the initial chosen sample. Furthermore, we have no guarantee to converge towards the smallest meaningful subset.

Few years later, Ivan Tomek has proposed two modifications of the Condensed Nearest Neighbor algorithm, one of which is called *Tomek link* (Tomek, 1976). To choose how to remove examples from the training set, we consider two instances which do not belong to the same class. They are said to form a Tomek Link if it does not exist any other examples that are closer to the ones considered. In this case, the two examples are considered as borderline instances and, in order to improve the decision frontier between the two classes, the two examples can be removed from the training set. This method can also be used as an under-sampling method by only removing instances from the majority class.

Other cleaning methods such as *Edited Nearest Neighbor* (ENN) introduced by Wilson (1972) follow the same principle and aim to remove examples from the majority class for which the label differs from its k-nearest neighbor. Another similar one, based on ENN is called *Neighberhood Cleaning Rule* (NCL) (Laurikkala, 2001).

Finally, let us cite two methods which combine the previous mentioned ones: *One Sided Selection* (Kubat et al., 1997) which results from the combination of the successive application of Tomek Link and the Condensed Nearest Neighbor algorithm. The other one consists in

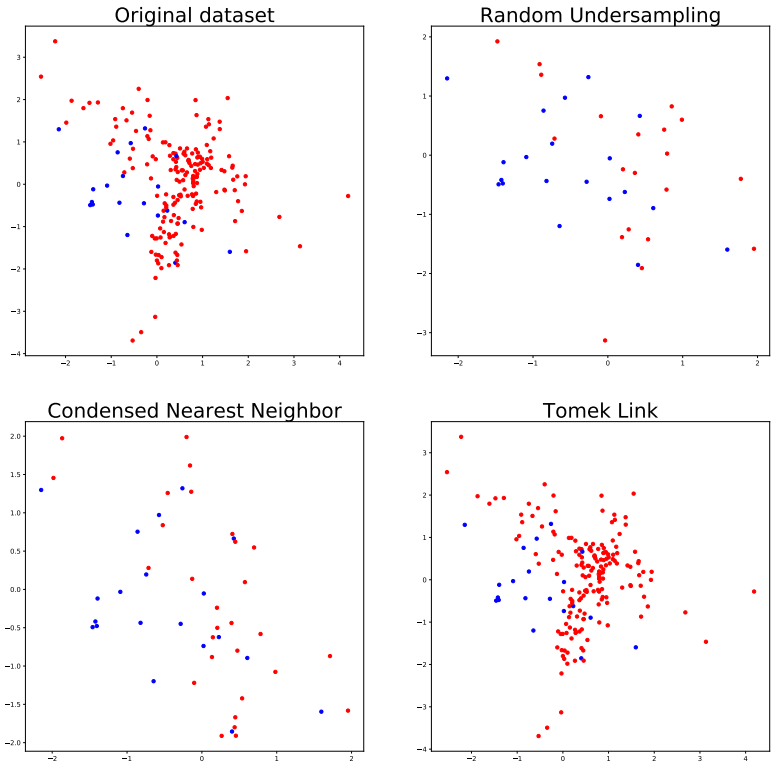

Figure 2.3: Illustration of some undersampling strategies: the top left plot is the original dataset with 200 instances among which 10% belong to the minority class. The other figures show respectively the results of the random under-sampling strategy, the Condensed Nearest Neighbor and Tomek Link strategies. Positive instances are depicted in blue and the negative ones in red.

applying CNN first followed by Tomek Link (del Jesus et al., 2006) The latter is preferred to the former because it is less computationally expensive when one is dealing with a large training set.

In Figure 2.3, we illustrate on a toy dataset the behavior of some of the under-sampling strategies. We note that the CNN procedure has removed a large number of examples from the majority class. On the other hand, due to the sparsity of the minority class in the dataset, we were not able to remove a lot of examples from the majority class using the Tomek Link Procedure.

2.3.2 Oversampling Strategies

Another way to balance the two classes by avoiding loosing to much majority class information is to increase the number of examples in the minority class. As for under-sampling methods, we can do it by randomly selecting the examples we want to duplicate. Directed oversampling strategies have been also developed to duplicate or create instances in the appropriate parts of the feature space.

The most popular one in the community is the SMOTE algorithm, i.e. the *Synthetic Minority Oversampling Technique* (Chawla et al., 2002). This algorithm, presented in Algorithm 2, uses the k nearest-neighbors belonging to the minority class to create a new one. One of the nearest neighbor of the considered example is randomly selected and a new instance is then created on the line (or the hyper-cube) between the considered example and the selected nearest neighbor.

Algorithm 2: SMOTE Algorithm.

Input: A training sample S with P positive and N negative examples.
Input: k: number of nearest neighbors to consider.
Input: R: rate of minority (positive) example to SMOTE.

Set $New = R \times P$: the number of synthetic instances Syn to be created.
Select randomly New instances among P and note *setind* their indexes
for all i in *setind* **do**
  search the k nearest-neighbor of x_i
  consider P_i, the i-th positive, and select randomly one of its nearest neighbors denoted by NN_i.
  for all attributes *attr* of P_i **do**
    choose a number $\alpha \in [0, 1]$
    set $newattr = \alpha P_i[attr] + (1 - \alpha) NN_i[attr]$
    set $Syn_i[attr] = newattr$
  end for
end for
Output: A training sample S' with $S' = S \cup Syn$

This method has been shown to be more efficient than the standard random oversampling one (Chawla et al., 2002). In this version, the procedure can be applied to any example in the minority class, even if adding information is not necessary while this can be interesting for borderline instances.

A variant of SMOTE algorithm called *Border SMOTE* proposed by Han et al. (2005) tries to focus on borderline instances in the minority class by first detecting the"unsafe" examples, i.e. for which a large enough number of its $k-$nearest neighbors belong to the majority class. For these instances, a SMOTE algorithm is applied to force the algorithm to capture the minority example in this region of the space. Meanwhile, if the number of nearest neighbors in the majority class of the studied instances is small, nothing is done. Following the same idea of generating data in appropriate regions of the space, *Safe Level SMOTE* (Bunkhumpornpat et al., 2009) was developed. It can be considered as a smoother version of Border SMOTE because the number of synthetic data generated depends on the safe level of the minority examples where the safe level is the rate of minority examples among the $k-$nearest neighbors. The safer an example is the more synthetic data are generated. For

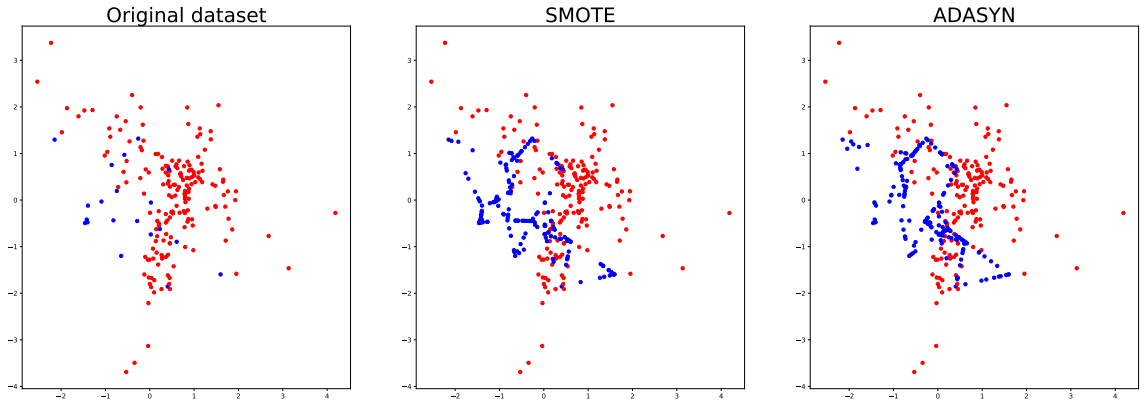

Figure 2.4: Illustration of two oversampling strategies: the first figure shows the original 2D dataset with 200 instances among which 10% belong to the minority class. The two other figures represent the dataset respectively after applying the SMOTE and ADASYN procedures.

an exhaustive list of the SMOTE based algorithms, the interested reader is referred to a recent review done by Chawla et al. (Fernández et al., 2018). Among them, *ADASYN* (He et al., 2008) is very similar to SMOTE, as shown in Figure 2.4. The difference lies in the use of a distribution to automatically decide the number of synthetic data that has to be generated for a given minority example.

In Chapter 4, we will compare these sampling strategies and the k-Nearest Neighbor algorithms to our Corrected Nearest Neighbor Algorithm.

Oversampling strategies can also be combined to under-sampling ones. Usually, SMOTE is combined with either ENN or Tomek Link (Batista et al., 2003).

A last way to re-balance the two classes is to assign costs on each class. We talk about *cost-sensitive Learning*.

2.3.3 Cost-Sensitive Learning

In most Machine Learning algorithms, when an example is miss-classified, the error suffered by the classifier is counted 1, and 0 otherwise. However, let us take the bank fraud detection problem. In this specific filed of application, it might be interesting to penalize the hypothesis according to the nature of the errors.

This penalization can be applied either at *the transaction* level, where a weight is assigned with respect to the amount of money involved (Bahnsen et al., 2013; Correa Bahnsen et al., 2017), or at *class level*, by taking into accounts, e.g. the potential dissatisfaction of the customers whose transaction has been rejected. (i.e. false positives). This second boils down to filling in the cost matrix presented in Table 2.2.

This idea is called cost-sensitive learning that received much attention since the seminal

	Predicted positive ($h(\mathbf{x}) = 1$)	Predicted negative ($h(\mathbf{x}) = 0$)
Actual positive	c_{TP}	c_{FN}
Actual negative	c_{FP}	c_{TN}

Table 2.2: Cost matrix for a binary classification task.

paper of Elkan (2001). Compared to sampling strategies, cost-sensitive methods do not remove information (like undersampling) nor add new (potentially noisy) information (like oversampling).

Furthermore, the meaning behind the given weights can be directed by the target application and provided by the expert of the domain (Bahnsen et al., 2013).

On the other hand, Liu and Zhou (2006) have studied the effect of the cost values when they are assigned to the classes. They have shown that these weigths have to be asymmetric to perform well. A study has been conducted to determine if the sampling methods perform better than cost-sensitive ones (López et al., 2012). The authors conclude that, in average, one method is not better than the other and that the problem depends on the structure of the data.

We will show how cost-sensitive methods can be used to improve the retailers benefits in Chapter 6 of this thesis, by assigning costs that are example dependent. We will also show that these methods can be used for theoretical studies. As we will see in Chapter 5, maximizing some complex performance measures can lead to cost-sensitive learning. This is the case for all linear fractional performance measures such as the F-measure (Boyd and Vandenberghe, 2004; Narasimhan et al., 2015b; Parambath et al., 2014).

2.4 Algorithms for Learning in Imbalanced Settings

In the previous section, we focused on how to prepare the data in imbalanced settings. We will now focus on the different types of algorithms we can use to tackle the imbalance problem.

Due to the presented contributions in this manuscript, we will focus on SVM, Boosting and Tree-based methods. Section 2.4.4 gives a brief over-review on other types of algorithms.

2.4.1 SVM-based Methods

We have already seen the linear SVMs are not directly suited for imbalanced learning since they are biased toward the majority class when the imbalance ratio is increasing in the dataset. However, the SVM algorithm has hyperparameters that can be modified, or tuned, to achieve our goal, such as the margin as proposed by Yang et al. (2009). It aims to reduce the bias and refine the decision boundary for nonlinear SVMs. In fact, due to the simplicity of the linear SVM and the complexity of the data in imbalanced scenarios, it has been shown that they

poorly perform and, by introducing non linearity, i.e. using a kernel (Akbani et al., 2004), we get better result in this context.

However, we still face the same problem: all the loss function used to train the model are based on the accuracy. In (Morik et al., 1999; Wu and Srihari, 2003; Bach et al., 2006; Masnadi-Shirazi and Vasconcelos, 2010), the authors proposed to combine the power of cost-sensitive learning and the nonlinear SVMs to improve their performances. It is done by optimizing a different loss function which gives asymmetric weights to each class:

$$\begin{aligned} & \min_{\mathbf{w},b,\boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|^2 + c_+ \sum_{i|y_i=1} \xi_i - c_- \sum_{i|y_i=-1} \xi_i, \\ \text{s.t.} \quad & y_i(\mathbf{w}^T \Phi(\mathbf{x}_i) + b) \geq 1 - \mathbf{x}_i, \quad \forall i = 1, 2, ..., m, \\ & \xi_i \geq 0 \quad \forall i = 1, 2, ..., m, \end{aligned}$$

where c_+ and c_- are the weights given to the positive and negative classes respectively. By this way, the authors can constrain the learned model to focus more on minority examples.

In Tang et al. (2009) and Wang et al. (2012), the authors propose a more sophisticated strategy. They both combine several pre-processing methods together, by assigning different weights to both classes and using sampling methods such as SMOTE or Tomek Link to balance the two classes. In Thai-Nghe et al. (2010), the authors go a little further by proposing a procedure to choose the assigned costs.

Another type of SVM-based method used in imbalanced scenarios is the Fuzzy SVM (Lin and Wang, 2002). In standard SVM, all the points from the training **equally** contribute to the learned decision boundary. The fuzzy property assumes that all the examples do not have the same importance. This importance is introduced by adding a parameter s_i on the slack variables of the $i-$th example which translates the role played by the instance. The Fuzzy SVMs can be made less sensitive to outliers, but they still poorly perform on imbalanced data (Batuwita and Palade, 2013).

To overcome this issue Batuwita and Palade (2010), proposed to combine the fuzzy SVM with a cost sensitive approach. The value of the fuzzy parameter s_i becomes a parameter of the imbalance ratio in order to delete the effect of the imbalance. Furthermore, this value also depends on the distance of the example from the center of its class, i.e. the closer the example is from the center of the class, the higher the fuzzy parameter is. This idea is based on the assumption that examples that are close to the center from an important cluster of same class instances.

Note there also exists an unsupervised version of the SVM algorithm called *One class Support Vector Machine (OCSVM)* (Scholkopf and Smola, 2001) which has been studied for anomaly detection. OCSVM does not use the label of the data anymore. The formulation of the optimization changes in the following form:

$$\begin{aligned} &\min_{\mathbf{w},\rho} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{m\nu}\sum_{i=1}^{m}\xi_i - \rho, \\ \text{s.t.} \quad &\mathbf{w}^T\Phi(\mathbf{x}_i) \geq \rho - \xi_i, \quad \forall i = 1, 2, ..., m, \\ &\xi_i \geq 0 \quad \forall i = 1, 2, ..., m. \end{aligned} \text{ ,}$$

where the parameter ρ is the margin and ν replaces the parameter C. In this formulation, the parameter ν is an upper bound on the fraction of outliers and a lower bound on the fraction of support vectors (Scholkopf and Smola, 2001). Basically, the OCSVM tries, in the feature space induced by the transformation Φ, to find a hyper-plane with the largest margin for which, at most a fraction ν of the training data lies under the hyper-plane. OCSVM has been used for document classification (Manevitz and Yousef, 2001) and in computer security (Heller et al., 2003). Variants of OCSVM have been proposed to improve their performance in anomaly detection, among which $\eta-$*one class SVM* (Amer et al., 2013) in which the idea is to give less importance to the outliers that lie on the learned decision boundary. This is done by introducing a variable which estimates the "normality" of a point.

2.4.2 Boosting

Boosting has also been shown to be a good candidate to address imbalanced problems. Boosting is an ensemble method which combines the diversity of *weak classifiers* into a single and strong classifier able to have better performance.

In practice, a weak classifier is a hypothesis h which achieves a slightly better performance than random guessing. A well-known boosting algorithm of boosting is *Adaboost* (Freund et al., 1999; Rätsch et al., 2001) which iteratively focuses on hard examples. More formally, at the first stage, all the examples have the same weights $w_i^{(0)} = 1/m$. Let us now assume that $j-1$ iterations were done so the examples have a weight equal to $w_i^{(j-1)}$ and a classifier $h^{(j)}$ is learned. The classification error $\varepsilon^{(j)}$ is then evaluate and used to update the weights of the training sample using an exponential function as follows:

$$\alpha^{(j)} = \log\left(\frac{1-\varepsilon^{(j)}}{\varepsilon^{(j)}}\right) \quad \text{and} \quad w_i^{(j)} = w_i^{(j-1)}\frac{\exp(-\alpha^{(j)} y_i h^{(j)}(\mathbf{x}_i))}{Z^{(j)}},$$

where $Z^{(j)}$ is a constant of normalization and $\alpha^{(j)}$ is a weight assigned to the hypothesis $h^{(j)}$ which directly depends on its performance. The weighted function gives more weight to the miss-classified examples and a new hypothesis is learned from this new distribution. The final model is defined by $F(\cdot) = \sum_{k=0}^{K}\alpha^{(k)}h^{(k)}(\cdot)$, and the sign of $F(\mathbf{x}_i)$ is used to predict the label of the example. This algorithm has been used to improve linear models and is also used for imbalanced learning (Fan et al., 1999; Wang and Japkowicz, 2010). In order to tackle the imbalance issue, the authors combine this method with a cost-sensitive learning scheme, such that the way the examples are re-weighted also depends on the class weights.

2.4.3 Tree-Based Methods

Compared to SVM, Tree-based algorithms are able to directly introduce non linearity in the learned model by partitioning the space according to the features. Decision trees are often used in the context of imbalanced data (Cieslak and Chawla, 2008; Parvin et al., 2013; Correa Bahnsen et al., 2017) because they are (i) easier to learn and so suitable to deal with large datasets, and (ii) interpretable (each path of the tree can be seen as a knowledge rule).

In decision tree learning, one has to set a number of hyper-parameters: *the depth, the number of examples in the leaves, a threshold, etc ...* . These hyper-parameters are crucial to avoid the main drawback of decision trees: the *over-fitting* phenomenon. This situation is usually met when the minority examples are rare or isolated: some of the leaves will contain at most one or two minority examples and the class boundaries will be too small so that new minority examples will rarely fall into these leaves. To improve decision tree algorithms, the use of *hybrid approaches* (Galar et al., 2012) is widespread in the community. It consists in mixing a sampling strategy or a cost-sensitive method with a learning algorithm (Lomax and Vadera, 2013). By using a sampling strategy, the algorithm is able to focus on minority examples even if they lie among a dense set of majority examples.

If a model over-fits, it will present a large variance during the test phase on multiple datasets. A first way to reduce the variance of the model is to combine several decision trees together where each of them is learned with a different sub-sample of the training set. This ensemble approach is called the *bagging* and the use of bagging with decision trees leads to the *Random Forest* (RF) algorithm (Breiman, 2001).

In the RF algorithm, each tree t_k is built using a sub-sample S_k of the training set, obtained by sampling strategies without replacement or by bootstrap. Each tree is built independently which makes this algorithm even more useful in practice, due to its ability to be performed in parallel (Khoshgoftaar et al., 2007). In its original version, a majority vote $F(\mathbf{x}_i)$ is then made between the trees to predict the label of a new income $\mathbf{x}_i$ where each tree has the same weight in the final prediction:

$$F(\mathbf{x}_i) = \sum_{k=1}^{T} t_k(\mathbf{x}_i).$$

The use of a random forest leads to more stable results, by avoiding the over-fitting phenomenon. A variant of random forests consists in giving different weights to the trees according to their performance or by learning the weights of each predictors in order to improve the final decision (Zenko et al., 2001). The weights $\boldsymbol{\alpha}^\star$ are learned in order to optimize a loss function ℓ such that:

$$\boldsymbol{\alpha}^\star = \underset{\boldsymbol{\alpha}\in\mathbb{R}^T}{\operatorname{argmin}}\ \mathbb{E}[\ell(F(\mathbf{x}), \mathbf{y})] \quad \text{where} \quad F(\mathbf{x}) = \sum_{k=1}^{T} \alpha_k t_k(\mathbf{x}).$$

A very efficient method consists in combining decision trees with boosting. This led to

the famous *Gradient Boosting* (Friedman, 2000).

Gradient boosting has been shown to be very efficient to deal with classification problems, and a very good candidate to address issues due to imbalanced data (Beygelzimer et al., 2015; Li et al., 2007). Unlike the well-known Adaboost algorithm (Freund and Schapire, 1999), gradient boosting performs an optimization in the *function* space rather than in the *parameter* space. At each iteration, a weak learner f_k is learned using the *residuals* (or the errors) obtained by the linear combination of the previous models. The linear combination F_k at time k is defined as follows:

$$F_k = F_{k-1} + \alpha_k f_k, \tag{2.1}$$

where F_{k-1} is the linear combination of the first $k-1$ models and α_k is the weight given to the k^{th} weak learner. The weak learners are trained on the residuals $r_i = y_i - F_{t-1}(\mathbf{x}_i)$ of the current model. These residuals are given by the negative gradient, $-g_t$, of the used loss function L with respect to the current prediction $F_{t-1}(\mathbf{x}_i)$:

$$r_i = g_k(\mathbf{x}_i) = -\left[\frac{\partial L(y_i, F_{k-1}(\mathbf{x}_i))}{\partial F_{k-1}(\mathbf{x}_i)}\right].$$

Once the residuals r_i are computed, the following optimization problem is solved:

$$(f_k, \alpha_k) = \underset{\alpha, f}{\operatorname{argmin}} \sum_{i=1}^{m} (r_i - \alpha f(\mathbf{x}_i))^2.$$

Finally, the update rule 2.1 is applied.

This algorithm has been first developed for classification and regression trees, and most of the work and libraries such as **XGBoost** (Chen and He) are using decision trees as weak learners. To be considered as weak learners, the learners mainly consist in decision stumps or tree with a small depth. Gradient boosting, on the contrary of Adaboost allows to use custom losses as we will see in Chapter 6.

2.4.4 Other Algorithms

The first sections have focused on SVM, Boosting and Tree-based algorithms. In this section, we present other algorithms which can be used to detect anomalies or frauds using a notion of distance between examples. We also present some techniques based on *Metric Learning* used in imbalanced scenarios. We end this section by presenting how neural networks can be also used.

Distance-Based Methods

A first category of distance-based outlier detection techniques and uses the $k-$NN algorithm. A definition of outlier based on the distance was given by Knox and Ng (1998):

Definition 2.1. *An instance $\mathbf{x}_i$ in a sample S is a $DB(\eta, D)$-outlier if at least a fraction η of the examples lies at a distance greater than D.*

In other words, an example $\mathbf{x}_i$ is an outlier if it lies to far from a cluster formed by most of the points, i.e. if the point is isolated. Following the same idea, Breunig et al. (2000) proposed a way to assign a score to the examples according to their distance to their k nearest neighbors and the reachability of these points among their neighbors. The proposed measure will return a score close to 1 if an example is close to an homogeneous cluster of points and 0 otherwise, i.e. if the point is isolated in the dataset or in a given cluster (Aggarwal, 2017; Zhang et al., 2009).

The main drawbacks of such methods is that they depend on the considered metric used to compute the distances between the examples. The use of an other distance may lead to completely different results.

The standard metric used to compute the similarity between two points is the euclidean distance defined by:

$$\forall \mathbf{x}, \mathbf{x}' \in \mathbb{R}^d, \quad d_2(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|_2 = \sqrt{\sum_{k=1}^{d} (x_k - x_{k'})^2}.$$

However, this distance gives the same weight to all the features and thus, fails to focus on the most discriminative features. Therefore, to learn a new representation which aims to separate better the two classes by learning the discriminative features. This domain of machine learning is called *Metric Learning* (Weinberger and Saul, 2009; Bellet et al., 2013), an area which typically aims to learn a suited projection $\boldsymbol{M}$ for the given task. In the context of Mahalanobis Metric Learning, the new similarity measure between two examples is defined by:

$$d_{\boldsymbol{M}}(\mathbf{x}, \mathbf{x}') = \sqrt{(\mathbf{x} - \mathbf{x}')^T \boldsymbol{M} (\mathbf{x} - \mathbf{x}')}.$$

To use $d_{\boldsymbol{M}}$ as a true distance for the $k-$Nearest Neighbor algorithm it has to verify the following axioms:

- $d_{\boldsymbol{M}}(\mathbf{x}, \mathbf{x}') \geq 0$ for all $\mathbf{x}, \mathbf{x}'$,
- $d_{\boldsymbol{M}}(\mathbf{x}, \mathbf{x}') = 0$ if and only if $\mathbf{x} = \mathbf{x}'$,
- $d_{\boldsymbol{M}}(\mathbf{x}, \mathbf{x}') = d_{\boldsymbol{M}}(\mathbf{x}', \mathbf{x})$ for all $\mathbf{x}, \mathbf{x}'$,
- $d_{\boldsymbol{M}}(\mathbf{x}, \mathbf{x}') \leq d_{\boldsymbol{M}}(\mathbf{x}, \mathbf{x}'') + d_{\boldsymbol{M}}(\mathbf{x}'', \mathbf{x}')$ for all $\mathbf{x}, \mathbf{x}'$ and $\mathbf{x}''$.

In order for $d_{\boldsymbol{M}}$ to be a distance, it suffices for $\boldsymbol{M}$ to be *Symetric Positive Semi Definite* (PSD). In practice, the learned metric $\boldsymbol{M} \in \mathbb{R}^{d \times d}$ is often of rank l lower than the initial dimension of data d if there are correlations. Therefore, finding a PSD matrix $\boldsymbol{M}$ leads to a new representation in a lower dimensional space obtained with a matrix $\boldsymbol{L}$ deduced from the Cholesky decomposition of $\boldsymbol{M} = \boldsymbol{L}^T \boldsymbol{L}$.

Because the dimension of the new representation of the data is unknown, we rather learn $\boldsymbol{M}$, by adding the constraint that it should be PSD, instead of $\boldsymbol{L}$. Furthermore, learning $\boldsymbol{L}$ directly leads to non-convex optimization problems (Bellet et al., 2015).

The main drawback of metric learning is the computation time. In fact, the metric $\boldsymbol{M}$ is typically learned using the distance between *similar* and *dissimilar* pairs. The aim of this metric is to bring together examples that are similar, i.e. belong to the same class and to push away pairs of examples which do not belong to the same class. At each step of a metric learning algorithm, such as in LMNN (Weinberger and Saul, 2009), we need to project the matrix $\boldsymbol{M}$ on the set of PSD matrices.

However, due to its capacity to learn a new representation of the data, metric learning based techniques have been shown to be relevant for imbalanced learning (Wang et al., 2018; Feng et al., 2018).

In Chapter 3 we will present an algorithm for imbalanced learning based on a metric learning approach for which the learned metric $\boldsymbol{M}$ is automatically PSD if we solve the dual formulation.

Neural Networks based Methods

Deep Learning algorithms were also developed for imbalanced learning and anomaly detection (Lebichot et al., 2019) from both supervised and unsupervised perspectives.

A way to use deep learning approaches for anomaly detection can be done using auto-encoders (Sakurada and Yairi, 2014; Zhou and Paffenroth, 2017). The aim of the auto-encoder is to learn a representation of the data in a lower or greater dimensional space without loss of information (i.e. a small error of construction). The auto-encoder is composed of two parts, the *encoder* part: in which the representation is learned and a *decoder* part which reconstructs the original example from the latent representation. The optimization problem can be written as:

$$\psi_e, \psi_d = \underset{\psi_1, \psi_2}{\operatorname{argmin}} \frac{1}{m} \sum_{i=1}^{m} (\mathbf{x}_i - \psi_2 \circ \psi_1(\mathbf{x}_i))^2,$$

where ψ_e represents the encoding function and ψ_d the decoding function. For anomaly detection, the majority class is given to train the auto-encoder. New instances are then passed into the network and the reconstruction error is evaluated. When this error reaches high values or greater values than the ones obtained during training, the example is considered to be an anomaly (Williams et al., 2002).

We have seen that the use of sampling methods has been shown to be efficient for imbalanced scenarios. It gives the possibility to learn a better decision boundary to separate the two classes.

Recently, a new type of algorithms, mainly used in computer vision has been developed to improve a class of hypotheses by iteratively sampling new instances each time the hypothesis

is updated. This method is called *Generative Adversarial Networks (GAN)* (Goodfellow et al., 2014). The idea is to iteratively improve the decision boundary of the learned classifier by alternating a phase in which the classifier is updated with a phase in which data are generated. These data have the particularity to be generated such that the classifier has to distinguish between the generated instances and the real ones. This process is done by solving a *min-max* problem:

$$\min_G \max_D \ell(D, G) = \mathbb{E}_x[\log(D(\mathbf{x}))] + \mathbb{E}_z[\log(1 - D(G(\mathbf{z})))],$$

where D is the *discriminator*, i.e. the learned model, and G is the *generator*, where $\mathbf{z}$ represents the noise used to generate new examples.

In its basic formulation, GANs are trained on a single class of examples. A classifier is trained to distinguish between the real and generated data. Once the classifier is trained, it can be used to detect anomalies. An anomaly is then seen as an example created by the part of the network that generates data. They can also be used in a supervised way by adding the label as an information for the GANs (Douzas and Bacao, 2018). By doing so, they are used as sampling strategy to generate example from both classes.

2.5 Conclusion

In this chapter, we gave an overview of the main approaches for tackling the imbalanced problem. Firstly, we introduced several performance metrics that are well suited for the context of this thesis. Secondly, we described how the imbalance in the data can be reduced using sampling techniques and how to force the classifiers to focus on the minority class examples by giving them a higher weight. Lastly, we have introduced a very wide variety of algorithms that are used in the field.

The contributions of this thesis consist in developing new techniques to deal with the imbalanced classification issue and to apply them for check fraud detection. We propose to study different aspects to address this problem (i.e. learning a new representation of the data, assigning weights to each classes, etc.) and sometimes by combining several of the methods described in this chapter in order to improve their performance. We also derive theoretical guarantees in some of our contributions, guarantees that are often absent in the state of the art.

Part II focuses on so-called geometric approaches for imbalanced classification by learning a new representation of the data or modifying the distance of a new query to a positive example. In Chapter 3, we propose a novel approach based on the One Class SVMs algorithm. Our proposed method consists in the use of the proven frauds. They are used to build region of the space for which a new data is likely to be a fraud, whose shape is that of an ellipsoid by using metric learning techniques. In Chapter 4, we propose a modification of the k-nearest neighbors algorithm which takes the sparsity of minority class examples into account by

modifying the distance of new query to its nearest positive examples.

Part III is devoted to the use of cost-sensitive methods. In Chapter 5, we propose to tackle the issue of optimizing the F-measure. Based on our theoretical study and our proposed bounds on the optimal F-measure, we derive an algorithm which iteratively choose the best cost parameters to assign to each class to maximize the performance metric. In Chapter 6, we propose several tree-based models to improve the model used by Blitz. These new models include the amount of money involved in a transaction as a cost parameter in order to improve the profits of Blitz customers.

Part II

Geometric Approaches

Chapter 3

Learning Maximum Excluding Ellipsoids

This chapter is based on the following publications

Guillaume Metzler, Xavier Badiche, Brahim Belkasmi, Elisa Fromont, Amaury Habrard, and Marc Sebban. Learning maximum excluding ellipsoids from imbalanced data with theoretical guarantees. In *Pattern Recognition Letters*, volume 112, pages 310–316. Elsevier BV, 2018a

Guillaume Metzler, Xavier Badiche, Brahim Belkasmi, Elisa Fromont, Amaury Habrard, and Marc Sebban. Apprentissage de sphères maximales d'exclusion avec garanties théoriques. In *Conférence francophone sur l'Apprentissage Automatique (CAp-17)*, 2017

Abstract

In this chapter, we present a learning method from imbalanced data (like in fraud detection) based on an unsupervised variant of the SVM called the *Support Vector Data Description*. We revisit this unsupervised learning algorithm to propose ME^2: *Maximum Excluding Ellipsoids*. Assuming that minority examples are locally concentrated in some regions of the space, the proposed approach aims to build an influence area around each positive by rejecting the negative ones. The areas are optimized in the form of ellipsoids in order to take into account the local specific geometry of the data using a metric learning approach. Unlike standard metric learning methods, ME^2 does not require a complex algorithm to learn and guarantee the PSD property of the metric. We show that, by solving the dual formulation, we get this property on the metric for free.

We derive theoretical guarantees on our algorithm using the uniform stability framework presented in Chapter 1. These guarantees prevent ME^2 from false alarms, i.e. from predicting negative examples as positives. Experiments conducted on several imbalanced datasets show the effectiveness of the proposed method compared to standard machine learning algorithms. We also performed experiments on a real dataset from the Blitz company.

3.1 Introduction

In Chapter 2, we have seen that anomalies can be characterized by the fact they lie in the tails of the distribution. Geometrically, it means that such points are far from clusters of points, i.e. the distance between the anomaly and the center of a given cluster is greater than the distance between all the points in a cluster and the center. This simple and intuitive idea led to the unsupervised technique based on the *Minimum Enclosing Ball Problem* (Sylvester, 1857). The goal is to find the smallest hypersphere which contains all the data where both the center and the radius of the hypersphere must be learned. An illustration of the problem is given in Figure 3.1 (left). Given a collection of data $\{\mathbf{x}_i\}_{i=1}^m$, we can write the corresponding optimization problem as follows:

$$\begin{array}{ll} \min\limits_{R,\mathbf{c}} & R^2, \\ s.t. & \|\mathbf{x}_i - \mathbf{c}\| \leq R^2, \ \forall i = 1, \ldots, m. \end{array} \tag{3.1}$$

For an anomaly detection task, it is now interesting to allow some points to lie outside the sphere. Such points will be then considered as anomaly. The resulting optimization problem aims to find a trade-off between the having the smallest radius and including all the points in the learned sphere (Tax and Duin, 2004) More formally, the optimization problem 3.1 will be rewritten as:

$$\begin{array}{ll} \underset{R,\mathbf{c},\boldsymbol{\xi}}{min} & R^2 + \frac{\mu}{n}\sum_{i=1}^m \xi_i, \\ s.t. & \|\mathbf{x}_i - \mathbf{c}\|^2 \leq R^2 + \xi_i, \quad \forall i = 1, \ldots, m, \\ & \xi_i \geq 0, \end{array} \tag{3.2}$$

where R and $\mathbf{c}$ are respectively the radius and the center of the ball and ξ_i is the slack variable associated to the i^{th} instance $\mathbf{x}_i$. The parameter μ is tuned in order to control the proportion of the data outside the sphere. (considered as anomalies - see Figure 3.1, right). In practice, the value of the parameter μ is chosen so that it corresponds to the (supposedly known) proportion of outliers in the dataset. Note that in Pauwels and Ambekar (2011), the authors have shown that using the radius instead of the square of the radius in this formulation is often preferable.

The optimization problem 3.2 is really similar to the optimization problem of the *Support Vector Machine* where the term R^2 plays the same role as $\|\mathbf{w}\|_2$, where $\mathbf{w}$ described the learned hyperplane. Note the constraints are also similar from a problem to the other.

For these reasons, the Minimum Enclosing Ball Problem will be now referred as *Support Vector Data Description* as presented by Tax and Duin (1999).

In the next section, we give an overview of SVDD based methods and present how this unsupervised algorithm inspired us to propose *Maximum Excluding Ellipsoids* (ME^2) which can be used to detect frauds in an imbalanced supervised setting.

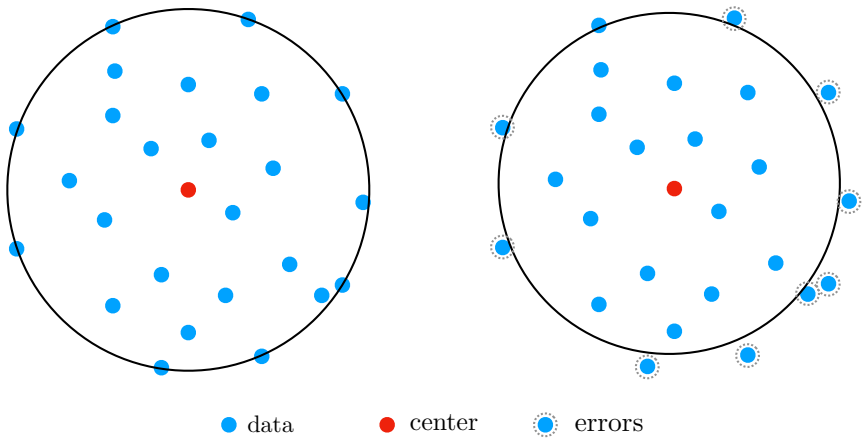

Figure 3.1: Illustration of the Minimum Enclosing Ball Problem (on the left) and of Support Vector Data Description (on the right). The points outside the sphere (the errors) are considered as by the algorithm.

3.2 Support Vector Data Description

It is worth noticing that there is a link between SVDD and the unsupervised Support Vector Machine algorithm called *One Class SVM* (Scholkopf and Smola, 2001).

It has been shown (Elzinga and Hearn, 1972) that the Problem 3.1 can be rewritten as:

$$\begin{aligned} \min_{\mathbf{c},\rho} \quad & \frac{1}{2}\|\mathbf{c}\|_2^2 - \rho, \\ s.t. \quad & \mathbf{c}^T\mathbf{x}_i \geq \rho + \frac{1}{2}\|\mathbf{x}_i\|_2^2, \ \forall i = 1, \ldots, m, \end{aligned} \tag{3.3}$$

where $\rho = \frac{1}{2}\left(\|\mathbf{c}\|_2^2 - R^2\right)$.

On the other hand, the optimization problem of the One Class SVM is as follows:

$$\begin{aligned} \min_{\mathbf{c},\rho'} \quad & \frac{1}{2}\|\mathbf{c}\|_2^2 - \rho', \\ s.t. \quad & \mathbf{c}^T\mathbf{x}_i \geq \rho', \ \forall i = 1, \ldots, m, \end{aligned}$$

where $\rho' = \rho + \frac{1}{2}\|\xi_i^2\|_2^2$. If we assume that $\|\mathbf{x}_i\|_2$ is a constant for all $i = 1, ..., m$, then we can see the One Class SVM optimization problem as a particular case of SVDD.

While One Class SVM and SVDD are unsupervised methods, a supervised variant has been proposed in Tax and Duin (2004). It aims to include the examples of the same class in a ball and to exclude the examples from the other class. Note that it can also be used for multi-class classification (Boujnouni et al., 2012) using multiple spheres, one sphere per class. A refinement of this approach has been proposed later by Liu and Zheng (2006), in which the authors learn two linear separators, one that include the examples from a given a class and the other which aims to exclude the examples from the other class. The first linear separator

aims to embrace the data from one class while the second linear separator aims to exclude the data from the other class. The objective is then to maximize the distance between these two separators. Such a framework is close to the SVM which aims to maximize the margin between the two classes. This type of algorithms works well when a class is located in a part of the space, so that the data are (linearly) separable. To ensure this point, most of the papers apply non linear transformations to the data (Le et al., 2013; Boujnouni et al., 2012; Kang et al., 2010; Duan et al., 2016) such as kernel methods. Even if the kernel-based methods are effective, the computation of the kernel is often expensive (according to the number of examples in the dataset) and does not scale well on most real datasets.

An interesting approach, which does not suffer from this drawback, is presented by Wang et al. (2010). The authors include a linear transformation of the data, in the form of a Positive Semi Definite (PSD) matrix $\mathbf{M}$ in the SVDD optimization problem. To avoid high computational costs, they set $\mathbf{M}$ to be the inverse of the covariance matrix, which allows the induction of ellipsoids rather than spheres. Such objects are able to cover a larger volume in the input space compared to the spheres (i.e. when $\mathbf{M} = \mathbf{I}$). Indeed, they are able to capture betters the geometry of the data.

Following the assumption that frauds are locally concentrated in different regions of the space, we follow the idea of Zhang et al. (2015), i.e. we learn several local models. We also assume that new frauds have more chance to appear near an existing one. For this reason, each model will be built around a known fraud. For each model, centered at a positive instance, we learn a linear transformation of the data. However, instead of computing the covariance matrix locally, as it is done by Wang et al. (2010), we have optimize this linear transformation under constraints. Combining local models with this metric learning-based approach leads to our algorithm ME^2, presented in the next section.

3.3 ME^2: a Metric Learning-based Algorithm for Optimizing Excluding Ellipsoids

In this section, we present the general formulation of our proposed method ME^2. It aims to create influence areas around each positive example and is based on the *Support Vector Data Description* and a local metric learning approach which aims to build new representations of the data. We show that the solution of the primal problem is a closed-form expression of the solution of the dual formulation, which leads to an easier optimization problem.

3.3.1 Problem Formulation

Let $S = \{\mathbf{x}_i\}_{i=1}^n$ be a sample of n *negative* instances (the majority class) and $P = \{\mathbf{c}_j\}_{j=1}^p$ a set of p *positive* examples (the minority class), with $n >> p$, $m = p + n$ and where each $\mathbf{x}_i, \mathbf{c}_j$ are feature vectors of $\mathbb{R}^d$. We aim at maximizing ellipsoids centered at each positive $\mathbf{c} \in P$ excluding (most of) the negative data $\mathbf{x}_i,\ i = 1, ..., n$. Learning such ellipsoids boils

down to optimizing a Mahalanobis distance, that is finding a positive semi-definite (PSD) $d \times d$ matrix $\mathbf{M} \in \mathbb{S}^+$ projecting the data linearly in a new space and allowing to obtain balls centered at each positive example of maximum radius R. Note that the size of the projection space is equal to the rank of matrix $\mathbf{M}$. This is important because a small rank matrix (when the features are correlated) may prevent the algorithm from over-fitting in the context of learning local ellipsoids from a small number of examples. Let B be an upper bound of the possible expected radius. Inspired from the slack formulation of SVDD (see Problem 3.2), our algorithm, called ME^2 for Maximum Excluding Ellipsoids, can be expressed in the following form:

$$\begin{array}{ll} \underset{R,\mathbf{M},\boldsymbol{\xi}}{min} & \frac{1}{n}\sum_{i=1}^{n}\xi_i + \mu(B-R)^2 + \lambda\left\|\mathbf{M}-\mathbf{I}\right\|_{\mathcal{F}}^2, \\ s.t. & \|\mathbf{x}_i - \boldsymbol{c}\|_{\mathbf{M}}^2 \geq R - \xi_i, \quad \forall i = 1,\ldots,n, \\ & \xi_i \geq 0, \\ & B \geq R \geq 0, \end{array} \tag{3.4}$$

where $\|\mathbf{x}_i - \mathbf{c}\|_{\mathbf{M}}^2 = (\mathbf{x}_i - \mathbf{c})^T\mathbf{M}(\mathbf{x}_i - \mathbf{c})$ is the learned Mahalanobis distance between a negative example $\mathbf{x}_i$ and a positive center $\mathbf{c}$; $\boldsymbol{\xi}$ is the vector of the slack variables (allowing some constraint violations), $\mu(B-R)^2 + \lambda\left\|\mathbf{M}-\mathbf{I}\right\|_{\mathcal{F}}^2$ is a regularization term with $\mu, \lambda > 0$ the corresponding regularization parameters. Note that the upper bound B of the radius is used in $\mu(B-R)^2$ to have a convex formulation allowing us to get a unique solution. We choose two different parameters for each part of the regularization term to control the surface area of the sphere in the transformed space and the complexity of the matrix $\mathbf{M}$. The parameter λ gives the possibility to control the entries of the learned matrix, and therefore the shape/orientation of the ellipsoid. In practice, the bigger λ, the closer $\|\mathbf{x}_i - \mathbf{c}\|_{\mathbf{M}}^2$ to the Euclidean distance (i.e. the ellipsoid looks like a ball). On the other hand, the parameter μ controls the size of the learned ellipsoids.

An illustration of our algorithm is given in Figure 3.2. The points are drawn from a uniform distribution in the unit square and the positive points (in red) are randomly selected. On the right, we constrain ME^2 to learn spheres (i.e. λ is set to a large value such that $\mathbf{M}$ tends to be the identity matrix). On the left, we allow ME^2 to optimize both the orientation and the size of the ellipsoid. We can see that ME^2 can capture local peculiarities of the feature space.

It is worth noticing that λ and μ are key parameters to deal with anomaly/fraud detection in imbalanced settings. As stated in Chapter 2, standard accuracy-based loss functions are not well suited to imbalanced scenarios as they tend to favor the majority class. That is why other performance criteria like the F-measure might be preferred over the standard hinge, exponential or logistic losses. Unfortunately, as already mentioned, optimizing directly the F-measure is not feasible because of its non smoothness and non convexity. In our algorithm,

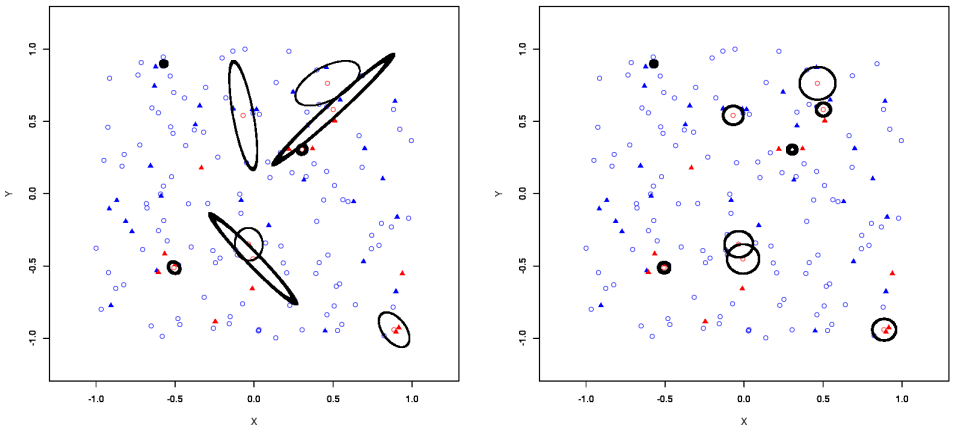

Figure 3.2: Illustration of the interest of learning ellipsoids (on the left) rather than simple spheres (on the right). Optimizing the size and the orientation of the ellipsoids allows us to better capture local peculiarities.

the parameters λ and μ have a direct impact on the F-measure by controlling the role played by the precision and the recall in the learned model.

In Figure 3.3, we present an illustration of the surface area of the ellipsoids according to increasing values of μ (on the left) and λ (on the right). We assume that the center of the ellipsoid is the point $(0, 0)$ and the data is generated on an ellipsoid on which a noise is added. As expected, the larger μ, the larger the size of the ellipsoid. On the other hand, when λ grows, the Mahalanobis distance is more and more similar to the euclidean distance. Note that there is a direct link between μ and the *recall* and *precision*. Indeed, the larger μ is, the higher the number of false positives and the smaller the number of false negatives are. The interpretation of the role played by λ is a bit more subtle. Indeed, λ controls the closeness between $\mathbf{M}$ and the identity matrix. According to the local density of the points and the need or not to stretch the ellipsoid, an increase of λ will lead to an increase or decrease of the surface area due to the use of the Identity matrix in our regularization term [1].

Note that we can establish a relationship between ME^2 and a decision tree algorithm (Quinlan, 1993). Indeed, in both cases, decision rules take the form of local geometric shapes (an ellipsoid for ME^2 and a rectangle for a decision tree). In Figure 3.4, we report on the same toy example as in Figure 3.2 the leaves learned by a decision tree algorithm containing each positive example as well as the ellipsoids optimized by ME^2. We can notice that while decision trees build axis-parallel hyperplanes to generate the leaves, ME^2 has a better expressiveness allowing to control the shape, the orientation and the size of the ellipsoids. We think that this is an interesting feature that can be favorably exploited to capture local specificities of the feature space and better estimate the density function of the positives. Coupled to

[1] In fact, there are several matrix $\mathbf{M}$ such that $\|\mathbf{M} - \mathbf{I}\|_{\mathcal{F}}^2 =$ constant. Thus, all the coefficients m_{ij} of $\mathbf{M}$ shall be on the hypersphere of equation $\sum_{i,j=1}^{d}(m_i j - \delta_{ij})^2 =$ constant where δ_{ij} is the Kronecker delta.

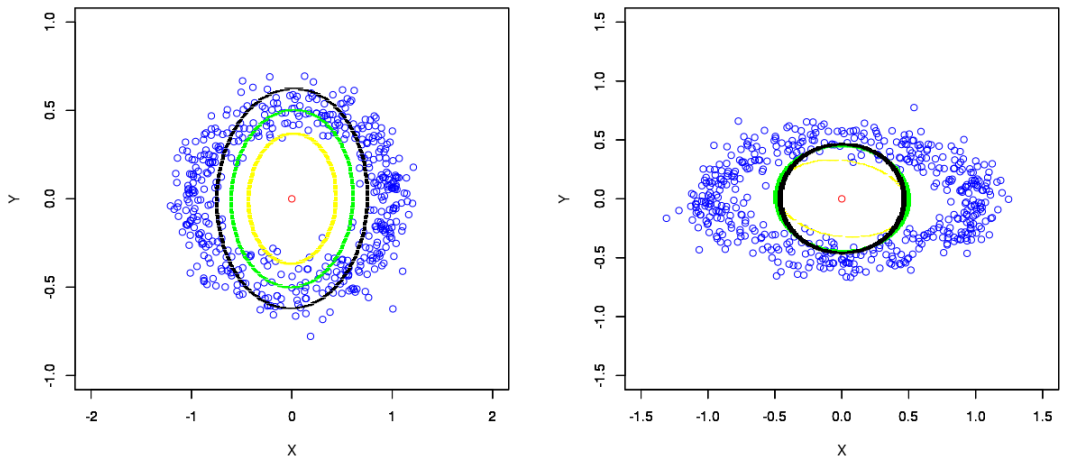

Figure 3.3: Illustration of the impact of μ (on the left) and λ (on the right) on the shape and size of the learned ellipsoids. The yellow ellipsoids correspond to small values of μ and λ, the green ellipsoids to medium values and the black ones to large values for both parameters.

the fact that we will derive generalization guarantees on the learned ellipsoids, we claim that ME^2 is a good candidate to learn from highly imbalanced data.

In the next section, we present the dual formulation of ME^2. Solving the dual problem is interesting when learning local models because it requires a few number of examples, so the problem presents a few number of variables.

3.3.2 Dual Version and Closed-Form Solution

Adding the constraint to the quantity we have to optimize, the Problem 3.4 can also be expressed in its dual form as:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\alpha}, \beta, \delta, \boldsymbol{\gamma}, R, \boldsymbol{\xi}, \mathbf{M}) \;=\; & \frac{1}{n}\sum_{i=1}^{n}\xi_i + \mu(B-R)^2 - \sum_{i=1}^{n}\alpha_i\left(\|\mathbf{x}_i - \mathbf{c}\|^2_{\mathbf{M}} - R + \xi_i\right) \\ & - \sum_{i=1}^{n}\gamma_i\xi_i + \lambda\left\|\mathbf{M} - \mathbf{I}\right\|^2_{\mathcal{F}} - \beta R + \delta(R-B),\end{aligned} \tag{3.5}$$

where $\boldsymbol{\alpha} = (\alpha_i)_{i=1,\dots,n}$, are the dual variables associated to the constraint $R - \xi_i - \|\mathbf{x}_i - \mathbf{c}\| \leq 0$; $\boldsymbol{\gamma} = (\gamma_i)_{i=1,\dots,n}$ the one associated to $-\xi_i \leq 0$; β and δ to $-R < 0$ and $R - B < 0$.

We aim to rewrite the primal variables as a function of the dual variables. To do this, we compute the derivatives of (3.6) with respect to the primal variables.

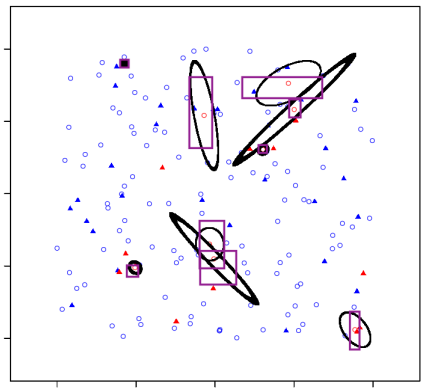

Figure 3.4: Boundaries of the decision rules with ME^2 and a decision tree algorithm. The expressiveness of ME^2 is better to capture local specificities of the density function of the positives.

$$\nabla_R \mathcal{L} = \sum_{i=1}^{n} \alpha_i + 2\mu R - 2\mu B - \beta + \delta, \tag{3.6}$$

$$\nabla_{\xi_i} \mathcal{L} = \frac{1}{n} - \gamma_i - \alpha_i, \quad \forall i = 1, \ldots, n, \tag{3.7}$$

The derivative of the Frobenius norm is:

$$\frac{\partial \left\| \mathbf{M} - \mathbf{I} \right\|_{\mathcal{F}}^2}{\partial \mathbf{M}} = 2(\mathbf{M} - \mathbf{I}).$$

This last equality implies:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{M}} = -\sum_{i=1}^{n} \alpha_k \left[(\mathbf{x}_k - \mathbf{c})(\mathbf{x}_k - \mathbf{c})^T \right] + 2\lambda(\mathbf{M} - \mathbf{I}). \tag{3.8}$$

We recall that the dual formulation is defined by taking the infimum of $\mathcal{L}(R, \mathbf{M}, \boldsymbol{\xi})$ and a necessary condition for which the minimum is reached is when the derivative with respect to the primal variables are equal to 0. This leads us to the following implications:

$$(3.6) \Rightarrow R = \frac{\beta - \delta + 2\mu B - \sum_{i=1}^{n} \alpha_i}{2\mu},$$

$$(3.7) \Rightarrow 0 \leq \alpha_i \leq \frac{1}{n},$$

$$(3.8) \Rightarrow \mathbf{M} = \mathbf{I} + \frac{1}{2\lambda} \sum_{i=1}^{n} \alpha_k (\mathbf{x}_k - \mathbf{c})(\mathbf{x}_k - \mathbf{c})^T.$$

The last equality shows that $\mathbf{M}$ is, by construction, *positive semi definite* as it is a convex combination of *positive semi definite matrices* of rank 1. Furthermore, because of the addition of the Identity matrix to the previous matrices, $\mathbf{M}$ is *positive definite* (PD). Fulfilling the PD constraint for free is very important because it prevents the algorithm to perform a singular value decomposition (in $\mathcal{O}(d^3)$) at each step of the gradient descent. Note also that $\mathbf{M}$ can be seen as an *optimized* covariance-matrix for which the weight α_k of each negative instance is learned.

Let us now insert the closed form solution of $\mathbf{M}$ and R in (3.6) in order to have the dual formulation of Problem 3.4.

Proposition 3.1. *[Dual Formulation] Let $\{\mathbf{x}_i\}_{i=1}^n$ be a set of n negative examples and $\mathbf{c}$ a positive example. We denote by $\mathbf{G}$ the Gram matrix defined by $\mathbf{G}_{ij} = \langle(\mathbf{x}_i - \mathbf{c}), (\mathbf{x}_j - \mathbf{c})\rangle$ and by $\mathbf{G}'$ the matrix of the Hadamard [2] product of $\mathbf{G}$ with itself. Given the derivatives* (3.6) *to* (3.8)*, the dual formulation of the optimization problem* (3.4) *is defined by:*

$$\begin{array}{ll} \underset{\boldsymbol{\alpha},\beta,\delta}{min} & \boldsymbol{\alpha}^T \left(\frac{1}{4\lambda}\mathbf{G}' + \frac{1}{4\mu}\mathbb{1}_{n\times n}\right)\boldsymbol{\alpha} + \frac{\beta^2}{4\mu} + \frac{\delta^2}{4\mu} + \\ & \boldsymbol{\alpha}^T \left(diag(\mathbf{G}) - \left(B + \frac{\beta}{2\mu} - \frac{\delta}{2\mu}\right)\mathbb{1}_n\right) + \beta\left(B - \frac{\delta}{2\mu}\right), \\ s.t. & 0 \leq \alpha_i \leq \frac{1}{n}, \quad \forall i = 1, \ldots, n, \\ & \beta, \delta \geq 0, \end{array}$$

where $\mathbb{1}_n$ (respectively $\mathbb{1}_{n\times n}$) represents a vector of length n (respectively a matrix of size $n \times n$) where entries are equal to 1*.*

Proof. For the sake of clarity, let us first restate the Lagrangian and the expression of both $\mathbf{M}$ and R.

$$\mathcal{L}(\boldsymbol{\alpha}, \beta, \delta, \boldsymbol{\gamma}, R, \boldsymbol{\xi}, \mathbf{M}) = \frac{1}{n}\sum_{i=1}^n \xi_i + \mu(B-R)^2 - \sum_{i=1}^n \gamma_i\xi_i - \sum_{i=1}^n \alpha_i\left(\|\mathbf{x}_i - \mathbf{c}\|_{\mathbf{M}}^2 - R + \xi_i\right) + \lambda\|\mathbf{M} - \mathbf{I}\|_{\mathcal{F}}^2 - \beta R + \delta(R - B),$$

and

$$\begin{aligned} R &= \frac{\beta - \delta + 2\mu B - \sum_{i=1}^n \alpha_i}{2\mu}, \\ \mathbf{M} &= \mathbf{I} + \frac{1}{2\lambda}\sum_{i=1}^n \alpha_k(\mathbf{x}_k - \mathbf{c})(\mathbf{x}_k - \mathbf{c})^T. \end{aligned}$$

We now inject the expression of $\mathbf{M}$ in the expression of the Lagrangian. We first set $A = \sum_{i=1}^n \alpha_i\|\mathbf{x}_i - \mathbf{c}\|_{\mathbf{M}}^2$ and $N = Tr((\mathbf{M}-\mathbf{I})^T(\mathbf{M}-\mathbf{I}))$ and we develop these two expressions.

[2] The Hadamard product of two matrices is the entry-wise product.

The weighted sum of Mahalanobis distances A can be rewritten:

$$\begin{aligned} A &= \sum_{i=1}^{n} \alpha_i (\mathbf{x}_i - \mathbf{c})^T (\mathbf{x}_i - \mathbf{c}) + \frac{1}{2\lambda} \sum_{i=1}^{n} \sum_{k=1}^{n} \alpha_i \alpha_k (\mathbf{x}_i - \mathbf{c})^T (\mathbf{x}_k - \mathbf{c})(\mathbf{x}_k - \mathbf{c})^T (\mathbf{x}_i - \mathbf{c}), \\ &= \boldsymbol{\alpha}^T \mathrm{diag}(\mathbf{G}) + \frac{1}{2\lambda} \sum_{i=1}^{n} \sum_{k=1}^{n} \alpha_i \alpha_k \mathbf{G}_{ik} \mathbf{G}_{ki}, \\ A &= \boldsymbol{\alpha}^T \mathrm{diag}(\mathbf{G}) + \frac{1}{2\lambda} \boldsymbol{\alpha}^T \mathbf{G}' \boldsymbol{\alpha}, \end{aligned}$$

where $\mathbf{G}$ is the Gram matrix defined by $G_{ij} = \langle (\mathbf{x}_i - \mathbf{c}), (\mathbf{x}_j - \mathbf{c}) \rangle$ and $\mathbf{G}'$ is the Hadamard product of $\mathbf{G}$ with itself. Because $\mathbf{G}$ is PSD, so is $\mathbf{G}'$.

Let us now focus on the expression of N. From now on, we set $\mathbf{v}_k = \mathbf{x}_k - \mathbf{c}$ for convenience. Let us denote by $v_{ki} = (\mathbf{x}_k - \mathbf{c})_i$ the i^{th} element of the vector $\mathbf{x}_k - \mathbf{c}$.

$$\begin{aligned} N &= \frac{1}{4\lambda^2} \sum_{i=1}^{d} \sum_{j=1}^{d} \left(\sum_{k=1}^{n} \alpha_k v_{ki} v_{kj} \right)^2, \\ &= \frac{1}{4\lambda^2} \sum_{k=1}^{n} \alpha_k^2 \left(\sum_{i=1}^{d} \sum_{j=1}^{d} v_{ki}^2 v_{kj}^2 \right) + \frac{2}{4\lambda^2} \sum_{k>l}^{n} \left(\alpha_k \alpha_l \sum_{i=1}^{d} \sum_{j=1}^{d} v_{ki} v_{kj} v_{li} v_{lj} \right), \\ &= \frac{1}{4\lambda^2} \sum_{k=1}^{n} \alpha_k^2 \left(\sum_{i=1}^{d} v_{ki}^2 \sum_{j=1}^{d} v_{kj}^2 \right) + \frac{2}{4\lambda^2} \sum_{k>l}^{n} \left(\alpha_k \alpha_l \sum_{i=1}^{d} v_{ki} v_{li} \sum_{j=1}^{d} v_{kj} v_{lj} \right), \\ &= \frac{1}{4\lambda^2} \sum_{k=1}^{n} \alpha_k^2 \mathbf{G}_{kk}^2 + \frac{2}{4\lambda^2} \sum_{k>l}^{n} \left(\alpha_k \alpha_l \mathbf{G}_{kl} \mathbf{G}_{lk} \right), \\ N &= \frac{1}{4\lambda^2} \boldsymbol{\alpha}^T \mathbf{G}' \boldsymbol{\alpha}. \end{aligned}$$

We do the same by replacing R by its expression in (3.6). For the sake of simplicity, we only consider the terms of the Lagrangian where R appears and set $D = \mu(B - R)^2 + R \sum_{k=1}^{n} \alpha_k + \delta(R - B) - \beta R$. We obtain:

$$\begin{aligned} D &= \mu \left(B - \frac{\beta - \delta + 2\mu B - \sum_{k=1}^{n} \alpha_k}{2\mu} \right)^2 + \sum_{k=1}^{n} \alpha_k \left(\frac{\beta - \delta + 2\mu B - \sum_{k=1}^{n} \alpha_k}{2\mu} \right) \\ &\quad + \delta \left(\frac{\beta - \delta + 2\mu B - \sum_{k=1}^{n} \alpha_k}{2\mu} \right) - \beta \left(\frac{\beta - \delta + 2\mu B - \sum_{k=1}^{n} \alpha_k}{2\mu} \right). \end{aligned}$$

By developing each term we have:

$$D = \frac{1}{4\mu} \left[\left(\sum_{k=1}^{n} \alpha_k \right)^2 + \beta^2 + \delta^2 - 2\delta\beta - 2\beta \sum_{k=1}^{n} \alpha_k + 2\delta \sum_{k=1}^{n} \alpha_k \right]$$

$$+\frac{1}{2\mu}\left[-\left(\sum_{k=1}^{n}\alpha_k\right)^2+\beta\sum_{k=1}^{n}\alpha_k-\delta\sum_{k=1}^{n}\alpha_k+B\sum_{k=1}^{n}\alpha_k\right]+\frac{1}{2\mu}\left[\delta\beta-\delta^2-\delta\sum_{k=1}^{n}\alpha_k\right]$$
$$+\frac{1}{2\mu}\left[-\beta^2+\delta\beta-B\beta+\beta\sum_{k=1}^{n}\alpha_k\right].$$

We reduce and factorize this expression to have:

$$D = -\frac{1}{4\mu}\left[\left(\sum_{k=1}^{n}\alpha_k\right)^2+\beta^2+\delta^2\right]+\sum_{k=1}^{n}\left(B+\frac{\beta}{2\mu}-\frac{\delta}{2\mu}\right)+\beta\left(-B+\frac{\delta}{2\mu}\right).$$

We can now express our Lagrangian with respect to α, β and δ only as:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\alpha},\beta,\delta) &= -\boldsymbol{\alpha}^T\mathrm{diag}(\mathbf{G})-\frac{1}{4\lambda}\boldsymbol{\alpha}^T\mathbf{G}'\boldsymbol{\alpha}-\frac{1}{4\mu}\left[\left(\sum_{k=1}^{n}\alpha_k\right)^2+\beta^2+\delta^2\right]\\ &\quad+\left(B+\frac{\beta}{2\mu}-\frac{\delta}{2\mu}\right)\sum_{k=1}^{n}\alpha_k+\beta\left(-B+\frac{\delta}{2\mu}\right).\end{aligned}$$

We then have the dual Problem by minimizing the opposite of the above Lagrangian with the associated constraints. □

The dual formulation has the main advantage to be easier to solve than the primal one in high dimensional spaces. While the use of such a model can be a barrier in the presence of a very large amount of data, it should be remembered that we only learn local models, which, therefore require a less number of data.

The next section is dedicated to the derivation of generalization guarantees of the proposed algorithm.

3.4 Generalization Guarantees

One of our main contributions takes the form of a generalization guarantee on the algorithm ME^2. Since we learn a local model from a subset of training examples (i.e. the positive example of interest and the negative examples in its close neighborhood), we need to prove the ability of ME^2 to perform well in generalization - that is - to exclude correctly new negative instances from the learned ellipsoid centered at this positive example. To do this, we derive in this section a generalization bound according to the theoretical framework of uniform stability.

3.4.1 Uniform Stability

In this section, we briefly restate the definition of stability and the generalization bound based on this notion.

Roughly speaking, an algorithm is *stable* if its output, in terms of difference between losses, does not change significantly under a small modification of the training sample. This variation must be bounded in $O(1/n)$ in terms of infinite norm where n is the size of the training set S *i.i.d.* from an unknown distribution $\mathcal{D}$.

Definition 3.1. *[Definition 6 (Bousquet and Elisseeff, 2002)] A learning algorithm $\mathcal{A}$ has a uniform stability in $\frac{\beta}{n}$ with respect to a loss function ℓ and parameter set θ, with β a positive constant if:*

$$\forall S,\ \forall i,\ 1 \leq i \leq n,\ \sup_{\mathbf{x}} |\ell(\theta_S, \mathbf{x}) - \ell(\theta_{S^i}, \mathbf{x})| \leq \frac{\beta}{n},$$

where S is a learning sample of size n, θ_S the model parameters learned from S, θ_{S^i} the model parameters learned from the sample S^i obtained by replacing the i^{th} example $\mathbf{x}_i$ from S by another example $\mathbf{x}'_i$ independent from S and drawn from $\mathcal{D}$. $\ell(\theta_S, \mathbf{x})$ is the loss suffered at $\mathbf{x}$.

In this definition, S^i represents the notion of small modification of the training sample. From Definition 3.1, one can obtain the following generalization bound[3]:

Theorem 3.1. *[Theorem 12 (Bousquet and Elisseeff, 2002)] Let $\delta > 0$ and $n > 1$. For any algorithm with uniform stability β/n, using a loss function bounded by K, with probability at least $1 - \delta$ over the random draw of S:*

$$\mathcal{R}(\theta_S) \leq \mathcal{R}_S(\theta_S) + \frac{2\beta}{n} + (4\beta + K)\sqrt{\frac{\ln 1/\delta}{2n}},$$

where $\mathcal{R}(\cdot)$ is the true risk and $\mathcal{R}_S(\cdot)$ its empirical estimate over S.

3.4.2 Generalization Bound

Given a centroid $\mathbf{c}$ (representing a positive instance) and a learning sample $S = \{\mathbf{x}_i\}_{i=1}^n$ of negative instances drawn *i.i.d.* from an unknown probability distribution $\mathcal{X}^-$ (i.e. the marginal distribution of the negative examples), the set of parameters to be learned by ME^2 is the pair $(R, \mathbf{M})$. For convenience, we consider the following optimization problem that is equivalent to Problem 3.4:

$$\begin{array}{ll} \min\limits_{R,\mathbf{M}} & \frac{1}{n}\sum_{i=1}^n \ell(R, \mathbf{M}, \mathbf{x}_i) + \mu(B - R)^2 + \lambda \left\|\mathbf{M} - \mathbf{I}\right\|_{\mathcal{F}}, \\ s.t. & B \geq R \geq 0, \end{array}$$

where $\ell(\cdot)$ represents the loss such that $\ell(R, \mathbf{M}, \mathbf{x}_i) = \left[R^2 - \|\mathbf{x}_i - \mathbf{c}\|^2_{\mathbf{M}}\right]_+$ with $[\cdot]_+$ the hinge loss function: $[a]_+ = \max(a, 0)$.

[3] If this result was proposed in the context of regression and classification tasks, the proof techniques are general enough - by considering generic bounded and lipschitz losses - so that it also holds for the setting considered in this section.

The true risk is defined by $\mathcal{R}(\mathbf{M}, R) = \mathbb{E}_{\mathbf{x}\sim\mathcal{X}^-}[\ell(\mathbf{M}, R, \mathbf{x})]$ and its empirical estimate over the sample S by $\mathcal{R}_S(\mathbf{M}, R) = \frac{1}{n}\sum_{i=1}^{n}\ell(\mathbf{M}, R, \mathbf{x}_i)$. We also denote the regularization term as $N(\mathbf{M}, R) = \mu(B - R)^2 + \lambda\|\mathbf{M} - \mathbf{I}\|_{\mathcal{F}}^2$. Let us define K such that $\max_{\mathbf{x}\sim\mathcal{X}^-}|\mathbf{x}| \leq K$ and $|\mathbf{c}| \leq K$. F_S represents the objective function to be minimized, *i.e.*:

$$F_S(\mathbf{M}, R) = \mathcal{R}_S(\mathbf{M}, R) + N(\mathbf{M}, R).$$

Note here that it can easily be checked that our loss function ℓ is convex with respect to $\mathbf{M}$ and R. To prove a generalization bound on our algorithm ME^2, we need to prove that our setting verifies the definition of uniform stability. For this purpose, we first prove that our loss function is actually k-lipschitz with respect to its first two arguments.

Lemma 3.1. *The loss ℓ is k-lipschitz w.r.t to $\mathbf{M}$ and R with $k = \max(1, 4K^2)$, i.e. for any $(\mathbf{M}, R) \in \mathbb{S}^+ \times \mathbb{R}^+$, $(\mathbf{M}', R') \in \mathbb{S}^+ \times \mathbb{R}^+$ and $\forall \mathbf{x} \in \mathbb{R}^d$:*

$$\left|\ell(\mathbf{M}, R, \mathbf{x}) - \ell(\mathbf{M}', R', \mathbf{x})\right| \leq k\left\|(\mathbf{M}, R) - (\mathbf{M}', R')\right\|,$$

where $\|(\mathbf{M}, R) - (\mathbf{M}', R')\| = |R - R'| + \|\mathbf{M} - \mathbf{M}'\|_{\mathcal{F}}$.

Proof.

$$\begin{aligned}
\left|\ell((\mathbf{M}, R), \mathbf{x}) - \ell((\mathbf{M}', R'), \mathbf{x})\right| &= \left|\left[R - \|\mathbf{x} - \mathbf{c}\|_{\mathbf{M}}^2\right]_+ - \left[R' - \|\mathbf{x} - \mathbf{c}\|_{\mathbf{M}'}^2\right]_+\right|, \\
&\leq \left|R - R'\right| + \left|\|\mathbf{x} - \mathbf{c}\|_{\mathbf{M}}^2 - \|\mathbf{x} - \mathbf{c}\|_{\mathbf{M}'}^2\right|), \qquad (3.9) \\
&= \left(\left|R - R'\right| + \left|(\mathbf{x} - \mathbf{c})^T(\mathbf{M} - \mathbf{M}')(\mathbf{x} - \mathbf{c})\right|\right), \\
&\leq \left(\left|R - R'\right| + 4K^2\left\|\mathbf{M} - \mathbf{M}'\right\|_{\mathcal{F}}\right), \qquad (3.10) \\
&\leq \max(1, 4K^2)\left(\left|R - R'\right| + \left\|\mathbf{M} - \mathbf{M}'\right\|_{\mathcal{F}}\right).
\end{aligned}$$

Line (3.9) uses the fact that the hinge loss is 1-lipschitz and a property of the absolute value. Line (3.10) can be obtained by the Cauchy-Schwarz inequality[4] and classic properties on norms. □

We now need a technical lemma on the objective function F_S.

Lemma 3.2. *Let S be a learning sample, let F_S and F_{S^i} be two objective functions with respect to two samples S and S^i (as defined in Definition 3.1) and let $(\mathbf{M}, R)$ and $(\mathbf{M}^i, R^i)$ be their respective minimizers. We also define $\Delta(\mathbf{M}, R) = (\mathbf{M}^i, R^i) - (\mathbf{M}, R)$ and recall that $N(\mathbf{M}, R) = \mu(B - R)^2 + \lambda\|\mathbf{M} - \mathbf{I}\|_{\mathcal{F}}^2$. For all $t \in [0, 1]$, we have:*

$$N(\mathbf{M}, R) - N((\mathbf{M}, R) + t\Delta(\mathbf{M}, R)) + N((\mathbf{M}^i, R^i) - N((\mathbf{M}^i, R^i) - t\Delta(\mathbf{M}, R)) \leq \frac{2t\max(1, 4K^2)}{n}\|\Delta(\mathbf{M}, R)\|.$$

[4]For any vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ we have $\langle \mathbf{x}, \mathbf{y}\rangle \leq \|\mathbf{x}\|_2\|\mathbf{y}\|_2$.

Proof. Since ℓ (the hinge loss) is convex, so is the empirical risk and thus for all $t \in [0,1]$ we have the two following inequalities:

$$\mathcal{R}_{S^i}((\mathbf{M},R)+t\Delta(\mathbf{M},R)) - \mathcal{R}_{S^i}(\mathbf{M},R) \leq t\mathcal{R}_{S^i}(\mathbf{M}^i,R^i) - t\mathcal{R}_{S^i}(\mathbf{M},R)$$

and

$$\mathcal{R}_{S^i}((\mathbf{M}^i,R^i)-t\Delta(\mathbf{M},R)) - \mathcal{R}_{S^i}(\mathbf{M}^i,R^i) \leq t\mathcal{R}_{S^i}(\mathbf{M},R) - t\mathcal{R}_{S^i}(\mathbf{M}^i,R^i).$$

We get the second inequality by swapping the role of $(\mathbf{M},R)$ and $(\mathbf{M}^i,R^i)$. If we sum these two inequalities, the right hand side vanishes and we obtain:

$$\mathcal{R}_{S^i}((\mathbf{M},R)+t\Delta(\mathbf{M},R))-\mathcal{R}_{S^i}(\mathbf{M},R)+\mathcal{R}_{S^i}((\mathbf{M}^i,R^i)-t\Delta(\mathbf{M},R))-\mathcal{R}_{S^i}(\mathbf{M}^i,R^i) \leq 0. \quad (3.11)$$

By assumption on $(\mathbf{M},R)$ and $(\mathbf{M}^i,R^i)$ we have:

$$\begin{aligned} F_S(\mathbf{M},R) - F_S((\mathbf{M},R)+t\Delta(\mathbf{M},R)) &\leq 0, \\ F_{S^i}(\mathbf{M}^i,R^i) - F_{S^i}((\mathbf{M}^i,R^i)-t\Delta(\mathbf{M},R)) &\leq 0, \end{aligned}$$

then, summing the two previous inequalities and using (3.11), we get:

$$\begin{aligned} &\mathcal{R}_{S^i}((\mathbf{M},R)+t\Delta(\mathbf{M},R)) - \mathcal{R}_S((\mathbf{M},R)+t\Delta(\mathbf{M},R)) - \mathcal{R}_{S^i}(\mathbf{M},R) + \mathcal{R}_S(\mathbf{M},R) \\ &\quad + \mu[(B-R)^2 + (B-R^i)^2 - (B-(R+t\Delta R))^2 - (B-(R^i-t\Delta R))^2] \\ &\quad + \lambda[\|\mathbf{M}-\mathbf{I}\|_F^2 + \|\mathbf{M}^i-\mathbf{I}\|_F^2 - \|\mathbf{M}+t\Delta\mathbf{M}-\mathbf{I}\|_F^2 - \|\mathbf{M}^i-t\Delta\mathbf{M}-\mathbf{I}\|_F^2] \leq 0. \quad (3.12) \end{aligned}$$

We now focus on the first part of the previous inequation. For the sake of simplicity, let us set:

$$H = \mathcal{R}_{S^i}((\mathbf{M},R)+t\Delta(\mathbf{M},R)) - \mathcal{R}_{S^i}(\mathbf{M},R) - \mathcal{R}_S((\mathbf{M},R)+t\Delta(\mathbf{M},R)) + \mathcal{R}_S(\mathbf{M},R).$$

We will use Lemma 3.1 to bound this term:

$$\begin{aligned} H &\leq \left|\mathcal{R}_{S^i}((\mathbf{M},R)+t\Delta(\mathbf{M},R)) - \mathcal{R}_{S^i}(\mathbf{M},R) - \mathcal{R}_S((\mathbf{M},R)+t\Delta(\mathbf{M},R)) + \mathcal{R}_S(\mathbf{M},R)\right|, \\ &\leq \left|\frac{1}{n}\sum_{\mathbf{x}_i\in S^i}\ell((\mathbf{M},R)+t\Delta(\mathbf{M},R),\mathbf{x}_i) - \frac{1}{n}\sum_{\mathbf{x}_i\in S}\ell((\mathbf{M},R)+t\Delta(\mathbf{M},R),\mathbf{x}_i)\right. \\ &\quad \left. + \frac{1}{n}\sum_{\mathbf{x}_i\in S}\ell((\mathbf{M},R),\mathbf{x}_i) - \frac{1}{n}\sum_{\mathbf{x}_i\in S^i}\ell((\mathbf{M},R),\mathbf{x}_i)\right|, \\ &\leq \frac{1}{n}\left|\ell((\mathbf{M},R)+t\Delta(\mathbf{M},R),\mathbf{x}_i) - \ell((\mathbf{M},R),\mathbf{x}_i)\right. \\ &\quad \left. -\ell((\mathbf{M},R)+t\Delta(\mathbf{M},R),\mathbf{x}_i') + \ell((\mathbf{M},R),\mathbf{x}_i')\right|, \\ &\leq \frac{1}{n}\left|\ell((\mathbf{M},R)+t\Delta(\mathbf{M},R),\mathbf{x}_i) - \ell((\mathbf{M},R),\mathbf{x}_i)\right| \\ &\quad + \frac{1}{n}\left|\ell((\mathbf{M},R)+t\Delta(\mathbf{M},R),\mathbf{x}_i') - \ell((\mathbf{M},R),\mathbf{x}_i')\right|, \\ H &\leq \frac{2t\max(1,4K^2)}{n}\|\Delta(\mathbf{M},R)\|. \end{aligned}$$

We have successively applied the definition of the empirical risk and triangle inequality to get the previous inequalities. The last one is obtained using Lemma 3.1. □

With this result, we are able to prove the uniform stability property of our algorithm.

Proposition 3.2. *There exists a positive constant η such that the algorithm ME^2 is uniformly stable with $\beta = \dfrac{2(\max(1, 4K^2))^2}{\eta \min(\mu, \lambda)}$.*

Proof. Setting $t = \dfrac{1}{2}$, from previous Lemma, we have:

$$\mu f_1(R) + \lambda f_2(\mathbf{M}) \leq \frac{\max(1, 4K^2)}{n} \|\Delta(\mathbf{M}, R)\|, \tag{3.13}$$

where

$$f_1(R) = (B - R)^2 + (B - R^i)^2 - (B - (R + \frac{1}{2}(R^i - R)))^2 - (B - (R^i - \frac{1}{2}(R^i - R)))^2 \tag{3.14}$$

and

$$f_2(\mathbf{M}) = \|\mathbf{M} - \mathbf{I}\|_{\mathcal{F}}^2 - \left\|\mathbf{M} + \frac{1}{2}(\mathbf{M}^i - \mathbf{M}) - \mathbf{I}\right\|_{\mathcal{F}}^2 + \left\|\mathbf{M}^i - \mathbf{M}\right\|_{\mathcal{F}}^2 - \left\|M^i - \frac{1}{2}(\mathbf{M}^i - \mathbf{M}) - \mathbf{I}\right\|_{\mathcal{F}}^2. \tag{3.15}$$

In the following, we will also set $\delta_{jk} = \begin{cases} 1 & \text{if} \quad j = k, \\ 0 & \text{otherwise.} \end{cases}$

By developing Equation (3.14) we get:

$$\begin{aligned} f_1(R) &= (B - R)^2 + (B - R^i)^2 - 2(B - \frac{1}{2}(R + R^i))^2, \\ &= 2B^2 - 2B(R + R^i) + R^2 + R^{i2} - 2\left(B^2 - B(R + R^i) + \frac{1}{4}(R + R^i)^2\right), \\ f_1(R) &= \frac{1}{2}\left(R - R^i\right)^2. \end{aligned}$$

Similarly for Equation (3.15), we have:

$$\begin{aligned} f_2(\mathbf{M}) &= \|\mathbf{M} - \mathbf{I}\|_{\mathcal{F}}^2 + \left\|\mathbf{M}^i - \mathbf{I}\right\|_{\mathcal{F}}^2 - \left\|\frac{1}{2}(\mathbf{M} + \mathbf{M}^i) - \mathbf{I}\right\|_{\mathcal{F}}^2 - \left\|\frac{1}{2}(\mathbf{M} + \mathbf{M}^i) - \mathbf{I}\right\|_{\mathcal{F}}^2, \\ &= \|\mathbf{M} - \mathbf{I}\|_{\mathcal{F}}^2 + \left\|\mathbf{M}^i - \mathbf{I}\right\|_{\mathcal{F}}^2 - \frac{1}{2}\left\|(\mathbf{M} - \mathbf{I}) + (\mathbf{M}^i - \mathbf{I})\right\|_{\mathcal{F}}^2, \\ &= \sum_{j,k=1}^{d} \left((\mathbf{M}_{jk} - \delta_{jk})^2 + (\mathbf{M}_{jk}^i - \delta_{jk})^2 - \frac{1}{2}(\mathbf{M}_{jk} + \mathbf{M}_{jk}^i - 2\delta_{jk})^2\right), \\ &= \frac{1}{2}\left(\sum_{j,k=1}^{d} \left[(\mathbf{M}_{jk} - \delta_{jk})^2 + (\mathbf{M}_{jk}^i - \delta_{jk})^2\right]\right) - \sum_{j,k=1}^{d} \left[(\mathbf{M}_{jk} - \delta_{jk})(\mathbf{M}_{jk}^i - \delta_{jk})\right], \\ &= \frac{1}{2}\sum_{j,k=1}^{d} \left[(\mathbf{M}_{jk} - \delta_{jk})^2 - (\mathbf{M}_{jk}^i - \delta_{jk})^2\right], \end{aligned}$$

$$= \frac{1}{2}\sum_{j,k=1}^{d}(\mathbf{M}_{jk}-\mathbf{M}^i_{jk})^2,$$

$$f_2(\mathbf{M}) = \frac{1}{2}\left\|\mathbf{M}-\mathbf{M}^i\right\|^2_{\mathcal{F}}.$$

We can then write the Inequality (3.13) as:

$$\mu(R^i-R)^2+\lambda\left\|\mathbf{M}^i-\mathbf{M}\right\|^2_{\mathcal{F}} \leq \frac{2\max(1,4K^2)}{n}\|\Delta(\mathbf{M},R)\|. \tag{3.16}$$

Recall that: $\|\Delta(\mathbf{M},R)\| = \left|R-R^i\right| + \left\|\mathbf{M}^i-\mathbf{M}\right\|_{\mathcal{F}}$. Because we are working in a space of finite dimension, all the norms are equivalent, *i.e.* there exists a positive constant η such that, $\forall(R,R^i)\in\mathbb{R}^+$, $\forall(\mathbf{M},\mathbf{M}^i)\in\mathbb{S}^+\times\mathbb{S}^+$ we have:

$$\eta(\left|R-R^i\right| + \left\|\mathbf{M}-\mathbf{M}^i\right\|_{\mathcal{F}})^2 \leq (R-R^i)^2+\left\|\mathbf{M}-\mathbf{M}^i\right\|^2_{\mathcal{F}}. \tag{3.17}$$

Finally, combining the previous inequalites, we have:

$$\min(\mu,\lambda)\eta\,\|\Delta(\mathbf{M},R)\|^2 \leq \min(\mu,\lambda)[(R^i-R)^2+\left\|\mathbf{M}^i-\mathbf{M}\right\|^2_{\mathcal{F}}] \leq \frac{2\max(1,4K^2)}{n}\|\Delta(\mathbf{M},R)\|,$$

thus,

$$\|\Delta(\mathbf{M},R)\| \leq \frac{2\max(1,4K^2)}{\eta\min(\mu,\lambda)}.$$

Starting from the left-hand side of Definition 3.1 and applying Lemma 3.1 once and the previous inequality, leads to our final result.

$$\begin{aligned}\left|\ell((\mathbf{M},R),\mathbf{x})-\ell((\mathbf{M}^i,R^i),\mathbf{x})\right| &\leq \max(1,4K^2)\|\Delta(\mathbf{M},R)\|,\\ &\leq \frac{2}{n\eta\min(\mu,\lambda)}(\max(1,4K^2))^2.\end{aligned}$$

□

To prove the generalization bound, it remains to show that our loss function ℓ is bounded. Because of the use of a hinge loss function, we directly have the following bound for our loss function ℓ:

$$\ell((\mathbf{M},R),\mathbf{x}) \leq B.$$

Given the stability constant and the fact that the loss is bounded, using Theorem 3.1, we obtain our final result:

Theorem 3.2. *Let $\delta>0$ and $n>1$, there exists a constant $\eta>0$, such that with probability at least $1-\delta$ over the random draw over S, we have for any $(\mathbf{M},R)$ solution of Problem 3.4.2:*

$$\mathcal{R}(\mathbf{M},R) \leq \mathcal{R}_S(\mathbf{M},R) + \frac{4(\max(1,4K^2))^2}{n\eta\min(\mu,\lambda)} + \left(\frac{8(\max(1,4K^2))^2}{\eta\min(\mu,\lambda)}+B\right)\sqrt{\frac{\ln 1/\delta}{2n}}.$$

Proof. We simply combine Proposition 3.2, the previous remark and Theorem 3.1. □

This generalization bound holds for any positive center $\mathbf{c}$. If one has p positive centers, by the union bound, we can extend the previous result for each of the p centers with probability $1-\delta/p$ showing that the models output can control negative instances with high probability. We can notice here that even if the bound suggests an independence from the number of features d, it turns out that the dimensionality of the feature space is intrinsically captured by the hyper-parameters μ and λ which is consistent with the fact that the dependency on d has already been noted for any Mahalanobis-based metric learning algorithm (Verma and Branson, 2015).

The presented bound depends on an unknown parameter $\eta > 0$. However, if the value of η is closed to 0, the bound will be loose. In the following section, we briefly discuss a lower bound on η.

3.4.3 About the Parameter η

We can note that all the parameters of our algorihm ME^2 are involved in the previous generalization bound (that is, B, μ, λ). It turns out that the constant of uniform stability, thus our bound, also depends on the parameter η which appears when we use the fact that two norms are equivalent in a space of finite dimension (see the proof of Proposition 3.2). To be usable in practice (for example, to check its empirical convergence on a given problem), this bound has to get rid of η.

We first rewrite the inequation in which the parameter η appears in the above mentioned proof.

$$\eta(\left|R - R^i\right| + \left\|\mathbf{M} - \mathbf{M}^i\right\|_{\mathcal{F}})^2 \leq (R - R^i)^2 + \left\|\mathbf{M} - \mathbf{M}^i\right\|_{\mathcal{F}}^2 .$$

Let us set $\mathbf{M} - \mathbf{M}^i = \Delta\mathbf{M}$ and $R - R^i = \Delta R$. We can rewrite the previous inequality as:

$$\begin{aligned} & \eta(|\Delta R| + \|\Delta\mathbf{M}\|_{\mathcal{F}})^2 \leq (\Delta R)^2 + \|\Delta\mathbf{M}\|_{\mathcal{F}}^2 , \\ \Leftrightarrow \quad & \eta \leq \frac{(\Delta R)^2 + \|\Delta\mathbf{M}\|_{\mathcal{F}}^2}{(|\Delta R| + \|\mathbf{M}\|_{\mathcal{F}})^2}. \end{aligned}$$

It is worth noticing that the larger η, the tighter the bound. Therefore, we need to find the largest value of η which satisfies the previous inequality. This worst optimal value for η is then given by the minimum of the function f defined by $f(x, y) = \dfrac{x^2 + y^2}{(x + y)^2}$. It is well known that for all x and y we have:

$$x^2 + y^2 \geq 2xy \qquad \text{and thus} \qquad (x + y)^2 \leq 2(x^2 + y^2).$$

We get:

$$f(x, y) = \frac{x^2 + y^2}{(x + y)^2} \geq \frac{x^2 + y^2}{2(x^2 + y^2)} \geq \frac{1}{2} = \eta.$$

Finally, the bound can be rewritten as:

$$\mathcal{R}(\mathbf{M}, R) \leq \mathcal{R}_S(\mathbf{M}, R) + \frac{8(\max(1, 4K^2))^2}{n \min(\mu, \lambda)} + \left(\frac{16(\max(1, 4K^2))^2}{\min(\mu, \lambda)} + B \right) \sqrt{\frac{\ln 1/\delta}{2n}}.$$

Thus, the convergence of our bound is, at worst, two times slower.

3.5 Experiments

3.5.1 Algorithms and Datasets

In this section, we aim at evaluating the behavior of ME^2 with respect to some machine learning algorithms described below. Those methods have been selected to characterize some specificities of ME^2.

- Since we established a link between our local ellipsoids and the rules induced by decision trees in the form of local rectangles (Figure 3.4), we compare ME^2 with standard **decision trees** (DT). The objective here is to show that the *learned ellipsoids better capture the local information* of the input space.

- To deal with imbalanced datasets, a commonly used strategy (as seen in Chapter 2) consists in sampling the data to fix the imbalance problem. Therefore, we also learn a decision tree DT_O (resp. DT_U) after a pre-processing step which consists in **over-sampling** (with replacement) the minority class examples (resp. **under-sampling** the majority class examples). We also combine the two previous approaches (DT_{OU}). Finally, we applied a SMOTE-like strategy (Chawla et al., 2002) (DT_{SMOTE}) which creates synthetic minority class examples in the neighborhood of the positive examples. The goal of this comparison is to show how ME^2 behaves even if *it does not resort to sampling processes.*

- When the proportion of positive examples is too small, some Support Vector Data Description methods (SVDD) (Azami et al., 2014; Pauwels and Ambekar, 2011; Tax and Duin, 2004) - like **one-class SVMs** (Pauwels and Ambekar, 2011) - address the anomaly detection problem as an unsupervised outlier detection task. We run here one-class SVMs with two kernels: a linear kernel (LOCSVM) and a RBF kernel (KOCSVM). The objective of this comparison is to check if ME^2 *makes a good use of the few positive labels* compared to unsupervised methods. We also made use of the labels and run standard linear SVMs (LSVM), and RBF kernel-based SVMs (KSVM).

All the classifiers are trained using the corresponding machine learning packages in **R**[5], that is **C50** for the decision trees, **e1071** for the SVMs, **DMwR** for SMOTE and **Rsolnp** for ME^2.

[5]https://www.r-project.org/

Dataset	Nb. of ex.	Nb. of feat.	IR
Yeast3	1 484	8	10.9%
Abalone	4 177	8	10.7%
Wine	1 599	11	3.3%
Abalone 17	2 338	8	2.5%
Yeast6	1 484	8	2.4%
Abalone 20	1 916	8	1.4%
Bank Fraud	15 000	17	1%

Table 3.1: Number of instances, number of features, Imbalance Ratio (i.e. number of positives over the number of instances).

The experiments are performed on 6 datasets coming from the UCI and KEEL databases[6] and a subset of the dataset of the Blitz Company on the bank fraud detection task. Their characteristics (number of examples, features, imbalance ratio IR) are described in Table 3.1. Note that the categorical variables have been replaced by binary features. For example, in the *Abalone* dataset, the attribute $V = \{M, I, F\}$ is changed into three new features M=(1,0,0), I=(0,1,0) and F=(0,0,1).

3.5.2 Experimental Setup

As explained before, the classic *accuracy* is not a suitable criterion to address issues due to the presence of imbalanced data. For this reason, we evaluate the algorithms using the *F-measure* (see Section 2.2).

For each series of experiments, the dataset is separated into a training/validation set S (80% of the total number of examples) and a test set (20%). We use then a 2-fold cross-validation on S while preserving the same IR in each fold to tune the parameters of the different methods. Each experiment is repeated 10 times and the reported results are the average over the 10 trials.

Remember that we learn an ellipsoid centered at each positive example of S. This ellipsoid defines in some way the local region of the projection space which is under the influence of the considered positive example. In this context, at both validation (to tune the parameters) and test time, a query x' is associated to its closest positive example $\mathcal{N}_{\mathbf{x}'}$ (with respect to the Euclidean distance) in the training set. Then, in order to take into account the local density of positives and negatives in the corresponding ellipsoid, and following the idea suggested in (Barandela et al., 2003), we apply the following decision rule, $\mathbf{x}'$ will be predicted as positive if:

[6]These datasets can be found either on the *UCI Repository* (https://archive.ics.uci.edu/ml/datasets.html) or the *KEEL* website in the "Imbalanced data sets for classification" repository (http://sci2s.ugr.es/keel/imbalanced.php?order=ir#sub60).

Algorithm	Yeast3	Abalone	Wine	Abalone17	Yeast6	Abalone20	Bank Fraud	Time
DT	**0.77 ± 0.06**	0.64 ± 0.03	0.00 ± 0.00	0.00 ± 0.00	**0.51 ± 0.23**	0.10 ± 0.15	0.00 ± 0.00	3.2
DT_O	0.75 ± 0.06	0.64 ± 0.04	0.06 ± 0.09	0.35 ± 0.08	0.45 ± 0.12	**0.30 ± 0.15**	0.04 ± 0.03	3.5
DT_U	0.76 ± 0.08	**0.67 ± 0.03**	0.09 ± 0.11	0.28 ± 0.11	0.49 ± 0.11	0.19 ± 0.20	0.04 ± 0.03	2.4
DT_{OU}	0.72 ± 0.04	0.61 ± 0.04	0.15 ± 0.06	0.34 ± 0.05	0.36 ± 0.12	**0.30 ± 0.12**	**0.05 ± 0.04**	2.6
DT_{SMOTE}	0.65 ± 0.10	0.57 ± 0.07	0.14 ± 0.12	0.24 ± 0.09	0.18 ± 0.09	0.27 ± 0.13	0.04 ± 0.03	28.1
LSVM	0.67 ± 0.05	0.62 ± 0.01	0.14 ± 0.08	0.29± 0.01	0.30 ± 0.05	0.23 ± 0.02	0.03 ± 0.02	702.5
RBFSVM	0.66 ± 0.09	0.63 ± 0.03	0.07 ± 0.08	0.17 ± 0.06	0.36 ± 0.09	0.13 ± 0.13	0.00 ± 0.00	39.2
LOCSVM	0.01 ± 0.02	0.11 ± 0.03	0.02 ± 0.07	0.10 ± 0.05	0.00 ± 0.00	0.05 ± 0.13	0.03 ± 0.01	28.3
KOCSVM	0.01 ± 0.02	0.21 ± 0.04	0.01 ± 0.07	0.06 ± 0.03	0.00 ± 0.00	0.05 ± 0.11	0.00 ± 0.00	1472.0
ME^2	0.67 ± 0.06	0.61 ± 0.03	**0.20 ± 0.09**	**0.46 ± 0.07**	0.46 ± 0.08	**0.30 ± 0.07**	**0.05 ± 0.02**	2.9

Table 3.2: Comparison of the methods in terms of F-Measure over 10 runs. The best results are indicated in bold font. Values in parenthesis represent the standard deviation. The last column reports the average running time (in seconds) for one run.

1. It is inside the ellipsoid centered at $\mathcal{N}_{\mathbf{x}'}$. This means that $\mathbf{x}'$ is actually under the influence of $\mathcal{N}_{\mathbf{x}'}$ which occurs when the corresponding learned Mahalanobis distance verifies: $\|\mathbf{x}' - \mathcal{N}_{\mathbf{x}'}\|_{\mathbf{M}_{\mathcal{N}_{\mathbf{x}'}}} \leq R_{\mathcal{N}_{\mathbf{x}'}}$, where $\mathbf{M}_{\mathcal{N}_{\mathbf{x}'}}$ is the PSD matrix learned by ME^2 corresponding to the ellipsoid centered at $\mathcal{N}_{\mathbf{x}'}$ and $R_{\mathcal{N}_{\mathbf{x}'}}$ is its associated radius.

2. Its nearest neighbor in the ellipsoid is a positive example w.r.t the learned local distance $\|\mathbf{x}' - \mathbf{x}\|_{\mathbf{M}_{\mathcal{N}_{\mathbf{x}'}}}$.

Otherwise, $\mathbf{x}'$ is predicted as negative.

Note that the hyper-parameters μ and λ are tuned respectively in the range $\{0.75, 0.8, 0.85, 0.9, 0.95, 1, 2, 10\}$ and $\{10^i\}_{i=-6}^{2}$ by maximizing the F-Measure for each local model according to the previous rule.

3.5.3 Results

The results are reported in Table 3.2. The datasets are sorted from the least to the most imbalance ratio to see the effect of ME^2 with a decreasing rate of positive examples. We can make the following remarks:

- On average, ME^2 gives better results than the other methods even if it does not resort to sampling processes. As shown in Figure 3.5, its average rank over the 7 datasets, **2.6** is better than the others. If we focus on the 5 datasets with large imbalance (*Wine*, *Abalone17*, *Yeast6*, *Abalone20* and the *Bank Fraud*), ME^2 is even better with an average rank of **1.4**.

- For the first two datasets (i.e. *Yeast3* and *Abalone*) where the rate of positive examples is greater than 10%, our method is not very useful. This behavior can be explained in two different ways: (i) when the number of learned ellipsoids grows, their overlapping is larger and larger and therefore the False Positive rate increases; (ii) a large number

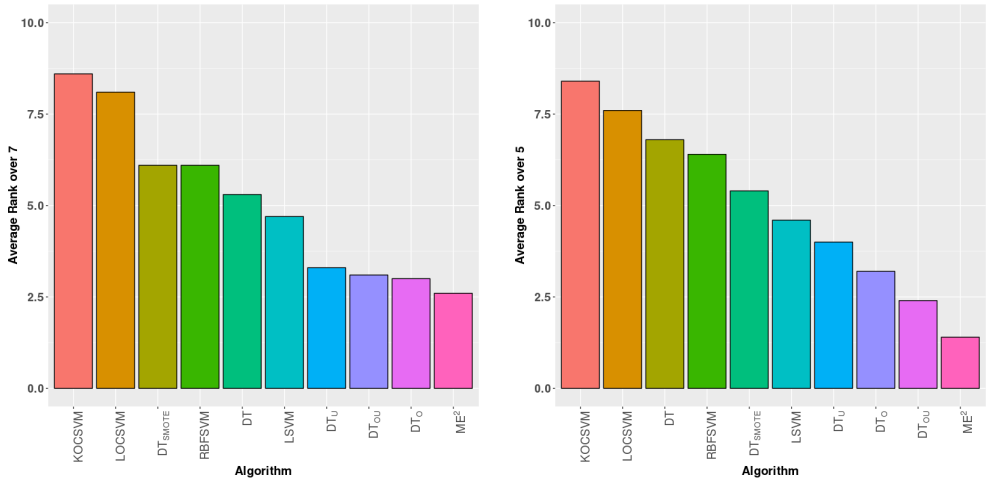

Figure 3.5: Average ranks over all the datasets (on the left) and over the five most imbalanced datasets (on the right).

of ellipsoids induces an increasing risk of generating close ellipsoids with different orientations and shapes. About this second point, an interesting perspective would consist in constraining close positive examples to have similar ellipsoids.

- Compared with decision trees, these experiments show that ME^2 has a much better capacity to capture the local specificities of the feature space than the local rectangles learned by decision trees. For three datasets, it is worth noticing that decision trees even do not capture anything (*Wine*, *Abalone17*, *Abalone20*).

- The results obtained by one-class SVMs are much worse than the other methods. This behavior shows that for all the datasets, including the bank fraud database, the positive examples cannot be considered as outliers - that is - being distant from other observations. Moreover, these results show that the underlying distribution of the minority class is likely to be multimodal.

- ME^2 works better than standard SVMs while the latter use a re-weighting scheme in the objective function to balance the data.

We also report in Table 3.2 the average running time for one run of each method. We can see that since ME^2 learns matrices that directly satisfy the positive semi definiteness, our method is effective, *i.e.* very close to decision trees. Furthermore, ME^2, as the other algorithms, can be easily parallelized. However, note that if one uses ME^2 without parallelizing the learning of the p ellipsoids, the running time will be on average p times the result reported in Table 3.2. But ME^2 will be still efficient (at least better then kernelized-SVMs) since p is supposed to be very small in highly imbalanced scenarios.

3.5.4 On the Impact of the Regularization Parameters μ and λ

As already mentioned before, μ and λ have an impact respectively on the size and on the shape/orientation of the learned ellipsoids. Therefore, if they are properly tuned, those two parameters should allow us to control the rates of False Positives (FP) and False Negatives (FN) which are involved in the *Precision* and *Recall* and thus in the *F-Measure*. To illustrate the capacity of ME^2 to control FP and/or FN, we used the F_β-Measure which is defined as follows:

$$F_\beta = (1+\beta^2).\frac{Precision \times Recall}{(\beta^2 \times Precision) + Recall}$$

and can be rewritten as:

$$F_\beta = \frac{(1+\beta^2) \times TP}{(1+\beta^2) \times TP + \beta^2 \times FN + FP},$$

where TP and TN are respectively the rates of true positives and true negatives. It is known that the larger β (resp. the smaller) the larger the role played by FN (resp. FP) in F_β. We performed the following series of experiments: for different values of β in the range]0,10], we run ME^2 on the bank fraud detection task by tuning for each learned ellipsoid the parameters (μ, λ) with respect to the F_β-Measure. We chose this dataset because of its Imbalance Ratio and almost because its consists of real data for a real application. We report on Figure 3.6, the FP and FN rates associated to each value of β. This figure illustrates that ME^2 is able to find good pairs (λ, μ) maximizing FP while β is growing. Said differently, according to the application at hand, ME^2 will be able to play with λ and μ to favors either the Recall or the Precision.

3.6 Conclusion

In this chapter, we have presented a method to learn *Maximum Excluding Ellipsoids* in the context of imbalanced binary classification tasks. Our algorithm, called ME^2, is simple, because based on local linear models, and theoretically supported by generalization guarantees that have been derived by using the uniform stability framework. We have shown that our method is particularly efficient and robust when the rate of positive examples is very small. The reason comes from the fact that ME^2 is able to learn decision boundaries in the form of ellipsoids (via a metric learning-based strategy) that are optimized locally to best fit the specificities of the space.

ME^2 is based on a very simple decision rule looking for the nearest ellipsoid to a test query. We think that this rule may benefit from further investigation, e.g. by considering a combination of ellipsoids to predict the label of a test data. This would be possible by considering a graph over the ellipsoids centers where information would be shared like in an information network. Besides, from a theoretical point of view, we have derived a guarantee on the learned matrix $\mathbf{M}$ and radius R. Since our decision rule is close to a nearest neighbor classifier decision rule, it would be interesting to establish a link between the quality of $\mathbf{M}$

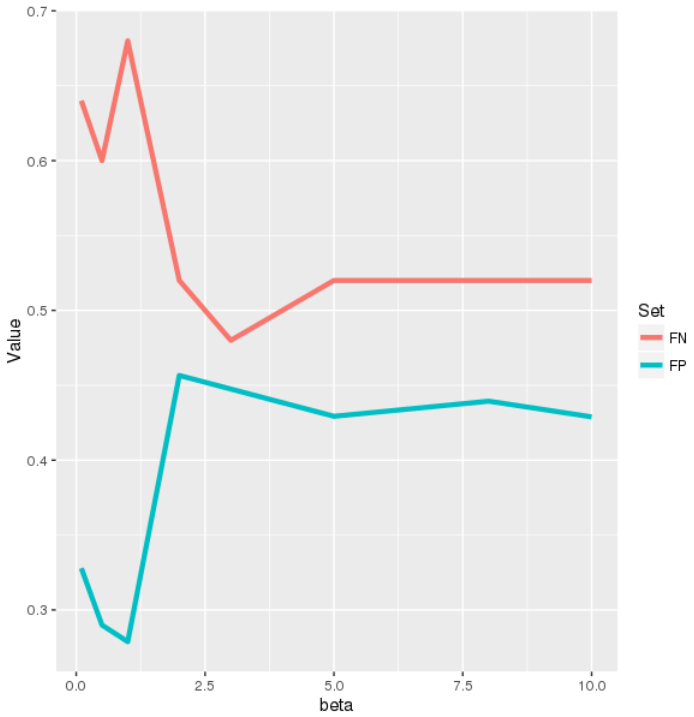

Figure 3.6: False positive and false negative rates according to an increasing value of β in the F_β-measure.

and R and the generalization error of such a classifier. Another perspective would be to partition the positive example space and constrain the ellipsoids to be similar at least in terms of orientation (using some regularization) if they have been learned from the same cluster. We have also compared the decision areas between ME^2 and a decision tree algorithm, a possible extension can be a *Random Forest of* ME^2. The idea is to build several ellipsoids around a given positive examples using a subset of the training examples and a subset of the features. Finally, in a context of fraud detection where the fraud strategy tends to evolve through time, developing an online version of our algorithm might be relevant to better capture distribution shifts.

Note that the decision rule used in this chapter is based on the Nearest-Neighbor algorithm and finding frauds highly depends on the distance to the closest center of an ellipsoid, thus the center of a known fraud. In the next chapter, we focus on the k-NN algorithm and show how we can adjust the distance function to better learn from imbalanced data.

Chapter 4

A Corrected Nearest Neighbor Algorithm Maximizing the F-Measure from Imbalanced Data

This chapter is based on the following submission

Rémi Viola, Rémi Emonet, Amaury Habrard, Guillaume Metzler, Sébastien Riou, and Marc Sebban. An adjusted nearest neighbor algorithm maximizing the f-measure from imbalanced data. In *In Proceedings of the 31st International Conference on Tools with Artificial Intelligence (ICTAI-2019)*, 2019a

Rémi Viola, Rémi Emonet, Amaury Habrard, Guillaume Metzler, Sébastien Riou, and Marc Sebban. Une version corrigée de l'algorithme des plus proches voisins pour l'optimisation de la f-mesure dans un contexte déséquilibré. In *Conférence francophone sur l'Apprentissage Automatique (CAp-19)*, 2019b

---

Abstract

In the previous chapter, we have seen that the distance to positive examples is a key information to take into account in imbalanced scenarios. The previous work built excluding areas over which a new query has no chance to be a fraud. The decision rule was based on Nearest-Neighbor algorithm applied between a new query and the (positive) centers of the learned ellipsoids On the other hand, when a NN classifier is directly applied on highly imbalanced dataset, composed of a few positives and a huge number of negatives, the probability for the nearest neighbor to belong to the majority class is much larger than that of the minority class. Indeed, small Voronoi regions for the positives (as we have seen in Chapter 1) drastically reduce the chance of being predicted positive. Based on this observation and a simple geometrical idea, we introduce an algorithm that weights the distance between a query sample and any positive training example. This leads to a modification of the Voronoi regions and thus of the decision boundaries of the NN algorithm. We provide a theoretical justification about the weighting scheme needed

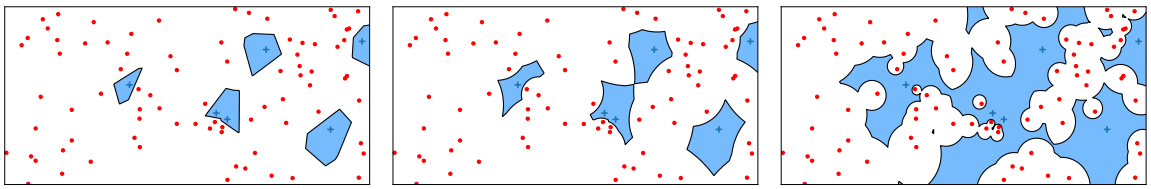

Figure 4.1: Impact on the Voronoi regions according to the weighted distance to the positive instances. On the left, the Voronoi regions around the positives are small. The risk to generate false negatives (FN) at test time is large. On the right: by increasing too much the regions of influence of the positives, the probability to get false positives (FP) grows. In the middle: an appropriate trade-off between the two previous situations.

to reduce the False Negative rate while controlling the number of False Positives. We perform an extensive experimental study on many public imbalanced datasets. The results presented in this chapter show that our method is very effective and, interestingly, yields the best performance when combined with state of the art sampling methods.

4.1 Introduction

One peculiarity of imbalanced datasets can be interpreted from a geometric perspective. As illustrated in Figure 4.1 (left) which shows the Voronoi cells on an artificial imbalanced dataset (where two adjacent cells have been merged if they concern examples of the same class), the regions of influence of the positive examples are much smaller than that of the negatives. This explains why at test time, in imbalanced learning, the risk to get a false negative is high, leading to a low F-measure. Note that increasing the regions of influence of the positives would allow us to reduce FN and improve the F-measure. However, not controlling the expansion of these regions may have a dramatic impact on FP, and so on the F-Measure, as illustrated in Figure 4.1 (right).

The main contribution of this chapter is about the problem of finding the appropriate trade-off (Figure 4.1 (middle)) between the two above-mentioned extreme situations (large FP or FN) both leading to a low F-Measure. A natural way to increase the influence of positives may consist in using generative models (like GANs (Goodfellow et al., 2014)) to sample new artificial examples, mimicking the negative training samples. However, beyond the issues related to the parameter tuning, the computation burden and the complexity of such a method, using GANs to optimize the precision and recall is still an open problem (see (Sajjadi et al., 2018) for a recent paper on this topic).

An other strategy, close to the use of GAN, consists in generating positive examples at the decision boundary as depicted on the Figure 4.2 to find a good trade-off between FP and FN. On this figure, the grey circles represent the area where a new query is more likely to be negative, also called negative area. It is represented by a sphere of radius equal to the distance to its nearest-neighbor. With the generation of points, we will be able to drastically

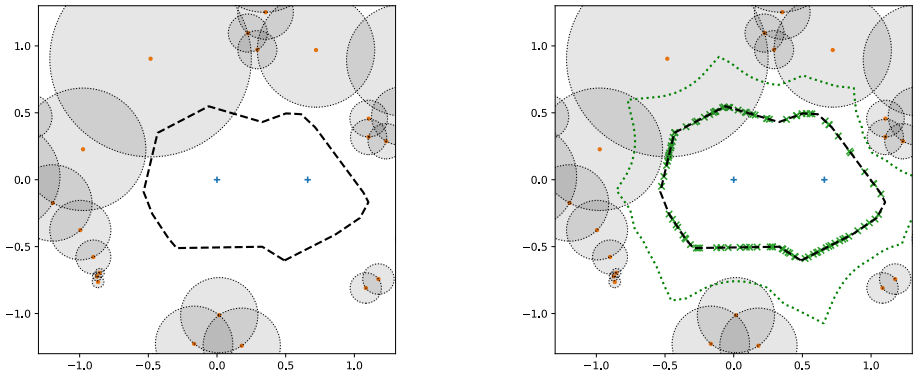

Figure 4.2: Decision boundary without (left) and with (right) the generation of positive points. The grey circles represent the region in which a new query is considered as negative. The green crosses represent the generated positives at the frontier and the dotted green line corresponds the new decision boundary.

increase the positive influence (see Figure 4.2 right). However, this can lead to a new decision boundary which can lie in the negative area and, thus, to a highest ratio of false positives.

That is why, we show in this chapter that a much simpler strategy can be used by modifying the distance exploited in a k-nearest neighbor (NN) algorithm (Cover and Hart, 1967) which enjoys many interesting advantages, including its simplicity, its capacity to approximate asymptotically any locally regular density, and its theoretical rootedness (Luxburg and Bousquet, 2004; Kontorovich and Weiss, 2015; Kontorovich et al., 2016). k-NN also benefited from many algorithmic advances during the past decade in the field of metric learning, aiming at optimizing under constraints the parameters of a metric, typically the Mahalanobis distance, as done in LMNN (Weinberger and Saul, 2009) or ITML (Davis et al., 2007) (see (Bellet et al., 2015) for a survey). Unfortunately, existing metric learning methods are dedicated to enhance the k-NN accuracy and do not focus on the optimization of criteria, like the F-measure, in scenarios where the positive training examples are scarce. A geometric solution to increase, at a very low cost, the region of influence of the minority class consists in modifying the distance when comparing a query example to a positive training sample. More formally, we show in this chapter that the optimization of the F-Measure is facilitated by weighting the distance to any positive by a coefficient $\gamma \in [0, 1]$ leading to the expansion of the Voronoi cells around the minority examples. An illustration is given in Figure 4.1 (middle) which might be seen as a good compromise that results in the reduction of FN while controlling the risk to increase FP. Note that our strategy boils down to modifying the local density of the positive examples. For this reason, we claim that it can be efficiently combined with SMOTE-based sampling methods whose goal is complementary and consists, as already presented, in generating examples on the path linking two (potentially far) positive neighbors. Our experiments will confirm this intuition.

4.2 Related Work

In this section, we present the main strategies that have been proposed in the literature to address the problem of learning from imbalanced datasets with the k-nn classifier. Some of them have already been cited in Chapter 2. We enter a bit more into details in this section and show where the notions of and closeness are involved in these approaches.

We consider a training sample $S = \{(\mathbf{x}_i, y_i), i = 1, ..., m\}$ of size m, drawn from an unknown joint distribution $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} = \mathbb{R}^p$ is the feature space and $\mathcal{Y} = \{-1, 1\}$ is the set of labels. Let us assume that $S = S_+ \cup S_-$ with m_+ positives $\in S_+$ and m_- negatives $\in S_-$ where $m = m_+ + m_-$.

4.2.1 Distance-based Methods

Several strategies have been devised to improve k-NN. The oldest method is certainly the one presented in (Dudani, 1976) which consists in associating to each neighbor a voting weight that is inversely proportional to its distance to a query point $\mathbf{x}$. The assigned label $\hat{y}$ of $\mathbf{x}$ is defined as:

$$\hat{y} = \sum_{\mathbf{x}_i \in \text{kNN}(\mathbf{x})} y_i \times \frac{1}{d(\mathbf{x}, \mathbf{x}_i)},$$

where kNN($\mathbf{x}$) stands for the set of the k nearest neighbors of $\mathbf{x}$. An other method consists in learning the weight associated to an example (Hajizadeh et al., 2014).

In (Barandela et al., 2003), the authors account both the label and the distance to the neighbors $(\mathbf{x}_i, y_i)$ to define a weighted metric d' from the euclidean distance d, as follows:

$$d'(\mathbf{x}, \mathbf{x}_i) = \left(\frac{m_i}{m}\right)^{1/p} d(\mathbf{x}, \mathbf{x}_i),$$

where m_i is the number of examples in the class y_i. As we will see later, this method falls in the same family of strategies as our contribution, aiming at weighting the distance to the examples according to their label. However, three main differences justify why our method will be better in the experiments: (i) d' is fixed in advance while we will adapt the weight that optimizes the F- measure; (ii) because of (i), d' needs to take into account the dimension p of the feature space (and so will tend to d as p grows) while this will be intrinsically captured in our method by optimizing the weight given the p-dimensional space; (iii) d' is useless when combined with sampling strategies (indeed, $\frac{m_i}{m}$ would tend to be uniform) while our method will allow us to weight differently the original positive examples and the ones artificially generated.

Another way to assign weights to each class, which is close to the sampling methods, is to duplicate the positive examples according to the Imbalance Ratio: m_-/m_+. Thus, it can be seen as a *uniform* over-sampling technique, where all positives are replicated the same number of times. However, note that this method requires to work with $k > 1$.

A last family of methods that try to improve k-NN is related to *metric learning*. LMNN (Weinberger and Saul, 2009) or ITML (Davis et al., 2007) are two famous examples which optimize

under constraints a Mahalanobis distance $d_{\mathbf{M}}(\mathbf{x}, \mathbf{x}_i) = \sqrt{(\mathbf{x} - \mathbf{x}_i)^\top \mathbf{M}(\mathbf{x} - \mathbf{x}_i)}$ parameterized by a positive semidefinite (PSD) matrix $\mathbf{M}$. Such methods seek a linear projection of the data in a latent space where the Euclidean distance is applied. As we will see in the following, our weighting method is a specific case of metric learning which looks for a diagonal matrix - applied only when comparing a query to a positive example - and that behaves well in terms of F-Measure.

4.2.2 Sampling Strategies

As already mentioned in the previous chapters, one way to overcome the issues induced by the lack of positive examples is to compensate artificially the imbalance between the two classes. Sampling strategies (Fernández et al., 2018) have been proven to be very efficient to address this problem. It turns out that, in most of them, the k-NN rule is involved.

The Synthetic Minority Over-sampling Technique (Chawla et al., 2002) (SMOTE) over-samples a dataset by creating new synthetic positive data. For each minority example $\mathbf{x}$, it randomly selects one of its k nearest positive neighbors and then creates a new random positive point on the line between this neighbor and $\mathbf{x}$. This is done until some desired ratio is reached.

Borderline-SMOTE (Han et al., 2005) is an improvement of the SMOTE algorithm. While the latter generates synthetic points from all positive points, BorderLine-SMOTE only focuses on those having more negatives than positives in their neighborhood. More precisely, new points are generated if the number n of negatives in the k-neighborhood is such that $k/2 \leq n \leq k$.

The Adaptive Synthetic (He et al., 2008) (ADASYN) sampling approach is also inspired from SMOTE. By using a weighted distribution, it gives more importance to classes that are more difficult to classify, *i.e.* where positives are surrounded by many negatives, and thus generates more synthetic data for these classes.

Two other strategies combine an over-sampling step with an under-sampling procedure. The first one uses the Edited Nearest Neighbors (Wilson, 1972) (ENN) algorithm on the top of SMOTE. After SMOTE has generated data, the ENN algorithm removes data that are miss-classified by their k nearest neighbors. The second one combines SMOTE with Tomek link (Tomek, 1976). A Tomek link is a pair of points $(\mathbf{x}_i, \mathbf{x}_j)$ from different classes for which there is no other point $\mathbf{x}_k$ verifying $d(\mathbf{x}_i, \mathbf{x}_k) \leq d(\mathbf{x}_i, \mathbf{x}_j)$ or $d(\mathbf{x}_k, \mathbf{x}_j) \leq d(\mathbf{x}_i, \mathbf{x}_j)$. In other words, $\mathbf{x}_i$ is the nearest neighbor of $\mathbf{x}_j$ and vice-versa. If so, one removes the example of $(\mathbf{x}_i, \mathbf{x}_j)$ that belongs to the majority class. Note both strategies tend to eliminate the over-lapping between classes.

Interestingly, we can note that all the previous sampling methods try to overcome the problem of learning from imbalanced data by resorting to the notion of k-neighborhood. This is justified by the fact that k-NN has been shown to be a good estimate of the density at a given point in the feature space. In our contribution, we stay in this line of research.

Rather than generating new examples, that would have a negative impact from a complexity perspective, we locally modify the density around the positive points. This is achieved by re-scaling the distance between a test sample and the positive training examples. We will show that such a strategy can be efficiently combined with sampling methods, whose goal is complementary, by potentially generating new examples in regions of the space where the minority class is not present.

4.3 Our Proposed Strategy

In this section, we present our $\gamma k-$NN method which works by scaling the distance between a query point and positive training examples by a factor.

4.3.1 A Corrected $k-$NN algorithm

Statistically, when learning from imbalanced data, a new query $\mathbf{x}$ has more chance to be close to a negative example due to the rarity of positives in the training set, even around the mode of the positive distribution. We have seen two families of approaches that can be used to counteract this effect: (i) creating new synthetic positive examples, and (ii) changing the distance according to the class. The approach we propose falls into the second category.

We propose to modify how the distance to the positive examples is computed, in order to compensate for the imbalance in the dataset. We artificially bring a new query $\mathbf{x}$ closer to any positive data point $\mathbf{x}_i \in S_+$ in order to increase the effective area of influence of positive examples. The new measure d_γ that we propose is defined, using an underlying distance d (e.g. the euclidean distance) as follows:

$$d_\gamma(\mathbf{x}, \mathbf{x}_i) = \begin{cases} d(\mathbf{x}, \mathbf{x}_i) & \text{if } \mathbf{x}_i \in S_-, \\ \gamma \cdot d(\mathbf{x}, \mathbf{x}_i) & \text{if } \mathbf{x}_i \in S_+. \end{cases}$$

As we will tune the γ parameter, this new way to compute the similarity to a positive example is close to a Mahalanobis-distance learning algorithm, looking for a PSD matrix, as previously described. However, the matrix $\mathbf{M}$ is restricted to be $\gamma^2 \cdot \mathbf{I}$, where $\mathbf{I}$ refers to the identity matrix. Moreover, while metric learning typically works by optimizing a convex loss function under constraints, our γ is simply tuned such as maximizing the non convex F-Measure. Lastly, and most importantly, it is applied only when comparing the query to positive examples. As such, d_γ is not a proper distance, however, it is exactly this which allows it to compensate for the class imbalance. In the binary setting, there is no need to have a γ parameter for the negative class, since only the relative distances are used. In the multi-class setting with K classes, we would have to tune up to $K-1$ values of γ.

Before formalizing the $\gamma k-$NN algorithm that will leverage the distance d_γ, we illustrate in Figure 4.3, on 2D data, the decision boundary induced by a nearest neighbor binary classifier that uses d_γ. We consider an elementary dataset with only two points, one positive and one negative. The case of $\gamma = 1$, which is a traditional 1-NN is shown in a thick black line.

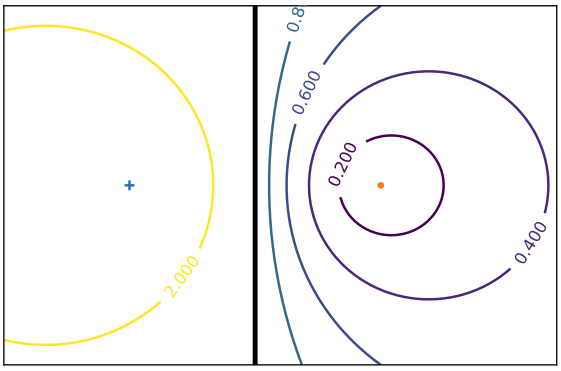

Figure 4.3: Evolution of the decision boundary based on d_γ, for a 1-NN classifier, on a 2D dataset with one positive (resp. negative) instance represented by a blue cross (resp. orange point). The value of γ is given on each boundary ($\gamma = 1$ on the thick line).

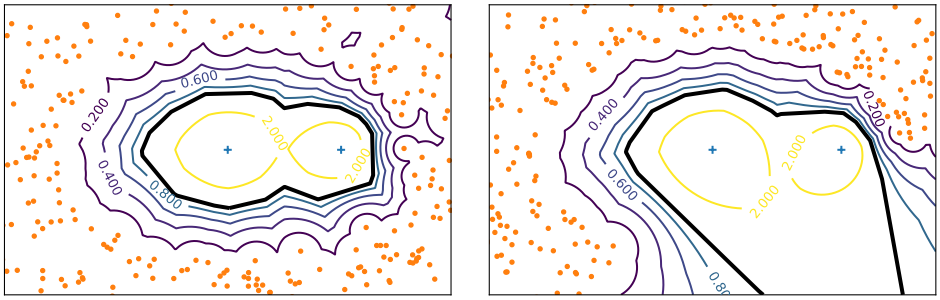

Figure 4.4: Behavior of the decision boundary according to the γ value for the 1-NN classifier on two toy datasets. The positive points are represented by blue crosses and the negatives by orange points. The black line represents the standard decision boundary for the 1-NN classifier, i.e. when $\gamma = 1$.

Lowering the value of γ below 1 brings the decision boundary closer to the negative point, and eventually tends to surround it very closely. In Fig 4.4, two more complex datasets are shown, each with two positive points and several negative examples. As intuited, we see that the γ parameter allows to control how much we want to push the boundary towards negative examples.

We can now introduce the $\gamma k-$NN algorithm (see Algo 3) that is parameterized by a γ parameter. It has the same overall complexity as $k-$NN. The first step to classify a query $\mathbf{x}$ is to find its k nearest negative neighbors and its k nearest positive neighbors. Then, the distances to the positive neighbors are multiplied by γ, to obtain d_γ. These $2k$ neighbors are then ranked and the k closest ones are used for classification (with a majority vote, as in $k-$NN). It should be noted that, although d_γ does not define a proper distance, we can still use any existing fast nearest neighbor search algorithm, because the actual search is done (twice but) only using the original distance d.

Algorithm 3: Classification of a new example with $\gamma k-$NN

Input : a query $\mathbf{x}$ to be classified, a set of labeled samples $S = S_+ \cup S_-$, a number of neighbors k, a positive real value γ, a distance function d

Output: the predicted label of $\mathbf{x}$

$\mathcal{NN}^-, \mathcal{D}^- \leftarrow nn(k, \mathbf{x}, S_-)$ // nearest negative neighbors with their distances

$\mathcal{NN}^+, \mathcal{D}^+ \leftarrow nn(k, \mathbf{x}, S_+)$ // nearest positive neighbors with their distances

$\mathcal{D}^+ \leftarrow \gamma \cdot \mathcal{D}^+$

$\mathcal{NN}_\gamma \leftarrow firstK\left(k, sortedMerge((\mathcal{NN}^-, \mathcal{D}^-), (\mathcal{NN}^+, \mathcal{D}^+))\right)$

$y \leftarrow +$ if $\left|\mathcal{NN}_\gamma \cap \mathcal{NN}^+\right| \geq \frac{k}{2}$ else $-$ // majority vote based on $\mathcal{NN}_\gamma$

return y

4.3.2 Theoretical analysis

In this section, we formally analyze what could be a good range of values for the γ parameter of our corrected version of the $k-$NN algorithm. To this aim, we study what impact γ has on the probability to get a false positive (and false negative) at test time and explain why it is important to choose $\gamma < 1$ when the imbalance in the data is significant. The following analysis is made for $k = 1$ but note that the conclusion still holds for a k-NN.

Proposition 4.1. *(False Negative probability) Let $d_\gamma(\mathbf{x}, \mathbf{x}_+) = \gamma d(\mathbf{x}, \mathbf{x}_+)$, $\forall \gamma > 0$, be our modified distance used between a query $\mathbf{x}$ and any positive training example $\mathbf{x}_+$, where $d(\mathbf{x}, \mathbf{x}_+)$ is some distance function. Let $FN_\gamma(\mathbf{z})$ be the probability for a positive example $\mathbf{z}$ to be a false negative using Algorithm (3). The following result holds: if $\gamma \leq 1$,*

$$FN_\gamma(\mathbf{z}) \quad \leq \quad FN(\mathbf{z})$$

Proof. (sketch of proof) Let ϵ be the distance from $\mathbf{z}$ to its nearest-neighbor $N_\mathbf{z}$. $\mathbf{z}$ is a false negative if $N_\mathbf{z} \in S_-$ that is all positives $\mathbf{x}' \in S_+$ are outside the sphere $\mathcal{S}_{\frac{\epsilon}{\gamma}}(\mathbf{z})$ centered at $\mathbf{z}$ of radius $\frac{\epsilon}{\gamma}$. Therefore,

$$FN_\gamma(\mathbf{z}) = \prod_{\mathbf{x}' \in S_+} \left(1 - P(\mathbf{x}' \in \mathcal{S}_{\frac{\epsilon}{\gamma}}(\mathbf{z}))\right) = \left(1 - P(\mathbf{x}' \in \mathcal{S}_{\frac{\epsilon}{\gamma}}(\mathbf{z}))\right)^{m_+} \tag{4.1}$$

while

$$FN(\mathbf{z}) = \left(1 - P(\mathbf{x}' \in \mathcal{S}_\epsilon(\mathbf{z}))\right)^{m_+}. \tag{4.2}$$

Solving (4.1) $\leq$ (4.2) implies $\gamma \leq 1$. □

This result means that satisfying $\gamma < 1$ allows us to increase the decision boundary around positive examples (as illustrated in Figure 4.4), yielding a smaller risk to get false negatives at test time. An interesting comment can be made from Eq.(4.1) and (4.2) about their convergence. As m_+ is supposed to be very small in imbalanced datasets, the convergence of $FN(\mathbf{z})$ towards 0 is pretty slow, while one can speed-up this convergence with $FN_\gamma(\mathbf{z})$ by increasing the radius of the sphere $\mathcal{S}_{\frac{\epsilon}{\gamma}}(\mathbf{z})$, that is taking a small value for γ.

Proposition 4.2. *(False Positive probability) Let $FP_\gamma(\mathbf{z})$ be the probability for a negative example $\mathbf{z}$ to be a false positive using Algorithm (3). The following result holds: if $\gamma \geq 1$,*

$$FP_\gamma(\mathbf{z}) \quad \leq \quad FP(\mathbf{z}).$$

Proof. (sketch of proof) Using the same idea as before, we get:

$$FP_\gamma(\mathbf{z}) = \prod_{\mathbf{x}' \in S_-} \big(1 - P(\mathbf{x}' \in \mathcal{S}_{\gamma\epsilon}(\mathbf{z}))\big) = \big(1 - P(\mathbf{x}' \in \mathcal{S}_{\gamma\epsilon}(\mathbf{z}))\big)^{m_-} \tag{4.3}$$

while

$$FP(\mathbf{z}) = \big(1 - P(\mathbf{x}' \in \mathcal{S}_{\epsilon}(\mathbf{z}))\big)^{m_-} . \tag{4.4}$$

Solving (4.3) $\leq$ (4.4) implies $\gamma \geq 1$. □

As expected, this result suggests to take $\gamma > 1$ to increase the distance $d_\gamma(\mathbf{z}, \mathbf{x}_+)$ from a negative test sample $\mathbf{z}$ to any positive training example $\mathbf{x}_+$ and thus reduce the risk to get a false positive. It is worth noticing that while the two conclusions from Propositions 4.1 and 4.2 are contradictory, the convergence of $FP_\gamma(\mathbf{z})$ towards 0 is much faster than that of $FN_\gamma(\mathbf{z})$ because $m_- >> m_+$ in an imbalanced scenario. Therefore, fulfilling the requirement $\gamma > 1$ is much less important than satisfying $\gamma < 1$. For this reason, we will impose our Algorithm (3) to take $\gamma \in]0, 1[$. As we will see in the experimental section, the more imbalanced the datasets, the smaller the optimal γ, confirming the previous conclusion.

4.4 Experiments

In this section, we present an experimental evaluation of our method on public and real private datasets with comparisons to classic distance-based methods and state of the art sampling strategies able to deal with imbalanced data. All results are reported using $k = 1$ and 3.

4.4.1 Experimental setup

For the experiments, we use several datasets from the classic UCI [1] and KEEL [2] repositories. The main properties of the datasets are summarized in Table 4.1, including the imbalance ratio (IR).

All the datasets are normalized using a min-max normalization such that each feature lies in the range $[0, 1]$. We randomly draw 80%-20% splits of the data to generate the training and test sets respectively. Hyperparameters are tuned with a 10-fold cross-validation over the training set. We repeat the process over 5 runs and average the results in terms of F-measure F_1. In a first series of experiments, we compare our method, named $\gamma k-$NN, to 6 other distance-based baselines:

[1] `https://archive.ics.uci.edu/ml/datasets.html`
[2] `https://sci2s.ugr.es/keel/datasets.php`

DATASETS	SIZE	DIM	%+	%−	IR
BALANCE	625	4	46.1	53.9	1.2
AUTOMPG	392	7	37.5	62.5	1.7
IONO	351	34	35.9	64.1	1.8
PIMA	768	8	34.9	65.1	1.9
WINE	178	13	33.1	66.9	2
GLASS	214	9	32.7	67.3	2.1
GERMAN	1000	23	30	70	2.3
VEHICLE	846	18	23.5	76.5	3.3
HAYES	132	4	22.7	77.3	3.4
SEGMENTATION	2310	19	14.3	85.7	6
ABALONE8	4177	10	13.6	86.4	6.4
YEAST3	1484	8	11	89	8.1
PAGEBLOCKS	5473	10	10.2	89.8	8.8 1
SATIMAGE	6435	36	9.7	90.3	9.3
LIBRAS	360	90	6.7	93.3	14
WINE4	1599	11	3.3	96.7	29.2
YEAST6	1484	8	2.4	97.6	41.4
ABALONE17	4177	10	1.4	98.6	71.0
ABALONE20	4177	10	0.6	99.4	159.7

Table 4.1: Information about the studied datasets sorted by imbalance ratio.

- the classic $k-$Nearest Neighbor algorithm ($k-$NN),
- the weighted version of $k-$NN using the inverse distance as a weight to predict the label (w$k-$NN) (Dudani, 1976),
- the class weighted version of $k-$NN (cw$k-$NN) (Barandela et al., 2003),
- the $k-$NN version where each positive is duplicated according to the IR of the dataset (dup$k-$NN),
- the metric learning method LMNN (Weinberger and Saul, 2009).

We set the number of nearest neighbors to $k = 3$ for all methods. The hyperparameter μ of *LMNN*, weighting the impact of impostor constraints (see (Weinberger and Saul, 2009) for more details), is tuned in the range $[0, 1]$ using a step of 0.05. Our γ parameter is tuned in the range $[0, 1]$[3] using a step of 0.1.

[3]We experimentally noticed that using a larger range for γ leads in fact to a potential decrease of performances due to overfitting phenomena. This behavior is actually in line with the analysis provided in Section 4.3.2.

DATASETS	3−NN	DUPk−NN	Wk−NN	CWk−NN	LMNN	γk−NN
BALANCE	0.954 (0.017)	0.954 (0.017)	0.957 (0.017)	0.961 (0.016)	**0.965** (0.009)	0.952 (0.032)
AUTOMPG	0.808 (0.077)	0.826 (0.033)	0.810 (0.076)	0.815 (0.060)	0.827 (0.054)	**0.835** (0.023)
IONO	0.752 (0.053)	0.859 (0.021)	0.756 (0.060)	0.783 (0.033)	0.890 (0.039)	**0.924** (0.016)
PIMA	0.500 (0.056)	0.539 (0.033)	0.479 (0.044)	0.514 (0.060)	0.499 (0.070)	**0.562** (0.028)
WINE	0.881 (0.072)	0.852 (0.057)	0.881 (0.072)	0.862 (0.099)	**0.950** (0.036)	0.861 (0.079)
GLASS	0.727 (0.049)	0.733 (0.061)	0.736 (0.052)	0.730 (0.041)	0.710 (0.062)	**0.761** (0.051)
GERMAN	0.330 (0.030)	0.449 (0.037)	0.326 (0.030)	0.347 (0.043)	0.323 (0.054)	**0.465** (0.025)
VEHICLE	0.891 (0.044)	0.867 (0.027)	0.891 (0.044)	0.873 (0.026)	**0.958** (0.020)	0.874 (0.043)
HAYES	0.036 (0.081)	0.183 (0.130)	0.050 (0.112)	0.239 (0.068)	0.036 (0.081)	**0.541** (0.092)
SEGMENTATION	0.859 (0.028)	0.862 (0.018)	0.877 (0.028)	0.844 (0.022)	**0.888** (0.035)	0.845 (0.025)
ABALONE8	0.243 (0.028)	0.318 (0.018)	0.241 (0.028)	0.328 (0.022)	0.247 (0.066)	**0.348** (0.031)
YEAST3	0.634 (0.066)	0.670 (0.034)	0.634 (0.066)	**0.690** (0.021)	0.667 (0.055)	0.686 (0.042)
PAGEBLOCKS	0.842 (0.020)	0.850 (0.024)	0.849 (0.019)	0.847 (0.030)	**0.855** (0.036)	0.844 (0.023)
SATIMAGE	0.454 (0.039)	0.457 (0.027)	0.454 (0.039)	0.458 (0.027)	**0.487** (0.026)	0.433 (0.018)
LIBRAS	**0.722** (0.228)	0.704 (0.279)	**0.722** (0.228)	0.706 (0.229)	0.677 (0.220)	0.684 (0.223)
WINE4	0.031 (0.069)	0.090 (0.086)	0.031 (0.069)	0.000 (0.000)	0.000 (0.000)	**0.093** (0.036)
YEAST6	0.503 (0.302)	0.449 (0.112)	0.502 (0.297)	0.342 (0.073)	0.505 (0.231)	**0.574** (0.172)
ABALONE17	0.057 (0.078)	**0.172** (0.086)	0.057 (0.078)	0.077 (0.032)	0.000 (0.000)	0.084 (0.017)
ABALONE20	0.000 (0.000)	0.000 (0.000)	0.000 (0.000)	**0.140** (0.123)	0.000 (0.000)	0.094 (0.046)
MEAN	0.538 (0.070)	0.570 (0.058)	0.540 (0.070)	0.556 (0.054)	0.552 (0.058)	**0.603** (0.054)

Table 4.2: Results for 3−NN on the datasets. The values correspond to the mean F-measure F_1 over 5 runs. The standard deviation is indicated between brackets. The best result on each dataset is indicated in bold.

In a second series of experiments, we compare our method to the five oversampling strategies described in Section 4.2.2: SMOTE (Chawla et al., 2002), Borderline-SMOTE (Han et al., 2005), ADASYN (He et al., 2008), SMOTE with ENN (Wilson, 1972), SMOTE with Tomek links (Tomek, 1976). The number of generated positive examples is tuned over the set of ratios $\frac{m_+}{m_-} \in \{0.1, 0.2, ..., 0.9, 1.0\}$ and such that the new ratio is greater than the original one before sampling. Other parameters of these methods are the default ones used by the package *ImbalancedLearn* (Lemaître et al., 2017) of *Scikit-learn*.

4.4.2 Results

In this section, we provide the results using k−NN algorithm with $k = 1$ and $k = 3$. The results with $k = 1$ are used to illustrate the study conducted in Section 4.3.2. However, we will see that the value of k does not change the results and leads to the same conclusion.

The results using distance-based methods are provided in Table 4.2. Overall, our γk−NN approach performs much better than its competitors by achieving an improvement of at least 3 points on average, compared to the 2nd best method (DUPk−NN). The different k−NN versions fail globally to provide models efficient whatever the imbalance ratio. The metric learning approach LMNN is competitive when IR is smaller than 10 (although algorithmically

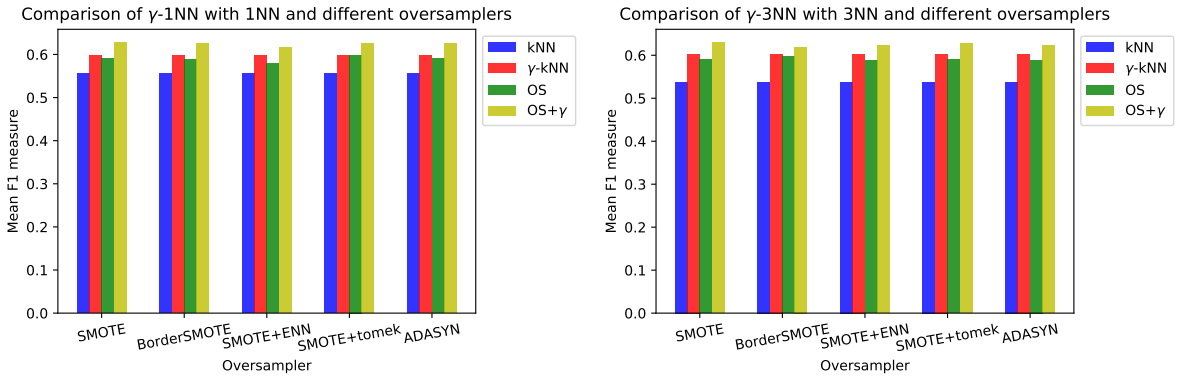

Figure 4.5: Comparison of different sampling strategies averaged over the 19 public datasets. OS refers to the results of the corresponding sampling strategy and $OS + \gamma$ to the case when the sampling strategy is combined with γk-NN. $k-$NN and $\gamma k-$NN refers to the results of these methods without oversampling as obtained in Table 4.2 for $k = 3$ on the right and for $k = 1$ on the left. (numerical values for these graphs are provided in supplementary material)

more costly). Beyond, it faces some difficulties to find a relevant projection space due to the lack of positive data. The efficiency of $\gamma k-$NN is not particularly sensitive to the imbalance ratio.

The results for our second series of experiments, focusing on sampling strategies, are reported on Figure 4.5. We compare each of the 5 sampling methods with the average performances of $3-$NN and γk-NN obtained over the 19 public datasets reported in Table 4.2. Additionally, we also use $\gamma k-$NN on the top of the sampling methods to evaluate how both strategies are complementary. However, in this scenario, we propose to learn a different γ value to be used with the synthetic positives. Indeed, some of them may be generated in some true negative areas and in this situation it might be more appropriate to decrease the influence of such synthetic examples. The γ parameter for these examples is then tuned in the range $[0, 2]$ using a step of 0.1. If one can easily observe that all the oversampling strategies improve the classic $k - NN$, none of them is better than our γk-NN method showing that our approach is able to deal efficiently with imbalanced data. Moreover, we are able to improve the efficiency of γk-NN when it is coupled with an oversampling strategy. The choice of the oversampler does not really influence the results. The gains obtained by using a sampling method with γk-NN for each dataset is illustrated in Figures 4.7 and 4.8 (top).

To study the influence of using two γ parameters when combined with an oversampling strategy, we show an illustration (Figure 4.6 (left)) of the evolution of the F-measure with respect to the γ values for synthetic and real positive instances. The best F-measure is achieved when the γ on real positives is smaller than 1 and when the γ on synthetic positives is greater than 1, justifying the interest of using two parameterizations of γ. In Figure 4.6 (right), we show how having two γ values gives the flexibility to independently control the increased influence of real positives and the one of artificial positives.

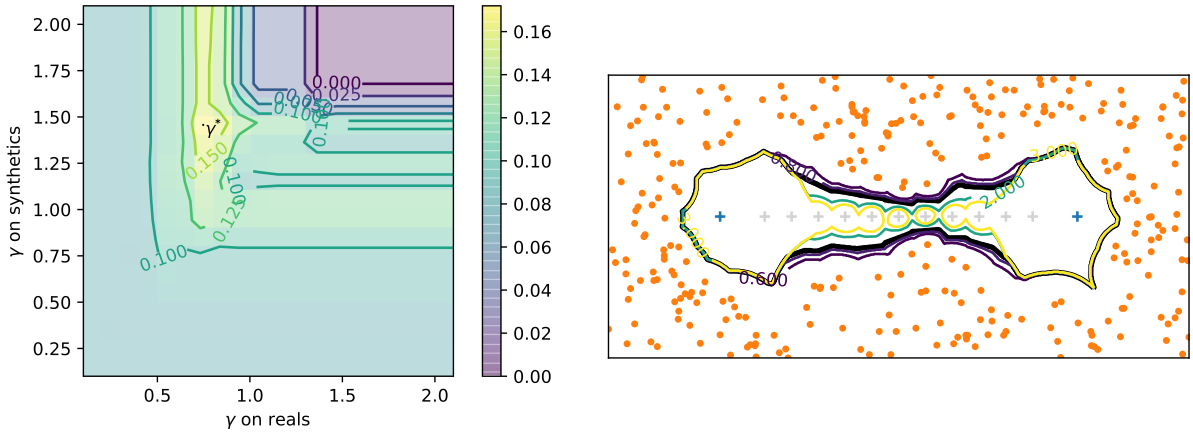

Figure 4.6: (Left) An example of heatmap that shows the best couple of γ for the OS+$\gamma k-$NN strategy on the yeast6 dataset with SMOTE and Tomek links. (Right) Illustration, on a toy dataset, of the effect of varying the γ for generated positive points (in grey) while keeping a fixed $\gamma = 0.4$ for real positive points.

We now propose a study on the influence of the imbalance ratio on the optimal γ-parameter. We consider the *Balance* dataset which has the smallest imbalance ratio that we increase by iteratively randomly under-sampling the minority class over the training set. We report the results on Figures 4.7 and 4.8 (bottom). As expected, we can observe that the optimal γ value decreases when the imbalance increases. However, note that from a certain IR (around 15), γ stops decreasing to be able to keep a satisfactory F-Measure.

4.4.3 Comparison with ME^2

In the previous chapter, we have seen that ME^2 classification rule is also based on the the distance of a new query to its nearest neighbor. For this reason, we propose to compare γk-NN with ME^2 in this section using the datasets described in Chapter 3. For a fair comparison with ME^2, γk-NN is Applied with $k = 1$ and a simple 2-fold cross validation is used to tune the parameter γ. The results of this experiment is presented in Table 4.3 and show that ME^2 outperforms γk-NN on most of the datasets and especially on the most imbalanced ones. This observation means that the use of a single parameter to modify the distance to a positive example is not enough. While γk-NN uniformly modifies the distance to positives in the space, ME^2 takes the geometry of the data into account (i.e. the importance of each feature) to compute the distance to the positive examples. Thus, we think that the proposed algorithm can be greatly improved by learning a similarity matrix $\mathbf{M}$ which will be used in the computation of the distance to positive examples. Furthermore ME^2 is also building a frontier outside which any new query is automatically predicted negative, a frontier that does not exist with γk-NN.

DATASETS	YEAST3	ABALONE	WINE	ABALONE17	YEAST6	ABALONE20	BANK FRAUD
γk-NN	0.67 ± 0.05	0.60 ± 0.02	0.10 ± 0.07	0.05 ± 0.08	0.39 ± 0.15	0.11 ± 0.11	0.03 ± 0.03
ME^2	0.67 ± 0.06	0.61 ± 0.03	**0.20 ± 0.09**	**0.46 ± 0.07**	**0.46 ± 0.08**	**0.30 ± 0.07**	**0.05 ± 0.02**

Table 4.3: Comparison of ME^2 with γk-NN with $k = 1$. The values correspond to the mean F-measure F_1 over 10 runs. The standard deviation is indicated between brackets. The best result on each dataset is indicated in bold.

4.5 Conclusion

In this chapter, we have proposed a new strategy that addresses the problem of learning from imbalanced datasets, based on the $k-$NN algorithm and that modifies the distance to the positive examples. It has been shown to outperform its competitors in term of F_1-measure. Furthermore, the proposed approach is complementary to oversampling strategies and can even increase their performance. Our $\gamma k-$NN algorithm, despite its simplicity, is highly effective even on real data sets.

Two lines of research deserve future investigations. We can note that tuning γ is equivalent to building a diagonal matrix (with γ^2 in the diagonal) and applying a Mahalanobis distance only between a query and a positive example. This comment opens the door to a new metric learning algorithm dedicated to optimizing a PSD matrix under F-Measure-based constraints. If one can learn such a matrix, the second perspective will consist in deriving generalization guarantees over the learned matrix. In addition, making γ non-stationary (a $\gamma(\mathbf{x})$ that smoothly varies in $\mathcal{X}$) would increase the model flexibility.

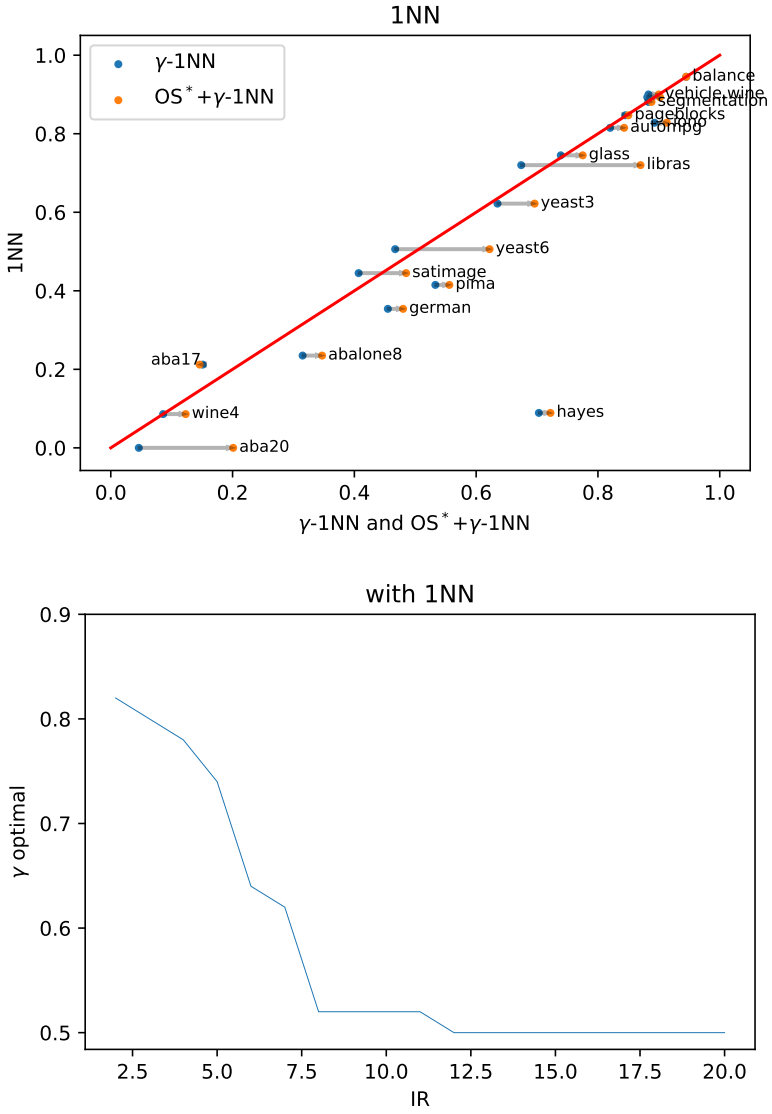

Figure 4.7: Top: comparison of k-NN with (i) γk-NN (points in blue) and (ii) γk-NN coupled with the best sampling strategy (OS*) (points in orange) for each dataset with $k = 1$. Points below the line $y = x$ means that k-NN is outperformed. Bottom: evolution of the optimal γ value with respect to the IR.

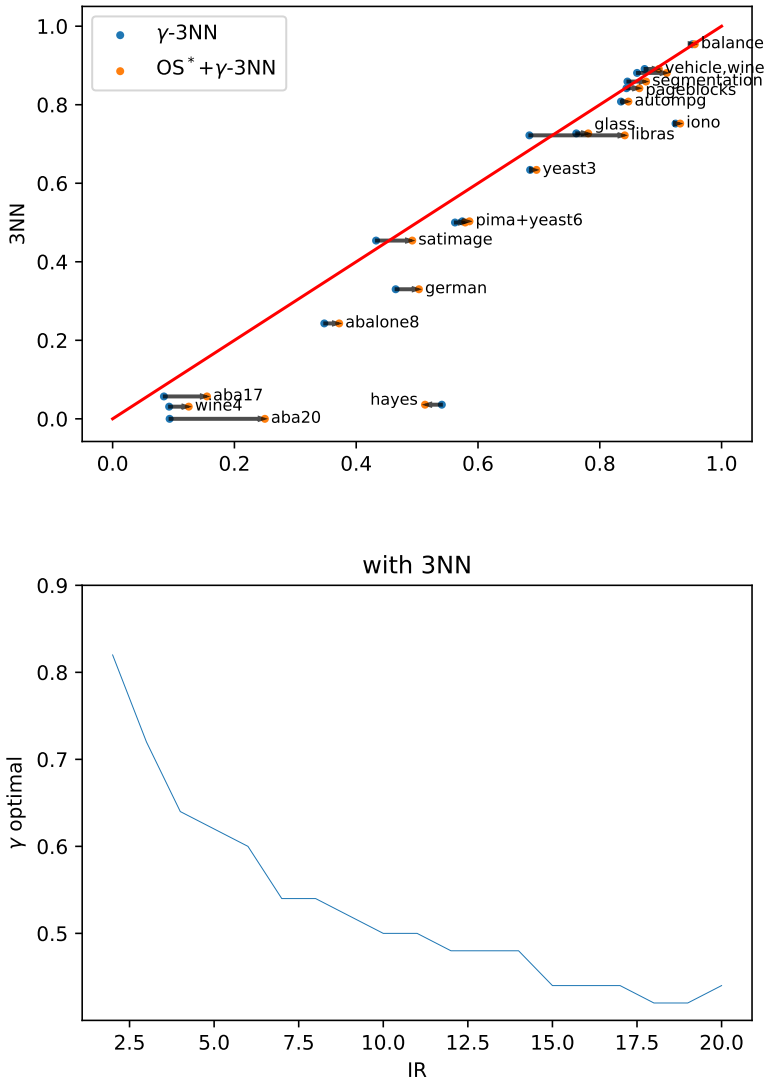

Figure 4.8: Top: comparison of k-NN with (i) γk-NN (points in blue) and (ii) γk-NN coupled with the best sampling strategy (OS*) (points in orange) for each dataset with $k = 3$. Points below the line $y = x$ means that k-NN is outperformed. Botoom: evolution of the optimal γ value with respect to the IR.

Part III

Cost-Sensitive Approaches

Chapter 5

From Cost-Sensitive Classification to Tight F-measure Bounds

This chapter is based on the following publications

Kevin Bascol, Rémi Emonet, Elisa Fromont, Amaury Habrard, Guillaume Metzler, and Marc Sebban. From cost-sensitive classification to tight F-measure bounds. In *In Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics (AISTATS-19)*, 2019

Kevin Bascol, Rémi Emonet, Elisa Fromont, Amaury Habrard, Guillaume Metzler, and Marc Sebban. Un algorithme d'optimisation de la F-mesure par pondération des erreurs de classification. In *Conférence francophone sur l'Apprentissage Automatique (CAp-18)*, 2018

Abstract

We have seen that the F-measure is a classification performance criterion, especially suited when dealing with imbalanced datasets. As this measure is non convex and non linear, its optimization remains a challenging task. In this chapter, we present the derivation of tight bounds on the optimal F-measure using cost-sensitive methods, i.e. by assigning costs on false positives and false negatives. Rewriting the F-measure as a linear fractional function, we show that the optimization of the F-measure can be reduced to a cost-sensitive task where the costs depend on an unknown parameter t. We then show that we can upper-bound the difference of F-measures between their respective proportion of false negatives and false positives and two values of t to get our final bound. A geometric interpretation of this bound is then given and used to create a new algorithm, **CONE**, which aims to iteratively optimize the F-measure by building non-reachable areas in the $(t, F)-$space and that directly looks for the value of t for which we can hope maximizing the F-measure.

The effectiveness of the proposed approach is tested on several imbalanced datasets. We show that the derived bounds are much tighter than the existing one introduced

in Parambath et al. (2014) From an experimental point of view, we show that our algorithm converges to the optimal value. Finally, we provide evidences that **CONE** is able to reach at least comparable and even better results than some state of the art methods with less iterations.

5.1 Related Work

We have seen that the F-measure (van Rijsbergen, 1974) is a performance measure used in classification to evaluate the ability of a classifier to predict the labels of new instances with a good recall and a good precision (as an harmonic mean of these two measures). It is the most commonly used measure in imbalanced settings where using the accuracy of the classifier would greatly favor the majority class (Chandola et al., 2009; Lopez et al., 2013). As already mentioned, this measure can be parameterized by a constant β that controls the relative importance of the precision and the recall. For $\beta < 1$ (resp. $\beta > 1$), more importance is given to the precision (resp. recall). When $\beta = 1$, they are considered equally important. The F-measure can be expressed in terms of the true positive rate and true negative rate of the model. These rates are count-based measures which makes the F-measure, in addition to being non convex, unsuitable for direct optimization (Narasimhan et al., 2015a).

Several methods have been proposed to optimize such a measure. They can roughly be separated into two categories: *Decision Theoretic Approaches (DTA)* (Dembczyński et al., 2017) which try to find the classifier that maximizes the expectation of the F-measure. More precisely, these methods usually fit a probabilistic model during the training followed by an inference procedure at test time (Decubber et al., 2018). The probabilistic model can be learned by optimizing a "simpler" surrogate function (e.g., (Dembczynski et al., 2011; Jansche, 2005; Ye et al., 2012; P.M. Chinta and Murty, 2013)). The second category consists of *Empirical Utility Maximization (EUM)* methods that learn multiple accurate models with different parameters and keep the model that maximizes the F-measure (Busa-Fekete et al., 2015; Joachims, 2005; Musicant et al., 2003; Parambath et al., 2014; Zhao et al., 2013; Narasimhan et al., 2015b). Here, the parameters can be the different classification thresholds for probabilistic models (Busa-Fekete et al., 2015; Joachims, 2005; Zhao et al., 2013; Narasimhan et al., 2015b) or the costs on the classification errors for cost-sensitive methods (Musicant et al., 2003; Parambath et al., 2014; Koyejo et al., 2014). Ye et al. (2012) have shown that these two types of approaches asymptotically give the same results. They also propose heuristics to decide on the category to use depending on the context.

The work presented in this chapter falls into the *EUM* based-methods and is built from a cost-sensitive classification approach that has been shown to be relevant and efficient to optimize the F-measure (Parambath et al., 2014).

For this reason, we have decided to compare our work with the one presented by Koyejo et al. (2014); Parambath et al. (2014) and Narasimhan et al. (2015a). Both authors used a cost-sensitive approach in a different manner, using either SVMs or a *Class Probability Estimator* (CPE) such as Logistic Regression as classification algorithms.

To maximize the F-measure, Koyejo et al. (2014) proposed two different approaches, each of which is based on CPE which gives the probability $\eta(\mathbf{x})$ that an example $\mathbf{x}$ belongs to a given class, typically the positive class $y = 1$, $\eta(\mathbf{x}) = Pr(y = 1 \mid \mathbf{x})$. This probabilistic model is learned by minimizing a surrogate of the 0-1 loss and the returned probability is compared to a threshold δ to decide in which class the examples belong to. This decision threshold is tuned in order to optimize the F-measure on a validation set. Despite its simplicity, this method has been shown to be F-consistent, i.e. the estimated value of the F-measure converges to the optimal value when the number of examples is growing.
A similar approach is proposed in the same paper. This time, the authors learn their probabilistic model using a weighted surrogate ℓ of the 0-1 loss parameterized by a constant δ. This parameter is used to assign costs on false negatives and false positives as follows:

$$\ell_\delta(\eta, y) = (1 - \delta)\mathbb{1}_{\{y=1\}}\ell(\eta, 1) + \delta\mathbb{1}_{\{y=0\}}\ell(\eta, 0),$$

where η is the CPE and $\ell(\eta, 1)$ (resp. $\ell(\eta, 0)$) is the value of the surrogate when a positive example is missclassified (resp. when a negative example is missclassified). The use of δ as a weighted parameter of the loss function is not innocent. In fact, Scott (2012) has shown that the optimal classifier used to optimize this loss is $sign[\eta(\mathbf{x}) - \delta]$. It remains to find the best threshold, thus, the best costs to assign to each class, to maximize the F-measure. For this purpose, the authors proposed to make a grid on $[0, 1]$ and to find the value of δ that maximizes the F-measure on a validation set. This method constitutes the main weakness of their method from a practical aspect by requiring a sufficiently fine grid.

The use of a grid is also proposed by Parambath et al. (2014) to optimize the F-measure. However, they present their study in the framework of *fractional-linear* functions, i.e. functions defined as the ratio of two affine functions. The authors make an interesting link between the definition of the F-measure and cost-sensitive learning using the fact that the F-measure is *fractional-linear* as we will see in the next section. Using this fact, they are able to provide a bound on the optimal F-measure, a bound we aim to improve in this chapter. We will show that, in practice, this bound is not informative unless the used grid is small enough. This method suffers from the same practical weakness as the previous approach.

Choosing the cost to assign to each class is crucial if we aim to optimize the F-measure and using a simple grid to find the optimal costs can be time consuming. To overcome this issue, Narasimhan et al. (2015b) proposed an algorithm to iteratively update the costs assigned to each class (and the optimal threshold undirectly). Like Parambath et al. (2014), this method is based on the fractional linear property of the F-measure and on the *bisection method* (Boyd and Vandenberghe, 2004). It has been shown to be more effective and faster than a grid based method. The weakness of their method lies in the fact that a single CPE is learned which is totally independant from the cost function and we will show that it is possible to improve the performance by learning a model per cost without falling in the weakness of Parambath et al. (2014) and Koyejo et al. (2014).

According to the above mentioned reasons, our contribution in this chapter is in three-fold:

1. From a theoretical perspective, we improve the bound given by Parambath et al. (2014),
2. We give a geometric interpretation of the obtained bound,
3. We derive an algorithm, **CONE**, which updates the cost to assign to each class using our bound.

The next section introduces the required notations and definitions used for our purpose.

5.2 Theoretical Bounds

In this section, we present how we can derive a tight bound on the optimal F-measure using the cost-sensitive framework and the pseudo-linearity property of the F-measure. We only present these results in the binary setting but they can easily be extended to the multi-class setting[1].

5.2.1 Notations

Let $\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_m)$, where $\mathbf{x}_i \in \mathbb{R}^d$, be the set of m training instances and $\mathbf{Y} = (y_1, ..., y_m)$ their corresponding labels, where $y \in \{0, 1\}$. Let $\mathcal{H}$ be a family of hypotheses e.g., linear separators.

Let P (resp. N) be the proportion of positive (resp. negative) instances. We also denote by $\mathbf{e} = \mathbf{e}(h)$ the vector (e_1, e_2) where e_1 and e_2 are respectively the proportion of false negatives (FN) and false positives (FP) obtained by a classifier $h \in \mathcal{H}$ on a dataset S. We then denote as $\mathcal{E}(\mathcal{H})$ the set of all possible error profiles $\mathbf{e}(h)$ for a given set of hypotheses $\mathcal{H}$: an error profile $\mathbf{e} = (e_1, e_2)$ is in $\mathcal{E}(\mathcal{H})$ if there exists an hypothesis $h \in \mathcal{H}$ that yields proportions of e_1 false negatives and e_2 false positives.

Let us recall the definition of the F-measure for any value of β:

$$F_\beta = \frac{(1+\beta^2)(P - FN)}{(1+\beta^2)P - FN + FP}.$$

Using the above notations, the F-measure, $F_\beta(\mathbf{e})$, defined in terms of the error profile $\mathbf{e}$ can be rewritten as:

$$F_\beta(\mathbf{e}) = \frac{(1+\beta^2)(P - e_1)}{(1+\beta^2)P - e_1 + e_2}. \tag{5.1}$$

This second writing will allow us to lighten the notations later on by manipulating the error vector $\mathbf{e}$. Furthermore, in the following, we will replace $F_\beta(\mathbf{e})$ by $F(\mathbf{e})$ since the used F-measure will always depend on $\beta > 0$.

[1] See Appendix A.1 for complete details about the multi-class setting.

5.2.2 Pseudo linearity property

As shown in Equation (5.1), the F-measure is a linear-fractional function, i.e. it can be written as the ratio of two affine functions of the error profile. We briefly recall how to show that the F-measure is a pseudo-linear function, which is one of the main properties of linear-fractional functions. This property is the starting point of our main theoretical result.

Definition 5.1. *[Pseudo Convexity Rapcsák (1991)] A real differentiable function f defined on an open convex set $\mathcal{C} \subset \mathbb{R}^q$ is said to be pseudo-convex if for every $\mathbf{e}, \mathbf{e}' \in \mathcal{C}$,*

$$\langle \nabla f(\mathbf{e}), (\mathbf{e}' - \mathbf{e}) \rangle \geq 0 \implies f(\mathbf{e}') \geq f(\mathbf{e}),$$

where ∇f denotes the gradient of the function f.

The pseudo-convexity is used to define the pseudo-linearity as shown below.

Definition 5.2. *[Pseudo Linearity] A function f defined on an open convex $\mathcal{C}$ is said to be pseudo-linear if both f and $-f$ are pseudo-convex.*

A first step consists in showing that the F-measure is pseudo-linear.

Proposition 5.1. *The F-measure is a pseudo-linear function.*

Proof. We need to show that both F and $-F$ are pseudo-convex, i.e. for all $\mathbf{e}, \mathbf{e}' \in \mathcal{E}(\mathcal{H})$, we have:

$$\langle \nabla F(\mathbf{e}), (\mathbf{e}' - \mathbf{e}) \rangle \geq 0 \implies F(\mathbf{e}') \geq F(\mathbf{e}). \tag{5.2}$$

For all $\mathbf{e}, \mathbf{e}' \in \mathcal{E}(\mathcal{H})$ the gradient of the F-measure is defined by:

$$\nabla F(\mathbf{e}) = -\frac{1+\beta^2}{((1+\beta^2)P - e_1 + e_2)^2} \begin{pmatrix} \beta^2 P + e_2 \\ P - e_1 \end{pmatrix}.$$

We now develop the left hand side of the implication (5.2):

$$\begin{aligned} 0 \leq\ & \langle \nabla F(\mathbf{e}), (\mathbf{e}' - \mathbf{e}) \rangle, \\ & \downarrow \text{ using the expression of the gradient} \\ 0 \leq\ & -\frac{1+\beta^2}{((1+\beta^2)P - e_1 + e_2)^2} \left[(\beta^2 P + e_2)(e_1' - e_1) + (P - e_1)(e_2' - e_2) \right]. \end{aligned}$$

Dividing both sides of the previous inequality by $-\frac{1+\beta^2}{((1+\beta^2)P - e_1 + e_2)^2} < 0$, we get:

$$\begin{aligned} 0 \ \leq\ & -(\beta^2 P + e_2)(e_1' - e_1) - (P - e_1)(e_2' - e_2), \\ & \downarrow \text{ developping} \\ 0 \ \leq\ & -\beta^2 P(e_1' - e_1) - e_1' e_2 + \cancel{e_1 e_2} + P(e_2 - e_2') + e_1 e_2' - \cancel{e_1 e_2}, \\ & \downarrow \text{ simplifying by } e_1 e_2 \\ 0 \ \leq\ & -\beta^2 P(e_1' - e_1) + P(e_2 - e_2') + e_1 e_2' - e_1' e_2, \end{aligned}$$

$$\downarrow \text{ developping again}$$
$$0 \leq -\beta^2 Pe'_1 + \beta^2 Pe_1 + Pe_2 - Pe'_2 + e_1e'_2 - e'_1e_2.$$

We then separate the terms on both sides of the inequality as follows:

$$-\beta^2 Pe'_1 + Pe_2 - e'_1e2 \geq -\beta^2 Pe_1 + Pe'_2 - e_1e'_2$$
$$\downarrow \text{ substracting } P(e_1 + e'_1) \text{ on both sides}$$
$$-\beta^2 Pe'_1 + Pe_2 - e'_1e_2 - P(e_1 + e'_1) \geq -\beta^2 Pe_1 + Pe'_2 - e_1e'_2 - P(e_1 + e'_1),$$
$$\downarrow \text{ factorizing}$$
$$-(1+\beta^2)Pe'_1 + Pe_2 - e'_1e_2 - Pe_1 \geq -(1+\beta^2)Pe_1 + Pe'_2 - e_1e'_2 - Pe'_1.$$

Then, we add $e_1e'_1$ on both sides and factorize the expression

$$-(1+\beta^2)Pe'_1 + Pe_2 - e'_1e_2 - Pe_1 + e_1e'_1 \geq -(1+\beta^2)Pe_1 + Pe'_2 - e_1e'_2 - Pe'_1 + e_1e'_1,$$
$$-(1+\beta^2)Pe'_1 + (P - e'_1)e_2 - (P - e'_1)e_1 \geq -(1+\beta^2)Pe_1 + (P - e_1)e'_2 - (P - e_1)e'_1.$$

Finally, by adding $(1+\beta^2)P^2$ on both sides we get:

$$(1+\beta^2)P(P - e'_1) - (P - e'_1)e_1 + (P - e'_1)e_2 \geq (1+\beta^2)P(P - e_1) - (P - e_1)e'_1 + (P - e_1)e'_2,$$
$$\downarrow \text{ factorizing left hand side by } P - e_1$$
$$\downarrow \text{ factorizing right hand side by } P - e'_1$$
$$(P - e'_1)((1+\beta^2)P - e_1 + e_2) \geq (P - e_1)((1+\beta^2)Pe'_1 + e'_2),$$
$$\downarrow \text{ dividing by } (1+\beta^2)P - e_1 + e_2$$
$$\downarrow \text{ dividing by } (1+\beta^2)P - e'_1 + e'_2$$
$$\frac{(P - e'_1)}{(1+\beta^2)P - e'_1 + e'_2} \geq \frac{(P - e_1)}{(1+\beta^2)P - e_1 + e_2},$$
$$\downarrow \text{ multiplying both sides by } 1+\beta^2$$
$$\frac{(1+\beta^2)(P - e'_1)}{(1+\beta^2)P - e'_1 + e'_2} \geq \frac{(1+\beta^2)(P - e_1)}{(1+\beta^2)P - e_1 + e_2},$$
$$\downarrow \text{ definition of the F-measure}$$
$$F(\mathbf{e}') \geq F(\mathbf{e}).$$

The proof is similar for $-F$. We have shown that both F and $-F$ are pseudo-convex so F is pseudo-linear. □

Using this property, we are able, using a result from Cambini and Martein (2009) to give a link between the F-measure and a cost-sensitive function, i.e. a function which assigns weights to each class.

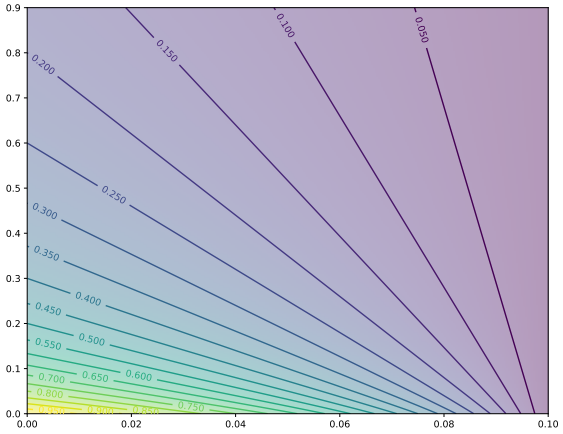

Figure 5.1: Level sets of the F-measure for $\beta = 1$ in the $(e_1, e_2)-$space with the corresponding value of the F-measure.

Proposition 5.2. *[Theorem 3.3.9 from Cambini and Martein (2009)] Let f be a non-constant differentiable function on an open convex set $\mathcal{C} \subset \mathbb{R}^q, q > 0$. Then f is pseudo-linear on $\mathcal{C}$ if and only if the following properties hold:*
(i) each of the level sets of f is the intersection of $\mathcal{C}$ with a hyperplane;
(ii) $\nabla f(\mathbf{e}) \neq 0$ for all $\mathbf{e} \in \mathcal{C}$.

Let us consider the set of error profiles $\{\mathbf{e} \in \mathbb{R}^2 \mid (1+\beta^2)P - e_1 + e_2 > 0\}$ (which is always the case in practice with the F-measure). Then according to the previous theorem, we rewrite (i) as follows:
There is $\mathbf{a} : \mathbb{R} \to \mathbb{R}^2$ and $b : \mathbb{R} \to \mathbb{R}$ such that

$$F(\mathbf{e}) = t \iff \langle \mathbf{a}(t), \mathbf{e} \rangle + b(t) = 0,$$

which can be rewritten:

$$\langle \mathbf{a}(F(\mathbf{e})), \mathbf{e} \rangle + b(F(\mathbf{e})) \quad = \quad 0. \tag{5.3}$$

The above proposition is illustrated on Figure 5.1. The level sets are represented in the $(e_1, e_2)-$space by the straight lines which can be seen as affine hyperplanes in a space of any dimension.

For the F-measure, the functions $\mathbf{a}$ and b are defined by $\mathbf{a}(t) = (1 + \beta^2 - t, t)$ and $b(t) = (1 + \beta^2)P(t - 1)$. The term $\langle \mathbf{a}(t), \mathbf{e} \rangle$ can be seen as a weighted error loss function, and thus $\mathbf{a}(t)$ can be seen as the costs to assign to each class.

This loss function, thus its associated empirical risk will be used, essentially in the experimental part, as the quantity to minimize to maximize the F-measure. The link between the minimization of such a risk and the F-measure maximization is explained in the Proposition 4 of Parambath et al. (2014) and is mainly based on the pseudo-linearity of the F-measure.

5.2.3 Bounds on the optimal F-measure

We now show the importance of the function $\mathbf{a}$ and of the parameter t to characterize the difference of F-measures between any two error profiles.

Step 1: impact of a change in the error profile

We first derive the relation between the difference in F-measures and the difference in error profiles. Let $\mathbf{e}$ and $\mathbf{e}'$ be any two error profiles and $F(\mathbf{e})$ and $F(\mathbf{e}')$ the corresponding F-measures.

From the pseudo-linearity property (Equation (5.3)), we have:

$$0 = \langle \mathbf{a}(F(\mathbf{e})), \mathbf{e} \rangle + b(F(\mathbf{e})), \tag{5.4}$$
$$0 = \langle \mathbf{a}(F(\mathbf{e}')), \mathbf{e}' \rangle + b(F(\mathbf{e}')). \tag{5.5}$$

We now develop $\langle \mathbf{a}(F(\mathbf{e}')), \mathbf{e} - \mathbf{e}' \rangle$ and make the difference in F-measures appears in its expression.

$$\begin{aligned}
\langle \mathbf{a}(F(\mathbf{e}')), \mathbf{e} - \mathbf{e}' \rangle = \; & \langle \mathbf{a}(F(\mathbf{e}')), \mathbf{e} \rangle + b(F(\mathbf{e}')), \\
& \downarrow \text{ using Equation (5.4)} \\
= \; & \underbrace{\langle \mathbf{a}(F(\mathbf{e}')), \mathbf{e} \rangle - \langle \mathbf{a}(F(\mathbf{e})), \mathbf{e} \rangle}_{} \underbrace{- b(F(\mathbf{e})) + b(F(\mathbf{e}'))}_{}, \\
& \downarrow \text{ using the linearity of the inner product and } b \\
= \; & \underbrace{\langle \mathbf{a}(F(\mathbf{e}')) - \mathbf{a}(F(\mathbf{e})), \mathbf{e} \rangle}_{} + P(1+\beta^2)(F(\mathbf{e}') - F(\mathbf{e})), \\
& \downarrow \text{ using the definition of } \mathbf{a} \\
= \; & (\mathbf{e}_2 - \mathbf{e}_1)(F(\mathbf{e}') - F(\mathbf{e})) + P(1+\beta^2)(F(\mathbf{e}') - F(\mathbf{e})), \\
& \downarrow \text{ factorizing} \\
\langle \mathbf{a}(F(\mathbf{e}')), \mathbf{e} - \mathbf{e}' \rangle = \; & \big(F(\mathbf{e}') - F(\mathbf{e})\big) \cdot \big((1+\beta^2)P - e_1 + e_2\big),
\end{aligned}$$

where the second line uses Equation (5.4). The third one uses the linearity of the inner product and the definition of b. The fourth one uses Equation (5.4) and the last line uses the definition of $\mathbf{a}$ and b defined in Section 5.2.2.

Now we can rewrite the difference in F-measures as:

$$F(\mathbf{e}') - F(\mathbf{e}) = \Phi_{\mathbf{e}} \cdot \langle \mathbf{a}(F(\mathbf{e}')), \mathbf{e} - \mathbf{e}' \rangle, \tag{5.6}$$

where $\Phi_{\mathbf{e}} = \dfrac{1}{(1+\beta^2)P - e_1 + e_2}$.

Step 2: a first bound on the F-measure $F(\mathbf{e}')$

We suppose that we have a value of t for which a weighted-classifier with weights $\mathbf{a}(t)$ has been learned. This classifier has an error profile $\mathbf{e}$ and a F-measure $F(\mathbf{e})$. Note that the value

t used to trained the model is not the value of the F-measure, i.e. $F(\mathbf{e}) \neq t$. We keep this notation for the sake of simplicity in the following. We now imagine a hypothetical classifier which leads to an error profile $\mathbf{e}^\star$ the optimal error profile, i.e. the one that maximized the F-measure and for which we have $F(\mathbf{e}^\star) = t^\star$ (see Proposition 4 of Parambath et al. (2014)). Starting from the result obtained in Equation (5.6), we have:

$$
\begin{aligned}
F(\mathbf{e}^\star) - F(\mathbf{e}) = \quad & \Phi_{\mathbf{e}} \left(\underbrace{\langle \mathbf{a}(t^\star), \mathbf{e} \rangle} - \langle \mathbf{a}(t^\star), \mathbf{e}^\star \rangle \right), \\
& \downarrow \text{ using } \mathbf{a}' = \mathbf{a} + \mathbf{a}^\star - \mathbf{a} \\
= \quad & \Phi_{\mathbf{e}} \left(\langle \mathbf{a}(t), \mathbf{e} \rangle + \underbrace{\langle \mathbf{a}(t^\star) - \mathbf{a}(t), \mathbf{e} \rangle} - \langle \mathbf{a}(t^\star), \mathbf{e}^\star \rangle \right), \\
& \downarrow \text{ using the defintion of } \mathbf{a} \\
= \quad & \Phi_{\mathbf{e}} \left(\underbrace{\langle \mathbf{a}(t), \mathbf{e} \rangle} + (t^\star - t)(e_2 - e_1) - \langle \mathbf{a}(t^\star), \mathbf{e}^\star \rangle \right), \\
& \downarrow \text{ introducing the sub optimality of the learned classifier} \\
\leq \quad & \Phi_{\mathbf{e}} \left(\underbrace{- \langle \mathbf{a}(t^\star), \mathbf{e}^\star \rangle + \langle \mathbf{a}(t), \mathbf{e}^\star \rangle} + \varepsilon_1 + (t^\star - t)(e_2 - e_1) \right), \\
& \downarrow \text{ using the definition of } \mathbf{a} \\
\leq \quad & \Phi_{\mathbf{e}} \left((t - t^\star)(e_2^\star - e_1^\star) + \varepsilon_1 + (t^\star - t)(e_2 - e_1) \right), \\
F(\mathbf{e}^\star) - F(\mathbf{e}) \leq \quad & \Phi_{\mathbf{e}} \varepsilon_1 + \Phi_{\mathbf{e}} \cdot (e_2 - e_1 - (e_2^\star - e_1^\star))(t^\star - t). \qquad (5.7)
\end{aligned}
$$

We have successively used the linearity of the inner product, introduced $\mathbf{a}(t)$ and its definition in the first three equalities. The first inequality uses $\langle \mathbf{a}(t), \mathbf{e} \rangle \leq \langle \mathbf{a}(t), \mathbf{e}_{best} \rangle + \varepsilon_1$, the sub-optimality of the $\mathbf{a}(t)$-weighted-error classifier. The value of ε_1 represents the excess of risk of the classifier which aims to minimize the $\mathbf{a}(t)$-weighted-error. More precisely, it represents the difference of risks between our classifier and the best classifier h_{best} (in terms of $\mathbf{a}(t)$-weight-error) in our set of hypotheses $\mathcal{H}$. We denote by $\mathbf{e}_{best}$ the error profile associated to h_{best}.

We are interested in upper bounding the optimal value of the F-measure using equation (5.7).

For this purpose, we consider any possible value of t', for which we learn a hypothetical classifier with weights $\mathbf{a}(t')$, that gives us an error profile $\mathbf{e}'$ and a F-measure $F(\mathbf{e}')$. If t' happens to be the optimal value which, by weighting the errors with $\mathbf{a}(t')$, maximizes the F-measure then $F(\mathbf{e}') = t'$ for $\mathbf{e}' = \underset{\hat{\mathbf{e}} \in \mathcal{E}(\mathcal{H})}{\operatorname{argmin}} \langle \mathbf{a}(t'), \hat{\mathbf{e}} \rangle$. Equation (5.7) can then be applied and gives:

$$
F(\mathbf{e}') - F(\mathbf{e}) \leq \quad \Phi_{\mathbf{e}} \varepsilon_1 + \Phi_{\mathbf{e}} \cdot (e_2 - e_1 - (e_2' - e_1'))(t' - t)
$$

By this way, the above equation can be interpreted in terms of two distinct variables: (i) the value t' and (ii) the achieved F-measure $F(\mathbf{e}')$ (if t' would be optimal) with respect to the value t'. Driven by this dissociation between the weighting t' and the F-measure $F(\mathbf{e}')$,

Section 5.3 explains how it can be used to look for the optimal weighting parameter $t^\star$ and, thus, to the optimal value $F(\mathbf{e}^\star)$. But before giving the interpretation of our bound, it remains to show that the difference $\mathbf{e}'_2 - \mathbf{e}'_1$ can be tightly bounded.

Step 3: a tight bound on $\mathbf{e}'_2 - \mathbf{e}'_1$

The bound described by Equation (5.7) remains actually incomplete as stated before. To complete this bound we need to study the difference $e'_2 - e'_1$. We aim to bound this difference according to the sign of $t - t'$ and under the constraint that $\mathbf{e}'$ is such that $F(\mathbf{e}') > F(\mathbf{e})$ in order to have a tighter bound.

We first need a preliminary result to explain if we have to give an upper bound or a lower bound on $e'_2 - e'_1$.

Lemma 5.1. *The difference $(e_1 - e_2)(t)$ is increasing when $\mathbf{e}(t)$ is obtained from an optimal weighted-classifier trained with costs $\mathbf{a}(t)$.*

Proof. Let $t > t'$, $\mathbf{e}(t)$ and $\mathbf{e}(t')$ the vector of missclassified examples obtained with an optimal classifier (in the Bayes sense) trained with costs $\mathbf{a}(t)$ and $\mathbf{a}(t')$ respectively. We thus have the following inequalities:

$$t \cdot e_2(t) + (1 + \beta^2 - t)\, e_1(t) \leq t \cdot e_2(t') + (1 + \beta^2 - t)\, e_1(t'),$$

and

$$t' \cdot e_2(t') + (1 + \beta^2 - t')\, e_1(t') \leq t' \cdot e_2(t) + (1 + \beta^2 - t')\, e_1(t).$$

By multiplying the second equation by -1 and summing the two equations, we get:

$$(t - t')(e_1(t) - e_2(t) \geq (t - t')(e_1(t') - e_2(t')).$$

Thus:

$$e_1(t') - e_2(t') \leq e_1(t) - e_2(t).$$

□

According to this Lemma the difference $(e_1 - e_2)(t)$ is an increasing function of t, thus, when $t' < t$, $e_2 - e_1 - (e'_2 - e'_1) < 0$ and $e'_2 - e'_1$ shall be maximized to have a tight bound that preserves this inequality. Similarly, when $t' > t$, we have to minimize the difference $e'_2 - e'_1$. Maximizing (resp. minimizing) this difference consists of a constraint optimization problem. The following Lemma gives the analytical solution of the two optimization problems.

Lemma 5.2. *Let $\mathbf{e}$ be the error profile obtained with an hypothesis h learned with the cost function $\mathbf{a}(t)$. Let $\mathbf{e}'$ be an error profile such that $F(\mathbf{e}') > F(\mathbf{e})$. Then the difference $e'_2 - e'_1$ is equal to:*

$$M_{max} = \max_{\substack{\mathbf{e}' \in \mathcal{E}(\mathcal{H}) \\ s.t.\ F(\mathbf{e}') > F(\mathbf{e})}} (e'_2 - e'_1) = e_2 + \min\left(N - e_2, \frac{e_1(\beta^2 P + e_2)}{P - e_1}\right) \quad \textit{when} \quad t' < t,$$

and

$$M_{min} = \min_{\substack{\mathbf{e}' \in \mathcal{E}(\mathcal{H}) \\ s.t.\ F(\mathbf{e}') > F(\mathbf{e})}} (e'_2 - e'_1) = -e_1 - \frac{e_2(P - e_1)}{\beta^2 P + e_2} \quad when \quad t' > t.$$

Proof. Before computing the value of both $M_{\max}$ and $M_{\min}$, we show that it is possible to rewrite the constraint of the optimization problem as a linear constraint over a rectangle.

a) Rewriting the constraint

In the binary setting, we recall that $\mathbf{e} = (e_1, e_2)$ and $\mathbf{e}' = (e'_1, e'_2)$. The constraint $F_\beta(\mathbf{e}') > F_\beta(\mathbf{e})$ can be rewritten:

$$\frac{\cancel{(1+\beta^2)}(P - e'_1)}{(1+\beta^2)P - e'_1 + e'_2} > \frac{\cancel{(1+\beta^2)}(P - e_1)}{(1+\beta^2)P - e_1 + e_2},$$

↓ reducing to the same denominator

$$(P - e'_1)[(1+\beta^2)P - e_1 + e_2] > (P - e_1)[(1+\beta^2)P - e'_1 + e'_2]),$$

$$\cancel{(1+\beta^2)P^2} - (1+\beta^2)Pe'_1 + (P - e'_1)(e_2 - e_1) > \cancel{(1+\beta^2)P^2} - (1+\beta^2)Pe_1 + (P - e_1)(e'_2 - e'_1),$$

↓ simplifying by $(1+\beta^2)P^2$ and factorizing

$$(1+\beta^2)P(e_1 - e'_1) + P(e_2 - e_1 + e'_1 - e'_2) > e_2e'_1 - e_1e'_2 + \cancel{e'_1e_1} - \cancel{e_1e'_1}$$

From now on, we set: $e'_1 = e_1 + \alpha_1$ and $e'_2 = e_2 + \alpha_2$. In other words, we study how $\mathbf{e}'$ shall deviate from $\mathbf{e}$ to solve our optimization problem. We can then write:

$$-(1+\beta^2)P\alpha_1 + P(\alpha_1 - \alpha_2) > e_2(e_1 + \alpha_1) - e_1(e_2 + \alpha_2),$$

↓ simplifying by e_1e_2 and rearranging the terms

$$\underbrace{\alpha_1(-(1+\beta^2)P + P - e_2)}_{} + \alpha_2(-P + e_1) > 0,$$

↓ simplfying the expression

$$\alpha_1(\beta^2 P + e_2) < -\alpha_2(P - e_1).$$

We thus have the linear constraint:

$$\alpha_1 < \frac{-\alpha_2(P - e_1)}{\beta^2 P + e_2}.$$

Furthermore, the optimization problem has to be solved over a rectangle because $\alpha_1 \in [-e_1, P - e_1]$ and $\alpha_2 \in [-e_2, P - e_2]$ by definition of $\mathbf{e}'$ and because $\mathbf{e} \in [0, P] \times [0, N]$. This set of constraints is represented in Figure 5.2. In the following, we also denote by $\mathfrak{D}$ the line of equation:

$$\alpha_1 = \frac{-\alpha_2(P - e_1)}{\beta^2 P + e_2}. \tag{5.8}$$

We are now ready to compute the value of both $M_{\max}$ and $M_{\min}$.

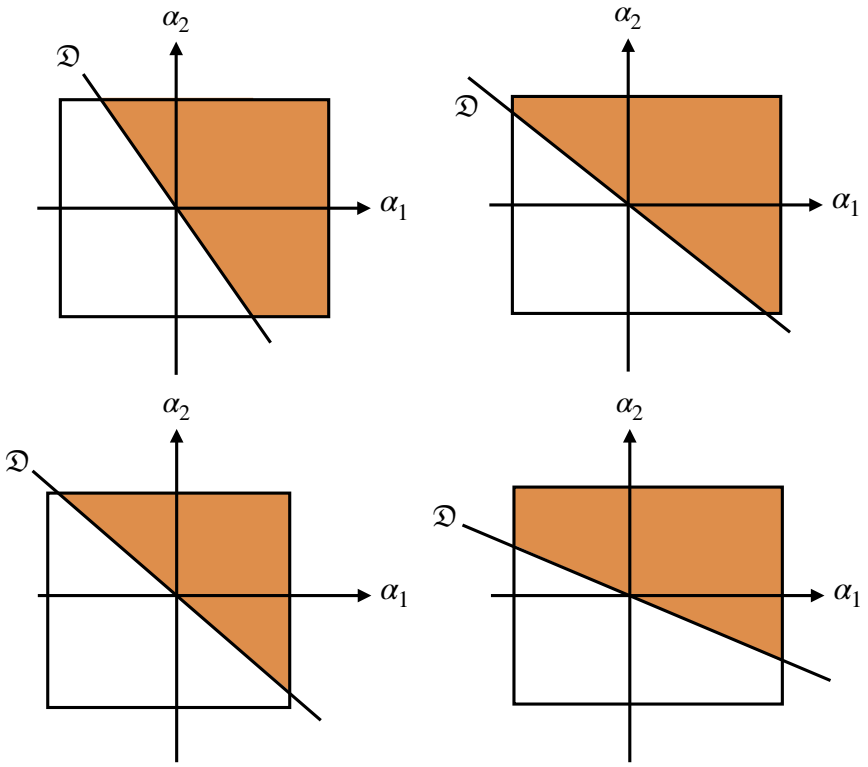

Figure 5.2: Geometric representation of the optimization problem. The rectangle represents the constraint $(\alpha_2, \alpha_1) \in [-e_2, N - e_2] \times [e_1, P - e_1]$. The white area represents the set of value (α_2, α_1) for which the inequality constraint holds. The four figures represent the four possibilities for the position of the line $\mathfrak{D}$ on the rectangle. Given the constraints of the optimization problem, the two cases represented at the bottom never happen.

b) Computation of $M_{\max}$

According to the previous step, our optimization problem can be rewritten:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \alpha_2 - \alpha_1, \\ s.t. \quad & \alpha_1 < \frac{-\alpha_2(P - e_1)}{\beta^2 P + e_2}, \\ & \alpha_1 \in [-e_1, P - e_1], \\ & \alpha_2 \in [-e_2, N - e_2]. \end{aligned}$$

The optimization problem consists of maximizing a difference under a polygon set of constraints. To maximize the difference, we should maximize the value of α_2 and minimize the value of α_1, i.e. the solution is located in the bottom right region of each graph of Figure 5.2. A quick study of these figures shows that the lowest value of α_1 we can reach is $-e_1$.

We shall now study where the line $\mathfrak{D}$ intersects the rectangle to have the solution with respect to α_2. If $\mathfrak{D}$ does not intersect the line of equation $\alpha_1 = -e_1$ in the rectangle (i.e. it intersects with the right side of the rectangle) then $\alpha_2 = N - e_2$. Else, it intersects with the bottom face of the rectangle, then we determine the value of α_2 using Equation (5.8) and $\alpha_2 = \dfrac{(\beta^2 P + e_2)e_1}{P - e_1}$.

Finally, the solution of the optimization problem is:

$$(\alpha_1, \alpha_2) = \left(-e_1, \min\left(N - e_2, \frac{(\beta^2 P + e_2)e_1}{P - e_1}\right)\right),$$

and the optimal value $M_{\max}$ is defined by:

$$M_{\max} = e_2 + \min\left(N - e_2, \frac{(\beta^2 P + e_2)e_1}{P - e_1}\right).$$

It remains now to compute the value of $M_{\min}$

c) Computation of $M_{\min}$

According to Step 1: the optimization problem can be rewritten:

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{min} \quad & \alpha_2 - \alpha_1, \\ s.t. \quad & \alpha_1 < \frac{-\alpha_2(P - e_1)}{\beta^2 P + e_2}, \\ & \alpha_1 \in [-e_1, P - e_1], \\ & \alpha_2 \in [-e_2, N - e_2]. \end{aligned}$$

The set of constraints remains unchanged. However, to minimize this difference, we have to maximize the value of α_1 and minimize the value of α_2, i.e. we are interested in the upper left region of each rectangles. In each case represented in Figure 5.2, we see that the minimum of α_2 is equal to $-e_2$.

If we have a look at the two figures at the bottom of Figure 5.2, we see that the optimal value of α_1 is equal to $P - e_1$. However, this value is not in the image of the function of α_2 defined by Equation (5.8). In fact, according to this same Equation, the image of $\alpha_2 = -e_2$ is equal to $\dfrac{e_2(P - e_1)}{\beta^2 P + e_2}$ which is lower than $P - e_1$. So the two figures at the bottom represent cases that never happen and the intersection of $\mathfrak{D}$ with the rectangle of constraints is on left part of the rectangle.

Finally, the solution of the optimization problem is:

$$(\alpha_1, \alpha_2) = \left(\frac{e_2(P - e_1)}{\beta^2 P + e_2}, -e_2\right),$$

and the optimal value M_{min} is defined by:

$$M_{min} = -e_1 - \frac{e_2(P - e_1)}{\beta^2 P + e_2}.$$

□

Step 4: bounds on the F-measure $F(\mathbf{e}')$

Proposition 5.3. *Let* $\mathbf{e}$ *be the error profile obtained with a classifier trained with the parameter* t *and* $F(\mathbf{e})$ *its associated F-measure value. Let us also consider* $\Phi_{\mathbf{e}}$ *as defined in Equation* (5.6) *and* $\varepsilon_1 > 0$ *the sub-optimality of our linear classifier. Then for all* $t' < t$*:*

$$F(\mathbf{e}') \leq F(\mathbf{e}) + \Phi_{\mathbf{e}}\varepsilon_1 + \Phi_{\mathbf{e}} \cdot (e_2 - e_1 - M_{max})(t' - t),$$

where $M_{max} = \max_{\substack{\mathbf{e}'' \in \mathcal{E}(\mathcal{H}) \\ s.t.\ F(\mathbf{e}'') > F(\mathbf{e})}} (e_2'' - e_1'')$
and, for all $t' > t$*:*

$$F(\mathbf{e}') \leq F(\mathbf{e}) + \Phi_{\mathbf{e}}\varepsilon_1 + \Phi_{\mathbf{e}} \cdot (e_2 - e_1 - M_{min})(t' - t),$$

where $M_{min} = \min_{\substack{\mathbf{e}'' \in \mathcal{E}(\mathcal{H}) \\ s.t.\ F(\mathbf{e}'') > F(\mathbf{e})}} (e_2'' - e_1'')$.

Proof. This result is a consequence of Lemmas 5.1 and 5.2. □

With this first result, we give an upper bound on the reachable F-measures for any value of t' given an observed value of F-measure with the parameter t. The bounds depend on the values of both M_{min} and M_{max} depending on the sign of $t - t'$ which leads to non symmetric slopes in the $(t, F)-$space.

The above proposition leads to the followings bounds on the optimal F-measure.

Corollary 5.1. *Given the same assumptions and considering* $t^\star$ *the value of* t *for which the best cost-sensitive learning algorithm leads to a model with an error profile* $\mathbf{e}^\star$ *associated to the optimal F-measure, we have: if* $t^\star < t$*:*

$$F(\mathbf{e}^\star) \leq F(\mathbf{e}) + \Phi_{\mathbf{e}}\varepsilon_1 + \Phi_{\mathbf{e}} \cdot (e_2 - e_1 - M_{max})(t^\star - t),$$

and, if $t^\star > t$*:*

$$F(\mathbf{e}^\star) \leq F(\mathbf{e}) + \Phi_{\mathbf{e}}\varepsilon_1 + \Phi_{\mathbf{e}} \cdot (e_2 - e_1 - M_{min})(t^\star - t).$$

This means that if we learn a model with a parameter t sufficiently close to $t^\star$ then, we guarantee to reach the optimal F-measure up to a constant equal to $\Phi_{\mathbf{e}}\varepsilon_1$.

In the next section, we present how we can use these theoretical bounds to propose a new algorithm to optimize the F-measure iteratively.

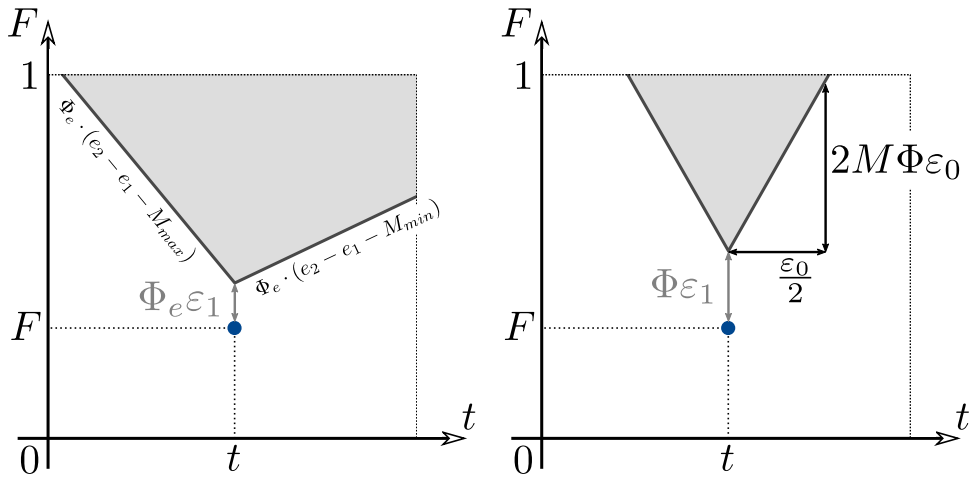

Figure 5.3: Geometric interpretation of our bound on the left and the one from Parambath et al. (2014) on the right. Note that our "cone" is not symmetric compared to the other one. On the left, the slanted values represent the slope of our cone on each side : $\Phi_{\mathbf{e}} \cdot (e_2 - e_1 - M_{max})$ and $\Phi_{\mathbf{e}} \cdot (e_2 - e_1 - M_{min})$.

5.3 CONE Algorithm

In this section, we provide a geometric interpretation of Proposition 5.3 and a comparison to the bound introduced in Parambath et al. (2014). We also show how this theoretical result can be an inspiration to create an algorithm, **CONE** for Cone-based Optimal Next Evaluation, which optimizes the F-measure by wrapping a cost-sensitive learning algorithm.

5.3.1 Unreachable regions

In Figure 5.3 (left), we give a geometric interpretation of the result from Proposition 5.3 in the 2-D space. In this (t, F) graph, the previous near-optimality result yields an upper cone of values where $F(\mathbf{e}^\star)$ cannot be found. More precisely, when a model is learned for a given value of t (with weights $\mathbf{a}(t)$), we measure the value $F(\mathbf{e})$ of this model and, given these two numbers, we are able to draw an upper cone which represents the unreachable values of F-measure for any t' on the x-axis. Furthermore, given ε_1, the sub-optimality of the cost-sensitive learning algorithm for the weighted 0-1 loss, $\Phi_{\mathbf{e}}\varepsilon_1$ corresponds to the vertical offset of this cone, which means that the peak of the cone is located at $(t, F(\mathbf{e}) + \Phi_{\mathbf{e}}\varepsilon_1)$.

Note that, even if the authors were focusing on asymptotic results, the bound given in Parambath et al. (2014) can also be interpreted geometrically.

For the sake of clarity, we restate the Proposition 5 of Parambath et al. (2014) for our purpose:

Proposition. *Let* $t, t' \in [0, 1]$ *and* $\varepsilon_1 \geq 0$. *Suppose that there is* $\Phi > 0$ *such that for all*

$\mathbf{e}, \mathbf{e}' \in \mathcal{E}(\mathcal{H})$ *satisfying* $F(\mathbf{e}') > F(\mathbf{e})$*, we have:*

$$F(\mathbf{e}') - F(\mathbf{e}) \geq \Phi \langle \mathbf{a}(t'), \mathbf{e} - \mathbf{e}' \rangle. \tag{5.9}$$

Furthermore, suppose that we have the two following conditions

$$(i) \;\; \|\mathbf{a}(t) - \mathbf{a}(t')\|_2 \leq 2 \left| t - t' \right|, \qquad (ii) \;\; \langle \mathbf{a}(t), \mathbf{e} \rangle \leq \min_{\mathbf{e}'' \in \mathcal{E}(\mathcal{H})} \langle \mathbf{a}(t), \mathbf{e}'' \rangle + \varepsilon_1.$$

Let us also set $M = \max_{\mathbf{e}'' \in \mathcal{E}(\mathcal{H})} \|\mathbf{e}''\|_2$*, then we have:*

$$F(\mathbf{e}') \leq F(\mathbf{e}) + \Phi \varepsilon_1 + 4M\Phi \left| t' - t \right|.$$

According to the authors, the point (i) is a consequence of $\mathbf{a}$ of being Lipschitz continuous with Lipschtiz constant equal to 2. The point (ii) is just the expression of the sub-optimality of the learned classifier.

Proof. For all $\mathbf{e}, \tilde{\mathbf{e}} \in \mathcal{E}(\mathcal{H})$ and $t, t' \in [0, 1]$, we have:

$$\begin{aligned} \langle \mathbf{a}(t), \tilde{\mathbf{e}} \rangle &= \underbrace{\langle \mathbf{a}(t) - \mathbf{a}(t'), \tilde{\mathbf{e}} \rangle} + \langle \mathbf{a}(t'), \tilde{\mathbf{e}} \rangle, \\ &\leq \langle \mathbf{a}(t'), \tilde{\mathbf{e}} \rangle + 2M \left| t' - t \right|, \end{aligned}$$

where we have successively applied the Cauchy-Schwarz inequality and (i). Then:

$$\min_{\mathbf{e}'' \in \mathcal{E}(\mathcal{H})} \langle \mathbf{a}(t), \mathbf{e}'' \rangle \leq \min_{\mathbf{e}'' \in \mathcal{E}(\mathcal{H})} \langle \mathbf{a}(t'), \mathbf{e}'' \rangle + 2M \left| t' - t \right| = \langle \mathbf{a}(t'), \mathbf{e}' \rangle + 2M \left| t' - t \right|, \tag{5.10}$$

where $\mathbf{e}'$ denotes the error profile learned by the optimal classifier trained with the cost function $\mathbf{a}(t')$ and is such that $F(\mathbf{e}') > F(\mathbf{e})$. Then, writing $\langle \mathbf{a}(t'), \mathbf{e} \rangle = \langle \mathbf{a}(t') - \mathbf{a}(t), \mathbf{e} \rangle + \langle \mathbf{a}(t), \mathbf{e} \rangle$ and applying the Cauchy-Schwarz inequality, we have:

$$\begin{aligned} \langle \mathbf{a}(t'), \mathbf{e} \rangle &\leq \underbrace{\langle \mathbf{a}(t), \mathbf{e} \rangle} + 2M \left| t' - t \right|, \\ &\qquad \downarrow \text{ using the fact that the learned classifier is sub-optimal} \\ &\leq \min_{\mathbf{e}'' \in \mathcal{E}(\mathcal{H})} \langle \mathbf{a}(t), \mathbf{e}'' \rangle + \varepsilon_1 + 2M \left| t' - t \right|, \\ &\leq \langle \mathbf{a}(t'), \mathbf{e}' \rangle + \varepsilon_1 + 4M \left| t' - t \right|, \end{aligned}$$

where the second inequality comes from (ii) and the last inequality comes from Equation (5.10). By plugging this last inequality in Inequality (5.9), we get the result.

Furthermore, the existence of the constant Φ has been proved by the authors and is equal to $(\beta^2 P)^{-1}$. □

This bound also defines a cone which is, this time, symmetric with a slope equal to $4\Phi M$, as illustrated in Figure 5.3 (right). Using real datasets, we will compare, in Section 5.4.2, the cones produced by this bound and ours.

5.3.2 CONE: A bound-inspired algorithm

We now leverage the previous geometric interpretation to design **CONE**, an iterative algorithm that wraps a cost-sensitive classification algorithm (e.g., a weighted SVM). The pseudo-code of **CONE** is described in Algorithm 3 and an illustration is given in Figure 5.4. At every iteration i, **CONE** proposes a new value t_i (with $findNext$) to be used by the cost-sensitive algorithm.

Algorithm 4: CONE

Input: training set S,
Input: weighted-learning algorithm $wLearn$,
Input: stopping criterion $shouldStop$.

Output: t, $classifier$ and F.

Initialize $T_{val} = \{0, 1\}$, $\mathcal{Z}_0 = \varnothing$ and $i = 1$.
repeat
  $t_i = findNextT(\mathcal{Z}_{i-1},\ T_{val})$
  $classifier_i = wLearn(1 + \beta^2 - t_i,\ t_i,\ S)$
  $F_i = F_\beta(classifier_i,\ S)$
  $\mathcal{V}_i = unreachableZone(t_i,\ F_i,\ S,\ classifier_i)$
  $\mathcal{Z}_i = \mathcal{Z}_{i-1} \cup \mathcal{V}_i$
  $T_{val} = T_{val} \cup \{t_i\}$
  $i = i + 1$
until $shouldStop(i,\ classifier_i,\ \mathcal{Z}_i,\ T_{val})$

The choice of t_i is based on the area $\mathcal{Z}_{i-1}$ which we define as the union of all cones obtained from previous iterations. t_i is chosen to reduce the maximum value of F for which (t, F) is not in any previous cone. To achieve this goal, **CONE** keeps track of a list T_{val}, initialized with the values 0 and 1, and enriched at each iteration with the values of t that have been considered. The selection of t_i is done as follows: (i) search the value t_{opt} which maximizes $F_{max}(t) = \max\{F, (t, F) \notin \mathcal{Z}_{i-1}\}$, (ii) search for the greatest value t_l in T_{val} such that $t_l < t_{opt}$ and the smallest value t_r such that $t_{opt} < t_r$. (iii) take the middle of the interval $[t_l, t_r]$ as the output, i.e. $t_i = \frac{1}{2}(t_l + t_r)$.

The cost sensitive classification algorithm then provides a new value of F_i obtained from the cost t_i and $classifier_i$, which is used to refine the unreachable area as $\mathcal{Z}_i = \mathcal{Z}_{i-1} \cup \mathcal{V}_i$, where $\mathcal{V}_i$ is the cone corresponding to (t_i, F_i). In the case where there are multiple values of t that maximize $F_{max}(t)$ (e.g., at the beginning, or when some range of t values yield $F = 1$), **CONE** selects as t_{opt} the middle of the widest range at the first stage (i) (see the white dotted lines in Figure 5.4).

From a practical perspective, $\mathcal{Z}_i$ can be represented as a combination of linear constraints or as a very dense grid of binary values (a rasterization of $[0, 1] \times [0, 1]$, the (t, F) space).

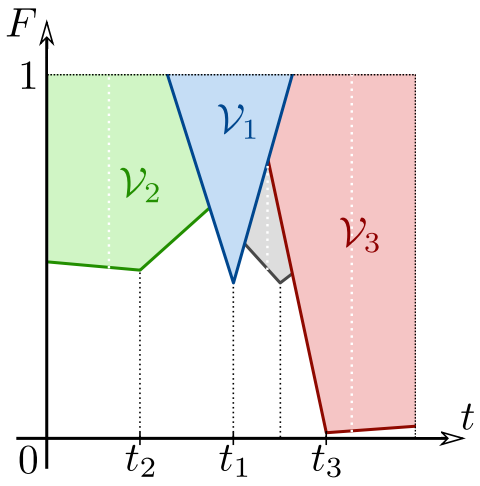

Figure 5.4: Illustration of the **CONE** algorithm in the middle of its fourth iteration. The colored areas represent the unreachable regions in the (t, F)-space.

Both approaches can be made efficient (and negligible compared to $wLearn$). The stopping criterion *shouldStop* can take different forms including a fixed number of iterations, a fixed time budget, or some rules on the current best F-measure and the current upper bound $max_t\, F_{max}(t)$. While **CONE** selects a single next value of t, it can easily be generalized to produce multiple values of t to consider in parallel (to exploit parallel computing of multiple instances of $wLearn$).

By always selecting a t_i that is in the middle of two previously tested t-values, **CONE** performs a progressive refinement of a grid. We can (and do, in practice) restrict the values of t in the (t, F)-space that the algorithm considers. More precisely, we can limit the depth of the progressive refinement to an integer value k. In this case, **CONE** will do at most $2^k - 1$ iterations, in order to cover all possible values on a grid with stride $\frac{1}{2^k}$. However, as the procedure is driven by the theoretical bounds. We will see in Section 5.4.3 that **CONE** finds good models in its very first iterations.

5.4 Experiments

In this section, we perform an experimental comparison of our bounds with the results obtained in Parambath et al. (2014) and study the behavior of the **CONE** algorithm on several datasets.

5.4.1 Datasets and experimental settings

The Table 5.1 describes the datasets we used for our experiments, with their corresponding Imbalance Ratio (I.R.). The higher this ratio, the more one should expect that optimizing the classification accuracy is a bad choice in terms of trade-off between precision and recall.

Dataset	Instances	Classes	I.R.	Features
Adult	48 842	2	3.19	123
Abalone10	4 174	2	5.64	10
SatImage	6 400	2	9.3	36
IJCNN'01	141 691	2	9.39	22
Abalone12	4 174	2	15.18	10
PageBlocks	5 500	2	22.7	10
Yeast4	1 484	2	28.1	8
Wine4	1 599	2	29.2	11
Letter	20 000	26	1.32	16
News20	19 928	20	1.12	62061
BLITZ	2 690 414	2	117.1	17

Table 5.1: Datasets details. The Imbalance Ratio (I.R.) is the ratio between negative and positive instances (or between sizes of the largest and smallest classes, in a multiclass setting).

The datasets *IJCNN'01* and *News20* are obtained from LIBSVM[2]. The other ones come from the UCI repository[3].

We reproduce the experimental setting from Parambath et al. (2014) which we describe here. For datasets with no explicit test set, $\frac{1}{4}$ of the data is kept for testing. The training set is split at random, keeping $\frac{1}{3}$ as the validation set, used to select the hyper-parameters using the F_1-measure. The penalty constraint of the SVM classifiers (hyper-parameter C) is considered in $\{2^{-6}, 2^{-5}, ..., 2^6\}$. In the experiments, t is taken in $[0, 1]$ as t belongs in the image space of the F-measure. Thus the class weights $\mathbf{a}(t)$ belongs to $[0, 1+\beta^2]$. The maximal number of training iterations is set to 50000. Fitting the intercept of the classifiers is achieved by adding a constant feature with value 1. We report the test-time F_1-measure averaged over 5 experiments.

We consider two different base cost-sensitive classification algorithms (both implementations use LIBLINEAR): linear SVM and Logistic Regression (LR) for a fair comparison with Koyejo et al. (2014). We report the performance of 5 different approaches:

- using a single standard classification algorithm with hyper-parameters tuned on the F-measure,
- an additional baseline (with the $_{I.R.}$ subscript), which consists in using a cost that rebalances the classes (the cost c of a false negative is the proportion of positive examples in the dataset and the cost of a false positive is $1 - c$),
- the **Grid** wrapper proposed in Parambath et al. (2014) that regularly splits the interval

[2] `https://www.csie.ntu.edu.tw/~cjlin/libsvm/`
[3] `https://archive.ics.uci.edu/ml/datasets.html`

$[0, 1]$ of t values,

- the algorithm **CONE** derived from our theoretical study,
- algorithm 2 from Narasimhan et al. (2015b) based on the bisection method.

About ε_1. The value of ε_1 (in all presented bounds) represents the $\mathbf{a}(t)$-weighted sub-optimality of the classifier, compared to the best one from the hypothesis class. This sub-optimality cannot be computed efficiently as it would require a learning algorithm that produces optimal classifiers in terms of $\mathbf{a}(t)$-weighted error. Our goal is not on estimating ε_1, we then set $\varepsilon_1 = 0$ which is computationnaly free, and shown by the experiment to be a reasonable choice both in terms of bound analysis (the bound is most of the time respected) and in terms of overall results from the **CONE** algorithm.

5.4.2 Evaluation of the tightness of the bound

In this section, we aim at illustrating and showing the tightness of our bounds. To do so, we consider the (t, F) values obtained by 19 weighted-SVM learned on a regular grid of t values. For these same 19 models, we consider the cones obtained from our bounds and from Parambath et al. (2014). We do not provide figures for all the datasets but only for two. We illustrate the tightness of our bounds on two datasets: *Adult* and *Abalone12*, that have been chosen because they are different in terms of number of instances, number of features and imbalance ratios.

Impact of ε_1. Both our bounds and the one from Parambath et al. (2014) are impacted by ε_1 which shows up as an offset, multiplied by $\Phi_{\mathbf{e}}$ for our bounds, and by Φ in the other one. As $\Phi_{\mathbf{e}} \leq \Phi$, our bounds are less impacted by an increased ε_1. With the 19-SVM setting, Figure 5.5 shows the evolution of the maximum still-achievable F-measure depending on the value of ε_1, with a hard maximum at 1. The values of ε_1 are expressed in number of points for an easier interpretation.

The bound from Parambath et al. (2014) gives loose guarantees and the aggregate bound is most of the time above 1. The values can, for example, start at $F = 1$ and end up at $F = 3$ on the *Abalone12* dataset. This representation shows once again that our bounds are very tight.

Visualizing unreachable zones. The grayed-out areas in Figure 5.6 are the unreachable zones. For both methods, this figure shows that the guarantees obtained with our bounds are much more relevant than the ones from Parambath et al. (2014). Our bounds give unreachable zones that go very close to the empirical points.

Looking at the cones with our tight bounds, we see that sometimes a point is in the cone generated by another point. This looks like a violation of our bounds but it rather shows that ε_1 cannot be considered to be 0 in the current setting. Naturally, $\varepsilon_1 \neq 0$ comes from the

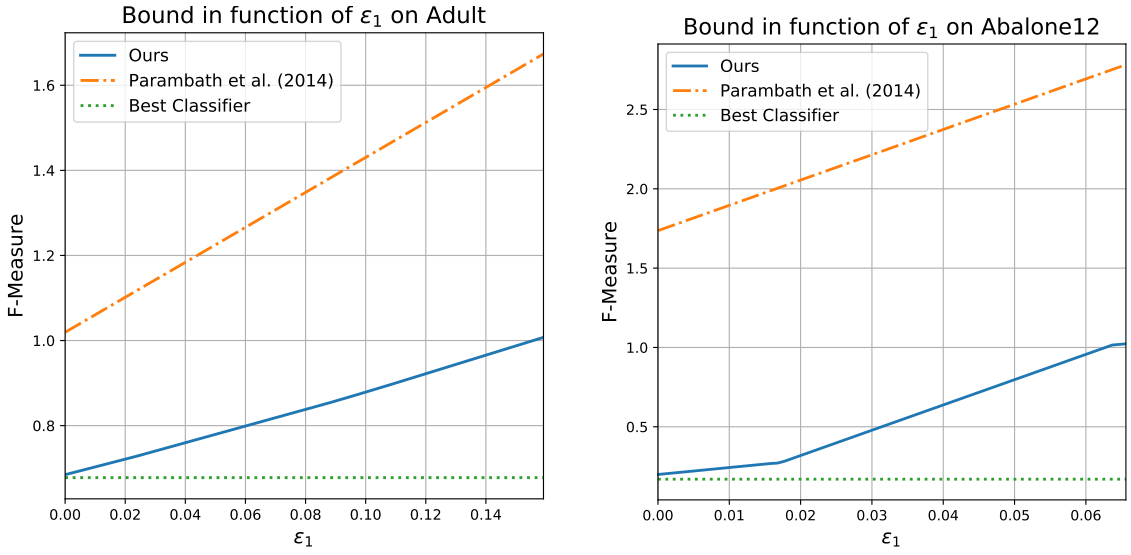

Figure 5.5: Bounds on the F-measure as a function of ε_1, the unknown sub-optimality of the SVM learning algorithm. Results are shown on two datasets: Adult (left) and Abalone12 (right). The **Grid** algorithm refers to the method presented in Parambath et al. (2014).

fact that the weighted-SVM is not robust and not optimal in terms of weighted 0-1 loss. Our intuition is that the SVM is less and less optimal as the weights become more extreme, such as when t gets closer to 0.

Bounds' evolution across iterations. We now study how the training performance and the overall bounds evolve as we add more models. In **CONE**, adding a model means doing one more iteration, while with the grid approach of Parambath et al. (2014) it requires to re-learn all models (as all grid locations change). Figure 5.7 illustrates that **CONE** tends to produce better models at a lower cost. This figure also outlines the fact that our upper bound is tight and goes down quickly as we add models.

5.4.3 Performance in F-measure at test time

Finally, we compare the performance of **CONE** (SVM_C), based on the SVM algorithm against its competitors: LR_B for the method of Narasimhan et al. (2015b), $\text{LR}_{I.R.}$ and LR^T for Koyejo et al. (2014) and SVM_G/LR_G for the method of Parambath et al. (2014). We present the results of all methods in Table 5.4.3, still giving a budget of 19 models for **CONE** and **Grid** based method. Overall, **CONE** performs at least as well as its competitors in average, and the very best results are obtained by combining **CONE** with thresholding.

The baseline of using a simple SVM completely fails on half of the datasets. The improved SVM which consists in rebalancing the classes ($\text{SVM}_{I.R.}$) still performs worse than the other approaches on average, and on most datasets. Even with thresholding, the approaches that learn a single model (LR^T and $\text{LR}^T_{I.R.}$) are still outperformed by the ones that learn multiple

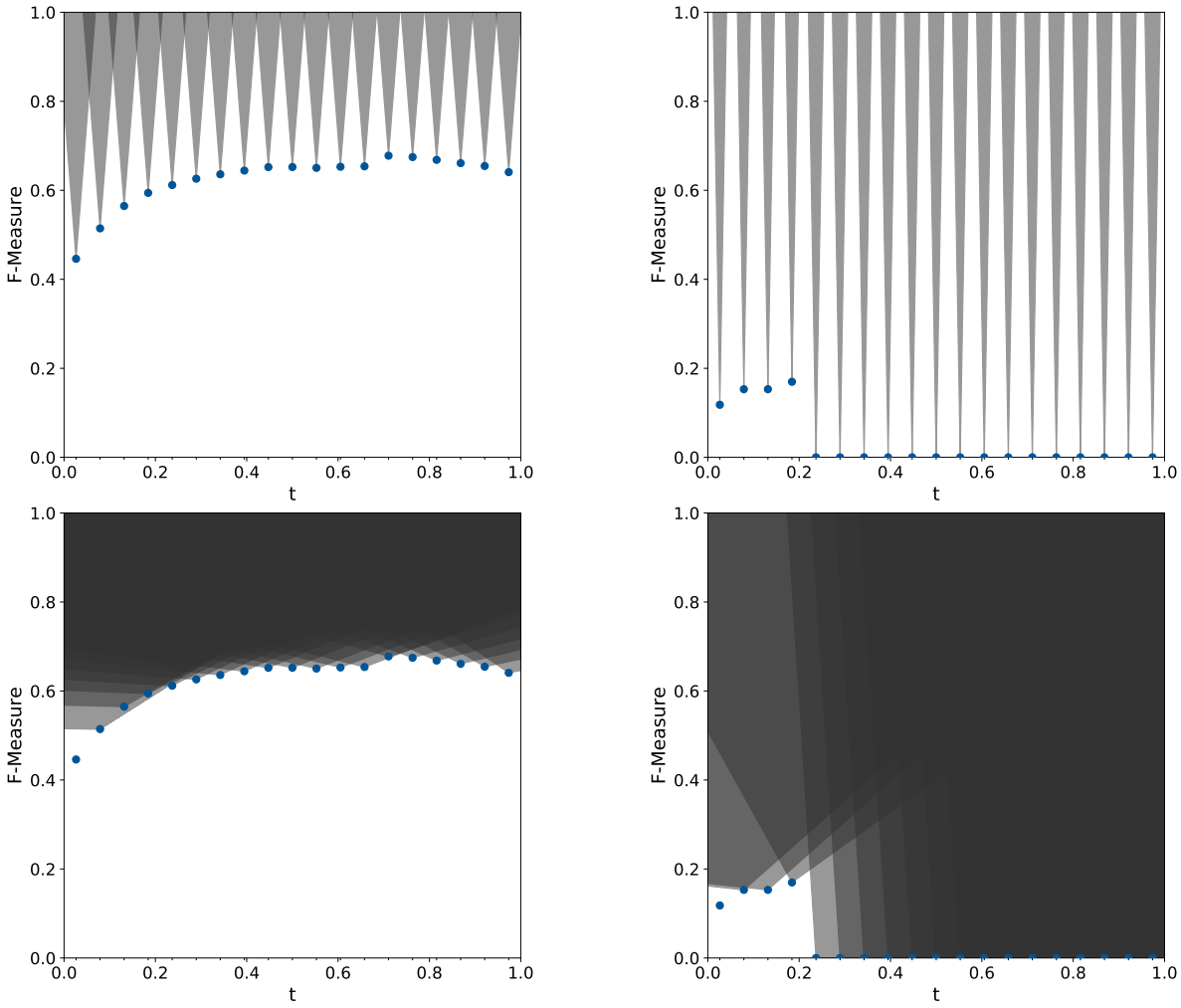

Figure 5.6: Unreachable region obtained from the same 19 (t, F) points corresponding to learning weighted-SVMs on a grid of t values. Cones are shown for the Adult (left) and Abalone12 (right) datasets, and with the bound from Parambath et al. (2014) (top) and with our tighter bounds (bottom).

models with different class-weights, like ours (all subscripts $_C$) and the grid one (subscripts $_G$). This last result shows that it is insufficient to solely rely on tuning the threshold of a single model. In average, both **CONE** and the grid approach outperform all other considered methods, including the bisection algorithm LR_B. We see that the results of **CONE** are very similar to the grid approach SVM_G. However, looking at Figure 5.8 at the bottom, we see that the proposed method is able to reach higher values with a limited number of iterations, i.e. after training fewer models.

5.5 Conclusion

In this chapter, we have presented new bounds on the F-measure based on a cost-sensitive classification approach. These bounds have been shown to be tighter than existing ones and less sensitive to the sub-optimality of the learned classifier (ε_1). Furthermore, we have shown that our bounds are useful from a practical point of view by deriving **CONE**, an algorithm which iteratively selects class weights to reduce the overall upper bound on the optimal F-measure. **CONE** has been shown to perform at least as well as its competitors on various datasets.

The presented work can be extended to any fractional-linear functions such as Jaccard index. Indeed, the development presented in Section 5.2.3 is F-measure independent and can lead to a similar proposition as Proposition 5.3 but the values of both $\Phi_{\mathbf{e}}, M_{min}$ and M_{max} depend on the performance measure.

Finally, note that we only focus in this work on linear classifiers such as SVMs or Logistic Regression. A first perspective would be the refinement of our theoretical framework for exploring more efficiently the search space or extensions to more difficult contexts such as SGD-based algorithms used with neural networks. We also aim to estimate, experimentally and/or theoretically, the sub-optimality ε_1 of the classifier using the work from Bousquet et al. (2004) for several algorithms. Indeed, the sub-optimality of the classifier is directly linked to the value of t and, in our **CONE** algorithm, this would lead to a vertical translation of the "cone". Thus it will modify the way the different values of t are explored.

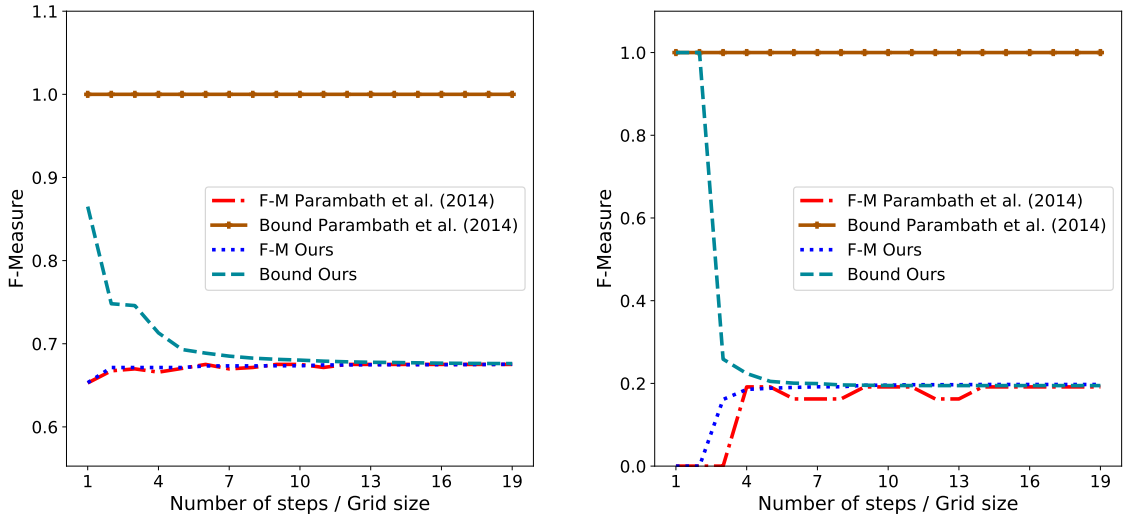

Figure 5.7: Training performance of **CONE** versus the grid approach from Parambath et al. (2014), together with their respective bounds. Results are shown on two datasets: Adult (left) and Abalone12 (right). We suppose $\varepsilon_1 = 0$, which explains that we observe empirical values that are higher than our upper bound (on Abalone12).

Dataset	SVM	$\text{SVM}_{I.R.}$	SVM_G	SVM_C	SVM_C^T	LR^T	$\text{LR}_{I.R.}^T$	LR_G^T	LR_B
Adult	62.5 (0.2)	64.9 (0.3)	66.4 (0.1)	66.5 (0.1)	66.4 (0.1)	66.5 (0.1)	66.5 (0.1)	66.5 (0.1)	66.6 (0.1)
Abalone10	0.0 (0.0)	30.9 (1.2)	32.4 (1.3)	32.2 (0.8)	31.8 (1.9)	30.8 (2.2)	30.7 (1.9)	30.7 (1.9)	31.6 (0.6)
Satimage	0.0 (0.0)	23.4 (4.3)	20.4 (5.3)	20.6 (5.6)	30.9 (2.0)	21.2 (11.1)	28.6 (1.9)	28.6 (1.9)	21.4 (4.6)
IJCNN	44.5 (0.4)	53.3 (0.4)	61.6 (0.6)	61.6 (0.6)	62.6 (0.4)	59.4 (0.5)	56.5 (0.3)	56.5 (0.3)	59.2 (0.3)
Abalone12	0.0 (0.0)	16.8 (2.7)	16.8 (4.2)	18.3 (3.3)	16.3 (3.0)	15.5 (3.1)	17.0 (3.3)	17.0 (3.3)	17.7 (3.7)
Pageblocks	48.1 (5.8)	39.6 (4.7)	66.4 (3.2)	62.8 (3.9)	67.6 (4.0)	59.2 (8.1)	55.9 (6.4)	55.9 (6.4)	55.7 (5.7)
Yeast	0.0 (0.0)	29.4 (2.9)	38.6 (7.1)	39.0 (7.5)	35.4 (15.6)	37.4 (10.1)	39.9 (6.5)	27.6 (6.8)	27.6 (6.8)
Wine	0.0 (0.0)	15.6 (5.2)	20.0 (6.4)	22.7 (6.0)	19.3 (7.9)	21.5 (3.7)	25.2 (4.5)	25.2 (4.5)	18.3 (7.2)
Letter	75.4 (0.7)	74.9 (0.8)	80.8 (0.5)	81.0 (0.3)	81.0 (0.4)	82.9 (0.3)	82.9 (0.3)	82.9 (0.3)	74.9 (0.5)
News20	90.9 (0.1)	91.0 (0.2)	91.1 (0.1)	91.0 (0.1)	91.0 (0.1)	90.6 (0.1)	90.6 (0.1)	90.6 (0.1)	89.4 (0.2)
Average	32.1 (0.7)	44.0 (2.3)	49.5 (2.9)	49.6 (2.8)	50.4 (3.0)	48.8 (1.0)	48.2 (2.3)	49.1 (3.6)	47.0 (3.9)
BLITZ	0.0 (0.0)	0.0 (0.0)	7.5 (0.2)	8.9 (0.1)	4.4 (2.2)	2.4 (0.5)	2.4 (0.5)	5.0 (2.2)	7.3 (0.3)

Table 5.2: Classification F-Measures for $\beta = 1$ with SVM and Logistic Regression algorithms. SVM_G and LR_G^T are reproduced experiments of Parambath et al. (2014) and the subscript $_{I.R.}$ is used for the classifiers trained with a cost depending on the Imbalance Ratio. The subscript $_B$ corresponds to the bisection algorithm presented by Narasimhan et al. (2015b). LR^T and $\text{LR}_{I.R.}^T$ are reproduced experiments of Koyejo et al. (2014). Finally the $_C$ stands for our wrapper **CONE** and SVM_C^T designed as a combination using the **CONE** + threshold. Reported F-measure values are averaged over 5 experiments (standard deviation between brackets).

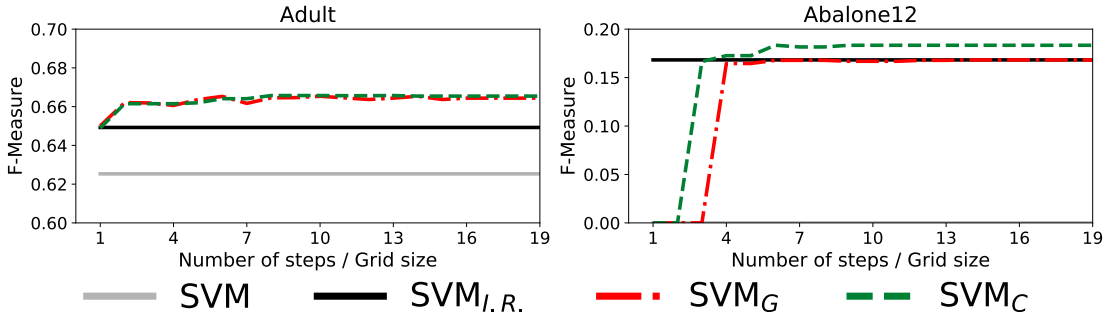

Figure 5.8: F-measure obtained on the test set for four considered approaches on Adult (top) and Abalone12 (bottom) datasets, plotted as a function of the computing budget (number of weighted SVM to learn).

Chapter 6

Tree Based Cost-Sensitive Learning

This chapter is based on the following publication

Guillaume Metzler, Xavier Badiche, Brahim Belkasmi, Elisa Fromont, Amaury Habrard, and Marc Sebban. Tree-based cost sensitive methods for fraud detection. In *International Symposium on Intelligent Data Analysis(IDA- 2018)*, pages 213–224. Springer International Publishing, 2018b

---

Abstract

In bank fraud detection, the impact of each case depends on the amount of money involved in the transaction. In this chapter, we aim to improve fraud detection model used by the Blitz Company currently based on random forests. We first propose a cost-sensitive splitting criterion for decision trees that takes into account the cost of each transaction and we extend it with a decision rule for classification with tree ensembles. We then propose a new cost-sensitive loss to train a gradient boosting model. Both methods are shown to be particularly relevant in the context of imbalanced data. The experiments are conducted on a Blitz dataset and we show that, using a cost sensitive approach, we are able to increase the retailer's benefits up to 1,43% compared to the current used model.

6.1 Introduction

The current model used by the Blitz company for fraud detection is a tree-based model where several non cost-sensitive trees are built and combined in order to predict if a new transaction is a fraud or not. Due to the fact that the model does not directly take into account some information on the amount of money involved in transaction, it fails to give more importance to expensive transactions. However, for Blitz's customers, all the frauds cannot be considered as equal: refusing a genuine transaction with a small amount of money does not have the same consequences as refusing a more costly one.

Inspired by the literature on imbalanced data (see Chapter 2), we propose a cost-sensitive method to take into account the amount of each transaction to improve the retailers' profits.

	Predicted Positive (fraud)	Predicted Negative (genuine)
Actual Positive (fraud)	c_{TP_i}	c_{FN_i}
Actual Negative (genuine)	c_{FP_i}	c_{TN_i}

Table 6.1: Cost Matrix associated to each example of the training set.

Starting from the currently used tree-based model, we modify the splitting criterion of the learning algorithm using a cost matrix. In a second part, we present a gradient boosting model which aims to optimize a surrogate of our loss function based on a given cost-matrix.

6.2 Problem Formulation

6.2.1 Notations

As in the rest of this document, we focus in this chapter on binary supervised classification (fraud vs. non fraud). Let $S = ((\mathbf{x}_1, y_1), ...(\mathbf{x}_m, y_m))$ be a set of m training instances where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{0, 1\}$ are their corresponding labels. The notation x_i^j is used to denote the value of the j^{th} variable of the instance i. The label 0 will be used for the negative or majority class (i.e. the genuine transactions) and the label 1 for the positive or rare class (i.e. the frauds). Let us remind that we denote by S_+ the set of m_+ positive examples and S_- the set of m_- negative ones (here $m_- >> m_+$). We will also note $\hat{y}_i$ the label predicted by our learned model for the instance i. We use the notation p for the predicted probability that an example belongs to the minority class. F is the output of the learned model, such that. $p_i = F(\mathbf{x}_i)$ is the probability that the transaction $\mathbf{x}_i$ is fraudulent. A threshold is then used to predict the final label.

6.2.2 Cost Sensitive Model and Loss Function

Our goal is to maximize the profits of the retailers by predicting, with decision trees, which transactions are genuine or not. While training the trees, the company might like to introduce some costs assigned to the training examples, according to the adequacy between the actual label of the transaction and the predicted one (see Table 6.1). For instance, the retailers will gain money by accepting a genuine transaction, i.e. $c_{TN_i} > 0$, where TN stands for *True Negative* or genuine transactions correctly classified. However, if the retailers accept a fraudulent one, they will loose the amount of the transaction $c_{FN_i} < 0$, where FN stands for *False Negative* or fraudulent transaction predicted as a genuine one.

In this chapter, we use a similar cost-sensitive approach as the one presented in Correa Bahnsen et al. (2017). However, instead of only minimizing the money loss due to the acceptation of a fraud, we rather focus on maximizing the retailers profits, i.e. we aim at

maximizing the loss function L defined as follows:

$$L(y \mid \hat{y}) = \sum_{i=1}^{m} \left[y_i(\hat{y}_i c_{TP_i} + (1-\hat{y}_i)c_{FN_i}) + (1-y_i)(\hat{y}_i c_{FP_i} + (1-\hat{y}_i)c_{TN_i}) \right]. \tag{6.1}$$

In the following, we use the word "profit" instead of "cost" because it is more meaningful for the retailers.

We show how this loss function can be optimized while learning decision trees.

6.3 Cost Sensitive Decision Trees

We recall that classification trees (see Chapter 1 for more details) usually split the nodes according to an "impurity" measure. One such measure is the Gini index of a set of m instances $(\mathbf{x}_i, y_i)$ defined as follows: $Gini = 1 - \sum_{k=1}^{C} p_k^2$, where p_k denotes the probability to belong to the class k and C is the number of classes ($C = 2$ in our case). In this chapter, the splitting criterion is based on the cost matrix defined in Table 6.1. We do not want to minimize an impurity but to *maximize the retailer profits* according to the cost matrix.

6.3.1 Splitting Criterion and Label Assignment

Our splitting criterion Γ_S on a given set of training instances S of size m is:

$$\Gamma_S = \frac{1}{m} \left[\sum_{i \in S_-} \left(\frac{m_+}{m} c_{FP_i}(\boldsymbol{x}_i) + \frac{m_-}{m} c_{TN_i}(\boldsymbol{x}_i) \right) + \sum_{i \in S_+} \left(\frac{m_+}{m} c_{TP_i}(\boldsymbol{x}_i) + \frac{m_-}{m} c_{FN_i}(\boldsymbol{x}_i) \right) \right], \tag{6.2}$$

where the first term corresponds to the profits due to genuine transactions and the second to the frauds.

Note that this quantity depends on the transaction amount of each example in S through the costs c. The best attribute A is the one which *maximizes* the quantity:

$$\sum_{v \in \text{Children}(A)} \Gamma_{S_v} - \Gamma_S.$$

Note that this quantity is very similar to the common splitting criterion used to minimize the Gini impurity up to the number of examples in the parent node. We simply take the opposite of the classical gain and divide it by the number of instances in the parent node.

The values Γ_{S_v} are computed using Equation (6.2) on each set S_v. It differs from the splitting criterion used in Correa Bahnsen et al. (2017) where the splits minimize the cost of wrongly accepting or blocking the transactions.

Once the induction tree stopping criterion is reached (see Section 6.5), a class label is associated to each leaf of the tree. For the sake of clarity, we introduce the following notations:

- $\gamma_0(l)$: the average profit associated to the leaf l if all the instances are predicted as genuine:

$$\gamma_0(l) = \frac{1}{|l|}\left(\sum_{i:\mathbf{x}_i \in l \cap S_-} c_{TN_i} + \sum_{i:\mathbf{x}_i \in l \cap S_+} c_{FN_i}\right),$$

- $\gamma_1(t)$: the average profit associated to the leaf l if all instances are predicted as frauds:

$$\gamma_1(l) = \frac{1}{|l|}\left(\sum_{i:\mathbf{x}_i \in l \cap S_-} c_{FP_i} + \sum_{i:\mathbf{x}_i \in l \cap S_-} c_{TP_i}\right),$$

  where $|l|$ denotes the number of examples in the leaf l and $i : \mathbf{x}_i \in l \cap S$ denotes the index i of the example x_i both in leaf l and in the set S.

A leaf is assigned the label 1 if $\gamma_1 > \gamma_0$, i.e. all the transactions in a given leaf are predicted fraudulent if the associated average profit is greater than the one associated when all instances are predicted genuine.

Note that this strategy can be easily extended to ensembles of trees (Breiman, 2001). In this case, a standard decision rule consists in applying a majority vote over the whole set of the T learned decision trees. However, this decision rule does not take into account the probability score that can be associated to each tree prediction using the class distribution of the examples in the leaf $l(\mathbf{x}_i)$ (as in Sahin et al. (2013)). Following this idea, we suggest here to label an instance as positive if the average $\bar{\gamma}_1(\mathbf{x})$ of the average profits $\gamma_1(l^j(\mathbf{x}_i))$ over the T trees is greater than $\bar{\gamma}_0(\mathbf{x})$, where $l^j(\mathbf{x}_i)$ is the leaf of the j^{th} tree containing $\mathbf{x}_i$:

$$\bar{\gamma}_1(\boldsymbol{x}) = \frac{1}{T}\sum_{j=1}^{T}\gamma_1(l^j(\mathbf{x})) \geq \frac{1}{T}\sum_{j=1}^{T}\gamma_0(l^j(\mathbf{x})) = \bar{\gamma}_0(\mathbf{x}).$$

6.4 Cost Sensitive Gradient Boosting

We present here a proper cost-sensitive loss function in order to implement it in a gradient boosting algorithm in an efficient way. Let us first present *XGBoost* (Chen and Guestrin, 2016)

6.4.1 An Introduction to XGBoost

XGBoost. is a tree gradient boosting algorithms (defined in Chapter 2). We first explain how the weights are computed for each leaf, then we explain the splitting criterion that is used.

Computing the optimal weight of a leaf

We consider a loss L and the following optimization problem:

$$\min \sum_{i=1}^{m} L(y_i, \hat{y}_i) + \beta\mathcal{L} + \frac{\lambda}{2}\sum_{j=1}^{\mathcal{L}}(f_j^{(t)})^2, \tag{6.3}$$

where $\beta\mathcal{L}$ and $\frac{\lambda}{2}\sum_{j=1}^{\mathcal{L}}(f_j^{(t)})^2$ are two regularization terms used to control the number of leaves and the weight of each leaf $f_j^{(t)}$ for the learned tree.

We recall that the models are learned in an additive manner, so let us denote $\hat{y}^{(t-1)}$, the predicted value by the first $t-1$ functions f_k, i.e. $\hat{y}_i^{(t-1)} = \sum_{k=1}^{t-1} f^{(k)}(\mathbf{x}_i) = F^{(t-1)}(\mathbf{x}_i)$. Let us now study how the next model is learned. For this purpose, we rewrite the quantity (6.3) to minimize as follows:

$$\sum_{i=1}^{m} L(y_i, \hat{y}_i^{(t-1)} + f^{(t)}(\mathbf{x}_i)) + \beta\mathcal{L} + \frac{\lambda}{2}\sum_{j=1}^{\mathcal{L}}(f_j^{(t)})^2. \tag{6.4}$$

In practice, Chen and Guestrin (2016) only consider a second order approximation of the function they aim to optimize. This second order approximation is done with respect to the predicted value at the previous iteration, i.e. $\hat{y}_i^{(t-1)}$. We will denote by respectively g and h the first and second order derivatives of the function L with respect to $\hat{y}^{(t-1)}$. We can rewrite (6.4) as follows:

$$\sum_{i=1}^{m}\left[L(y_i, \hat{y}_i^{(t-1)}) + f_t(\mathbf{x}_i)g(\mathbf{x}_i) + \frac{1}{2}f_t^2(\mathbf{x}_i)h(\mathbf{x}_i)\right] + \beta\mathcal{L} + \frac{\lambda}{2}\sum_{j=1}^{\mathcal{L}}(f_j^{(t)})^2. \tag{6.5}$$

Remember that we aim to learn the function $f^{(t)} = (f_j^{(t)})_{j=1,\dots,\mathcal{L}}$. So let us consider a leaf j and denote by I_j the set of index i such that $\mathbf{x}_i$ falls in the leaf l_j. Thus, using (6.5), the function $f_j^{(t)}$ shall minimize the following quantity V:

$$V = \sum_{i\in I_j}\left[g(\mathbf{x}_i)f_f^{(t)}(\mathbf{x}_i) + \frac{1}{2}\left(\lambda + h(\mathbf{x}_i)\right)\left(f_f^{(t)}(\mathbf{x}_i)\right)^2\right]. \tag{6.6}$$

This function is convex and the minimum is given by the solution of *Euler's equation*, i.e. the function $f^{(t)}$ for which the gradient vanishes. This solution is given by:

$$f_j^{(t)} = -\frac{\sum_{i\in I_j} g(\mathbf{x}_i)}{\sum_{i\in I_j} h(\mathbf{x}_i) + \lambda}. \tag{6.7}$$

The splitting criterion

Once the optimal weight is found for each leaf (6.7), we can compute the optimal value $V^\star$ of the loss by using (6.6), we get:

$$V^\star = \sum_{i\in I_j}\left[\underbrace{-g(\mathbf{x}_i)\frac{\sum_{i\in I_j} g(\mathbf{x}_i)}{\sum_{i\in I_j} h(\mathbf{x}_i)+\lambda}}+\underbrace{\frac{1}{2}\left(\lambda+h(\mathbf{x}_i)\right)\left(-\frac{\sum_{i\in I_j} g(\mathbf{x}_i)}{\sum_{i\in I_j} h(\mathbf{x}_i)+\lambda}\right)^2}\right],$$

$$= -\frac{\left(\sum_{i\in I_j} g(\mathbf{x}_i)\right)^2}{\sum_{i\in I_j} h(\mathbf{x}_i)+\lambda}+\frac{1}{2}\frac{\left(\sum_{i\in I_j} g(\mathbf{x}_i)\right)^2}{\sum_{i\in I_j} h(\mathbf{x}_i)+\lambda},$$

$$V^\star = -\frac{1}{2}\frac{\left(\sum_{i\in I_j} g(\mathbf{x}_i)\right)^2}{\sum_{i\in I_j} h(\mathbf{x}_i)+\lambda}.$$

This formula is used to measure the quality of a leaf. It can be seen as a generalized formula for Gini Impurity for any loss function. Using this new measure, they define their splitting criterion, i.e. the gain associated to a split, as follows:

$$\frac{1}{2}\left[\frac{\left(\sum_{i\in I_L} g(\mathbf{x}_i)\right)^2}{\sum_{i\in I_L} h(\mathbf{x}_i)+\lambda}+\frac{\left(\sum_{i\in I_R} g(\mathbf{x}_i)\right)^2}{\sum_{i\in I_R} h(\mathbf{x}_i)+\lambda}-\frac{\left(\sum_{i\in I} g(\mathbf{x}_i)\right)^2}{\sum_{i\in I} h(\mathbf{x}_i)+\lambda}\right]-\beta,$$

where $I = I_L \cup I_R$ for a binary tree and the parameter β is used to control the number of leaves.

6.4.2 Cost-Sensitive Loss for Gradient Tree Boosting

In this section, we aim to use the framework presented in Buja et al. (2005) to give a proper formulation of our loss function (6.1) in the context of a boosting algorithm, using the gain matrix presented in Table 6.1.

Using a Bayes rule for classification (Elkan, 2001), an instance i is predicted fraudulent if $\gamma_1 > \gamma_0$, i.e:

$$p_i c_{TP_i} + (1-p_i)c_{FP_i} - p_i c_{FN_i} - (1-p_i)c_{TN_i} > 0,$$

where p_i denotes the probability of the instance to be a genuine transaction. It gives us a threshold over which the transaction is rejected (or predicted fraudulent):

$$p_i > \frac{c_{TN_i} - c_{FP_i}}{c_{TP_i} - c_{FN_i} + c_{TN_i} - c_{FP_i}} = s_i.$$

Using the threshold s_i, our cost-weighted miss-classification loss can be rewritten as:

$$L(y \mid p) = -\frac{1}{m}\sum_{i=1}^{m}\left(y_i c_{TP_i} + (1-y_i)c_{FP_i}\right)\mathbb{1}_{p_i > s_i} + \left(y_i c_{FN_i} + (1-y_i)c_{TN_i}\right)\mathbb{1}_{p_i \le s_i}. \tag{6.8}$$

Then, following the framework presented in Buja et al. (2005), $L(y \mid p)$ can be rewritten as follows:

$$L(y \mid p) = \frac{1}{m}\sum_{i=1}^{m} \xi_i \left[y_i(1-s_i)\mathbb{1}_{p_i \leq s_i} + (1-y_i)s_i\mathbb{1}_{p_i > s_i} \right] - \frac{1}{m}\sum_{i=1}^{m}(y_i c_{TP_i} + (1-y_i)c_{TN_i}), \quad (6.9)$$

where $\mathbb{1}$ is an indicator function and where we use the fact that $s = s\mathbb{1}_{p>s} + s\mathbb{1}_{p\leq s}$ and set $\xi_i = c_{TN_i} - c_{FP_i} + c_{TP_i} - c_{FN_i}$ which is positive in our context. In fact, $c_{TN} > c_{FP}$: we earn more if we correctly classify a genuine transaction. Furthermore, if we accept a fraudulent transaction then we loose a part[1] of the transaction amount, otherwise we earn nothing, i.e. $0 = c_{TP} > c_{FN}$.

The first part of Equation (6.9), which we will note L_{s_i}, corresponds to the cost-sensitive loss introduced in Buja et al. (2005) with $s_i \in [0,1]$. Each term of the sum is multiplied by a constant ξ_i which depends on the data. The second term represents the maximum that our loss can reach if the predictions were perfect. Note that this second term does not depend on p_i. Therefore, we want to minimize:

$$\underset{p\in[0,1]}{\operatorname{argmin}}\ \mathbb{E}_y[L(y \mid p)] = \underset{p\in[0,1]}{\operatorname{argmin}}\ \mathbb{E}_y\left[\frac{1}{m}\sum_{i=1}^{m}\xi_i L_{s_i}(y_i \mid p_i)\right].$$

However, it has been shown that in the context of Boosting, it is more convenient to use an exponential approximation (Friedman, 2000). We adapt it to consider the output of a prediction model F directly in our approach as follows [2]:

$$\ell_{s_i} = (1-s_i)y_i e^{-F(\mathbf{x}_i)} + s_i(1-y_i)e^{F(\mathbf{x}_i)}.$$

Solving $\dfrac{\partial \mathbb{E}_{\mathbf{Y}}[\ell_{s_i}]}{\partial F(vx_i)} = 0$, we obtain the link function ψ_i between p_i, the probability of being a fraud and $\hat{F}_i = F(\mathbf{x}_i)$, the output of the model:

$$p_i = \psi_i(\hat{F}_i) = \frac{1}{1 + \dfrac{1-s_i}{s_i}e^{-2\hat{F}_i}},$$

and its inverse ψ_i^{-1} is given by:

$$e^{\hat{F}_i} = \left(\frac{1-s_i}{s_i}\right)^{1/2}\left(\frac{p_i}{1-p_i}\right)^{1/2}. \quad (6.10)$$

The way to transform the output of a boosting model into a probability (the calibration process) plays a key role in the performance of the predictive algorithm. It has been shown that we can achieve at least the same performance with a well calibrated boosting model as a

[1] By accepting a fraudulent transaction or a transaction from an account without funds, a store has the option of reselling the receivable (at a price lower than the amount of the check) to a private organization that will recover the amount of the transaction.

[2] Note that there exists a direct link between a predicted probability and the output of a model. See Section 3 of Friedman et al. (2000) and Section 4 of Buja et al. (2005) for further details.

cost-sensitive one (Nikolaou et al., 2016). However, the use of a cost-sensitive matrix is more flexible and meaningful for Blitz's customers.

It is worth noticing that we can use Equation (6.10) to provide a smooth approximation of the indicator function, such that:

$$\mathbb{1}_{p_i > s_i} \leq \left(\frac{1-s_i}{s_i}\right)^{1/2} \left(\frac{p_i}{1-p_i}\right)^{1/2} = e^{\hat{F}_i}.$$

Note that: $p_i > s_i \iff \psi_i(\hat{F}_i) > s_i \iff e^{\hat{F}_i} > 1 \iff \hat{F}_i > 0$. So it is enough to check the sign of the score to predict the label of each transaction. Finally, we are minimizing an upper bound $\tilde{L}$ of L:

$$L(y \mid p) \leq \tilde{L}(y \mid F) = \frac{1}{m} \sum_{i=1}^{m} \xi_i \left[(1-s_i) y_i e^{-\hat{F}_i} + s_i (1-y_i) e^{\hat{F}_i}\right].$$

When defining the optimal separation or the calculation of the optimal parameters of the model, we noticed, in the previous section, that only the first and second derivatives are necessary. We thus compute the first and second order derivatives of $\tilde{L}$ for each instance i with respect to $\hat{F}_i$. They are given by:

$$\frac{\partial \tilde{L}}{\partial \hat{F}_i} = \xi_i \left[-(1-s_i) y_i e^{-\hat{F}_i} + s_i (1-y_i) e^{\hat{F}_i}\right],$$

and

$$\frac{\partial^2 \tilde{L}}{\partial \hat{F}_i^2} = \xi_i \left[(1-s_i) y_i e^{-\hat{F}_i} + s_i (1-y_i) e^{\hat{F}_i}\right].$$

6.5 Experiments

In this section, we evaluate the decision rule presented in Section 6.3 and the loss function presented in Section 6.4. We compare the results of our method on the retailers' profits compared to a classic Random Forest (RF) algorithm based on the Gini impurity criterion (baseline).

6.5.1 Dataset and Experiments

The *Blitz* dataset consists of 10 months of bank transactions of a retailer. The first six months are used as the training set (1,663,009 transactions) and the four remaining ones as the test set (1,012,380 transactions). The data are described by 17 features and are labeled as *fraud* or *genuine*. The Imbalance Ratio (IR) of the dataset is equal to 0.33%.

The first series of experiments compares the random forest baseline (**RF**) to the tree ensemble algorithm which uses the decision rule presented in Section 6.3 ($\mathbf{RF_X}$). We made different variants of the decision rule:

1. $\mathbf{RF}_{\text{prof}}$: each leaf is labeled according to the majority class of the examples that fall into the leaf. Thus the output of each tree is in {0,1}. The voting criterion is detailed below.

2. $\mathbf{RF}_{\text{max-prof}}$: each leaf is labeled to maximize the profits (i.e. benefits) over the set of all examples in the leaf (the label is 0 if $\gamma_0 > \gamma_1$ and 1 otherwise).

3. $\mathbf{RF}_{\text{mean-prof}}$: this model is the one described in Section 6.3.

For each tree ensemble algorithm, we have used 24 trees with the same maximum depth. Furthermore, for the models $\mathbf{RF}, \mathbf{RF}_{\text{prof}}$ and $\mathbf{RF}_{\text{max-prof}}$, the ensemble classifies a transaction as"fraud" if at least 9 trees agree on this positive class (these hyperparameters are the same as the ones used currently by the Blitz company thus they are chosen for the sake of comparison).

The second series of experiments is dedicated to the analysis of the gradient boosting approaches. We compare our approach $\mathbf{GB}_{\text{prof}}$, presented in Section 6.4.2, with three gradient boosting algorithms which aim to minimize the logistic loss:

1. $\mathbf{GB}_{\text{tune-pre}}$: the threshold has been chosen so that we have the same precision on the validation set as the model **RF** in the training phase.

2. $\mathbf{GB}_{\text{tune-prof}}$: the threshold has been chosen to maximize the benefits on the validation dataset.

3. $\mathbf{GB}_{\text{tune-F1}}$: the threshold has been chosen to maximize the F-Measure F_1 on the validation dataset.

For each of these three experiments, we need a validation dataset to choose the optimal threshold over which a transaction is predicted as fraudulent. For this purpose, the training set is split into two sets: the first one is used to train the model and the other is used to find the best threshold for the given criterion we want to optimize. To do so, the first four months of transactions constitute the training set, the two remaining months are used as the validation set. Finally these three experiments have been conducted on the same *training/validation* set and the algorithms were implemented in **R** [3] using the package **XGBoost**.

For privacy reasons, the explicit expressions of $c_{TP_i}, c_{FP_i}, c_{TN_i}, c_{FN_i}$ of the cost matrix can not be given. Note that they are simple functions of the amount of money M considered in each transaction. For example, we define c_{FP_i} as follows $c_{FP_i} = h(M) - \zeta$, where ζ is a parameter used to translate in financial terms, the dissatisfaction of the customer whose transaction has been rejected.

6.5.2 Results

To measure the performance of each algorithm, we measure the gap between the maximum profits, i.e. the profits obtained if no error is made, and the profits given by the algorithms.

[3] https://www.r-project.org/

We use the classic performance measures that are often used in an imbalanced setting such as the Precision, Recall and F-Measure.

All the experiments have been conducted with the same cost matrix where $\zeta = 5$. The results are presented in Table 6.2. We first notice that we are able to reduce the gap to the maximum profits of 1.43%: from 2.99% with the Random Forest model to 1.56% with the gradient boosting model $\mathbf{GB}_{\text{prof}}$. To give an idea to the reader, having a 1% gap to the maximum profits represents a loss of $43,000$ euros. So, by reducing the gap to 1.43%, we increase the profits of the retailer by $60,000$ euros.

Regarding the Random Forest models, we note that the proposed approaches are able to improve the profits of the retailer compared to the **RF** model. However, we note that $\mathbf{RF}_{\text{prof}}$, which uses the number of examples and their labels to predict the class of the examples in the leaves, gives similar performance as **RF** even if it is built differently. This means that the way to label the leaves has, at least, the same importance as the way to build the trees. The models $\mathbf{RF}_{\text{maj-prof}}$ and $\mathbf{RF}_{\text{mean-prof}}$ which directly use the notion of average profits in each leaf are the two models that give the best results, in terms of both profits, recall and F-Measure even if the precision is reduced. This is explained by the fact that refusing a genuine transaction will have small impact on the retailers profits while accepting a fraud will represent a loss for the retailers that is close to the amount of money in the transaction. Using only our proposed Random Forest algorithm, we are able to reduce the gap to 1.18%.

If we focus now on gradient boosting models, we first note that the model $\mathbf{GB}_{\text{tune-pre}}$ is the one with the highest precision. On the other hand, the other models have a significantly smaller precision but exhibit a higher recall: by maximizing the profits they actually try the highest number of frauds. As mentioned previously, our cost-sensitive approach $\mathbf{GB}_{\text{prof}}$ is the one that achieves the best results in terms of profits. But it has also the worst precision (18.8%) for the reasons given in the previous paragraph. This model provides also better results than $\mathbf{GB}_{\text{tune-prof}}$ which emphasizes the interest of a cost-sensitive approach compared to a simple classification model. However, we note that the $\mathbf{GB}_{\text{tune-F1}}$ model is not the one achieving the best F-Measure at test time. Note that the F_1 score remains low for each presented algorithms. We think that low values are observed because of the complexity of the data and the problem: frauds are rare and spread in the entire data set, thus making them comparable to noise whose behavior is difficult to capture.

In a second part, we want to analyze the effect of the parameter ζ. Indeed, some retailers, for marketing reasons, do not want to reject the transactions of good customers, i.e. they prefer to have a higher precision on their predictions. A simple way to take this into account in our model is to increase the value of ζ. Figure 6.1 shows the impact of the parameter ζ on the *Precision*, *Recall* and F_1, while the gap to the maximum profits is still evaluated with $\zeta = 5$. Recall that some retailers do not want to reject the payment of the good customers, however, the model which maximizes is the one with the lowest precision (16.6%). So, it can be interesting to propose to the retailers several models using different values of ζ and give them the choice of the compromise between profits and precision. We first notice that the higher the value of ζ, the higher the precision and the smaller the recall. However, we see

Experiments	Gap max profits	Precision	Recall	$\mathbf{F_1}$
RF	2.99%	68.1%	5.66%	10.5%
$\mathbf{RF}_{\text{prof}}$	2.88%	73.8%	4.71%	8.86%
$\mathbf{RF}_{\text{maj-prof}}$	1.81%	30.2%	10.6%	15.7%
$\mathbf{RF}_{\text{mean-prof}}$	1.87%	30.3%	9.52%	14.5%
$\mathbf{GB}_{\text{tune-pre}}$	3.01%	**61.0%**	6.49%	11.7%
$\mathbf{GB}_{\text{tune-prof}}$	2.26%	19.1%	**16.6%**	**17.8%**
$\mathbf{GB}_{\text{tune-F1}}$	2.70%	45.4%	9.24%	15.4%
$\mathbf{GB}_{\text{prof}}$	**1.56%**	18.8%	13.3%	15.6%

Table 6.2: Gap to the maximal benefits of each algorithm. In this table, the value of ζ was set to 5. The results are separated into two groups: Random Forest **RF** models and Gradient Boosting **GB** models.

that it is possible to reach a precision which is twice superior to the $\mathbf{GB}_{\text{prof}}$ one by setting $\zeta = 20$ and the gap will still be low with a value of 1.94%.

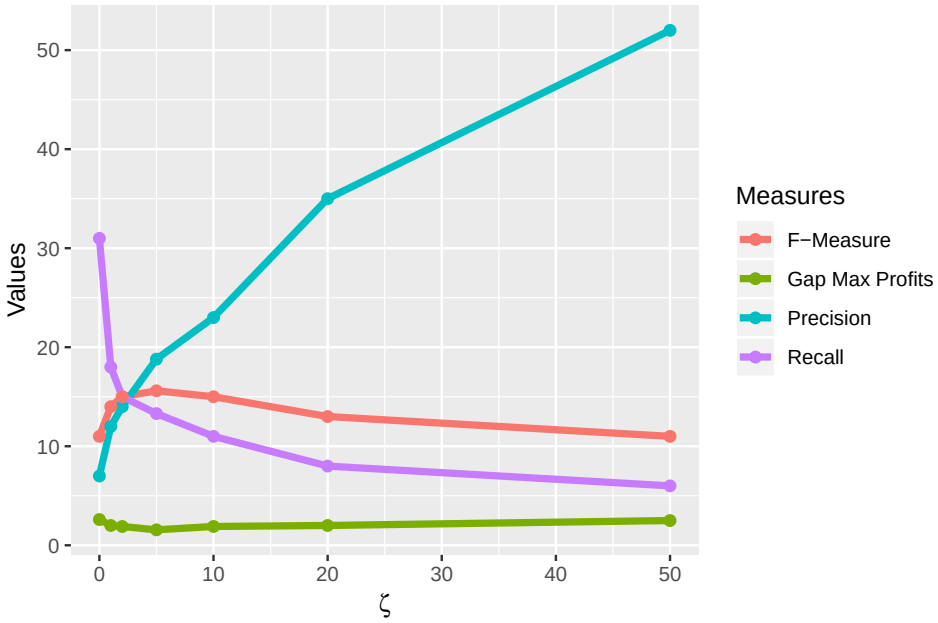

Figure 6.1: Influence of the parameter ζ in the definition of C_{FP_i}. We illustrate the behavior of the Precision, Recall and F_1 according to ζ. We also represent the gap to the maximum profits with respect to ζ. If the model is learned with the ζ values indicated on the x-axis, the gap is computed with $\zeta = 5$ for a fair comparison. By doing so, we are able to see the evolution of the other quantities (Precision, Recall and F_1) with respect to ζ while keeping the same formula to compute the gap.

6.6 Conclusion

We have presented different cost-sensitive tree-based learning strategies to detect frauds in imbalanced retail transaction data. The first strategy is a tree ensemble algorithm which uses a new decision rule which tries to directly optimize the retailer profit. The second one is a gradient boosting algorithm which optimizes a new cost-sensitive loss function. Experiments show that our cost-sensitive algorithms increase the retailers' benefits by 1,43% compared to non cost-sensitive ones and that the gradient boosting approach outperforms all its competitors.

The model presented in this chapter has also been used by the company to compare their performance with the ones of their competitors. For their costumer, it gives an idea of how much they can save by taking the service proposed by the company and how much they can save with the same service proposed by a competitor.

Conclusion and Perspectives

In this thesis, we addressed the problem of learning from imbalanced data in the context of fraud and anomaly detection. The work achieved during this thesis was intended to be eclectic and explored different ways to address the problem of learning in this non trivial setting. Our contributions are mainly separated into two categories: geometric and cost-sensitive ones. In the former, we learned with ME^2 a new representation of the data around each known frauds in order to build risky areas. We also modify the distance to existing frauds in order to increase Voronoi cells and thus the model capacity to capture new frauds. In the latter, we assign costs to each class of examples in order to optimize a suited performance metric for imbalanced scenarios, that is the F-measure, but also to improve the current model of check fraud detection algorithm used by the Blitz company. The contributions of this thesis are both algorithmic and theoretical and led to several papers, among them three were published in international conferences (*ICTAI*, *AISTATS* and *IDA*) and one in an international journal (*Pattern Recognition Letter*). This work also leads to several communications in a french conference (CAp).

Summary of the contributions

There are many techniques in place to address the learning problem in an imbalanced context. This may involve (i) sampling methods to balance the different classes, (ii) learning a new representation of the data in which the classification problem is easier; (iii) exploring the optimization of new cost functions and performance metrics that more adapted to this type of context.

A metric learning method has been the first research axis of this thesis. We studied the notion of risk areas around proven frauds and derived theoretical guarantees on the learned linear projection. This first contribution, essentially theoretical, was to measure how stable the risk areas are, that is, what are the guarantees that a non-fraudulent transaction is predicted to be fraudulent at test time. This work has shown how important the distance to positive examples (or existing frauds) is when learning from imbalanced data. This comment led to our second contribution for which we proposed an adjusted distance in a Nearest Neighbor algorithm. A brief theoretical study explained why it is necessary to compare a test example with a positive data if we want to increase the performance of our model in terms of F-measure. Many experiments have confirmed this claim.

The use of cost-sensitive approaches has constituted the core of our second research axis. Assigning a more important weight on the minority class examples has been shown to be relevant in such context. However, we have shown that this has to be done very carefully. Out third contribution is dedicated to the design of an algorithm which automatically selects the optimal weights to assign to each class for a given set of performance metrics. The proposed algorithm is based on a thorough theoretical study providing potential weighting areas where it is possible to maximize the F-measure. But the use of cost-sensitive methods also presents more concrete applications such as the implementation and optimization of cost functions that have a concrete meaning for the industrial world. Our last contribution used this type of model to improve Blitz's fraud detection system. We proposed a new cost function to optimize a company's profits. This new model takes into account the amount of a transaction in decision-making, which was not the case until now in Blitz.

Perspectives

Beyond the perspectives given in the different chapters, we would like to provide some possible extensions of the contributions of this thesis.

In Chapter 3, a part of the contribution was to provide generalization guarantees on the learned decision areas. A first perspective would be to directly derive generalization guarantees on the F-measure. The main difficulty with the F-measure comes from its non-linearity. We wish to exploit the link between the weighted error function and the performance in terms of F-measure developed in Chapter 5. Using the uniform stability framework (Bousquet and Elisseeff, 2002) on the weighted error function, we aim to draw the link between the observed F-Measure and the expected F-measure at test time. Due to its non-linearity, another possibility to study the performance in terms of F-measure would be to use the bootstrap estimation theory (Efron, 1979; Xu and Goodacre, 2018). We might provide confidence intervals on the estimation and thus on the expected F-measure at test time. It might be also interesting to study the convergence of the proposed algorithm in Chapter 5 which has only been shown in practice.

Regarding the company, several improvements can be made on the currently used fraud detection system. The model is updated every month, assuming that the change in the fraud distribution is significant after a month. However, it would be interesting to set up a method of drift detection to automate the updating of the model (Baena-Garcıa et al., 2006; de Lima Cabral and de Barros, 2018) which could lead to a dramatic improvement of the used system. This opens the door to the development of new domain adaptation methods.

The company is currently working on adding new information in the model that can be provided by the retailers: the *family of products* bought by the customers. This information can be used to detect suspicious behavior because some of these families are more risky than others. However, this information is very parsimonious and takes the form of very large feature vectors with a lot of zeros. To benefit from this new attribute, we will need to study sparsity-inducing learning methods. In this same direction, we also aim to develop a new type of models

which takes the history of the retailers' customers into account. This information is currently based on variables that contain information about the customers history. An interesting perspective would consist in using Long Short Term Memory Networks (LSTM) (Hochreiter and Schmidhuber, 1997) in order to account the history more precisely.

List of Publications

Publication in Journal

Guillaume Metzler, Xavier Badiche, Brahim Belkasmi, Elisa Fromont, Amaury Habrard, and Marc Sebban. Learning maximum excluding ellipsoids from imbalanced data with theoretical guarantees. In *Pattern Recognition Letters*, volume 112, pages 310–316. Elsevier BV, 2018a

Publications in International Conferences

Rémi Viola, Rémi Emonet, Amaury Habrard, Guillaume Metzler, Sébastien Riou, and Marc Sebban. An adjusted nearest neighbor algorithm maximizing the f-measure from imbalanced data. In *In Proceedings of the 31st International Conference on Tools with Artificial Intelligence (ICTAI-2019)*, 2019a

Kevin Bascol, Rémi Emonet, Elisa Fromont, Amaury Habrard, Guillaume Metzler, and Marc Sebban. From cost-sensitive classification to tight F-measure bounds. In *In Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics (AISTATS-19)*, 2019

Guillaume Metzler, Xavier Badiche, Brahim Belkasmi, Elisa Fromont, Amaury Habrard, and Marc Sebban. Tree-based cost sensitive methods for fraud detection. In *International Symposium on Intelligent Data Analysis(IDA- 2018)*, pages 213–224. Springer International Publishing, 2018b

Communications in National Conferences

Rémi Viola, Rémi Emonet, Amaury Habrard, Guillaume Metzler, Sébastien Riou, and Marc Sebban. Une version corrigée de l'algorithme des plus proches voisins pour l'optimisation de la f-mesure dans un contexte déséquilibré. In *Conférence francophone sur l'Apprentissage Automatique (CAp-19)*, 2019b

Kevin Bascol, Rémi Emonet, Elisa Fromont, Amaury Habrard, Guillaume Metzler, and Marc Sebban. Un algorithme d'optimisation de la F-mesure par pondération des erreurs de classification. In *Conférence francophone sur l'Apprentissage Automatique (CAp-18)*, 2018

Guillaume Metzler, Xavier Badiche, Brahim Belkasmi, Elisa Fromont, Amaury Habrard, and Marc Sebban. Apprentissage de sphères maximales d'exclusion avec garanties théoriques. In *Conférence francophone sur l'Apprentissage Automatique (CAp-17)*, 2017

Available Code

CONE Algorithm for F-measure Optimization:
`https://github.com/KBascol/FIAFMO`

Appendix A

Extensions of Chapter 5

A.1 An extension to the multi-class setting

In this section, we consider consider that the output space is $\mathcal{Y} = 1, ..., L$, where L is the number of classes in the data. In this context, given an hypothesis $h \in \mathcal{H}$ learned from a training sample S, the errors that h makes can be summarized in an error profile $\mathbf{e} = \mathbf{E}(h) \in \mathbb{R}^{2L}$ defined as:

$$\mathbf{E}(h) = (\texttt{FN}_1(h), \texttt{FP}_1(h), ..., \texttt{FN}_L(h), \texttt{FP}_L(h)),$$

where $\texttt{FN}_i(h)$ (resp. $\texttt{FP}_i(h)$) is the proportion of False Negative (resp. False Positive) that h yields for class i.

In a multi-class setting with L classes, P_k, $k = 1, ..., L$ denote the proportion of examples in class k and $\mathbf{e} = (e_1, e_2, ..., e_{2L-1}, e_{2L})$ is used to denote the proportions of misclassified examples composing the error profile.

In this setting, we are interested in optimizing the multi-class-micro F-measure, $mcF(\mathbf{e})$ defined by:

$$mcF(\boldsymbol{e}) = \frac{(1+\beta^2)(1 - P_1 - \sum_{k=2}^{L} e_{2k-1})}{(1+\beta^2)(1-P_1) - \sum_{k=2}^{L} e_{2k-1} + e_1}.$$

In this section, we aim to derive all the results presented in the binary case in the multi-class setting.

A.1.1 Pseudo-linearity

Proposition. *The multi-class-micro F-measure, mcF, is a pseudo-linear function with respect to* $\mathbf{e}$.

Proof. As in the binary cases, we have to prove that both mcF and $-mcF$. The gradient of the multi-class-micro F-measure, mcF_β, is defined by:

$$\nabla mcF(\mathbf{e}) = \frac{-(1+\beta^2)}{(1+\beta^2)(1-P_1) - \sum_{k=2}^{L} e_{2k-1} + e_1} \begin{cases} 1 - P_1 - \sum_{k=2}^{L} e_{2k-1} & \text{w.r.t. } e_1, \\ \beta^2(1-P_1) + e_1 & \text{w.r.t. } e_k, \ \ \forall k = 2, ..., L. \end{cases}$$

The proof is similar to the proof of Proposition 5.1, we simply have to do the following changes of notation in the proof:

$$\begin{aligned} e_1 &\leftarrow \sum_{k=2}^{L} e_{2k-1}, \\ e_2 &\leftarrow e_1, \\ P &\leftarrow 1 - P_1. \end{aligned}$$

□

A.1.2 A tight bound on the optimal micro F-measure

As it was done in the binary case, we will use the property of pseudo-linearity of $mcF(\mathbf{e})$ to bound the difference of micro F-measure in terms of the parameters of our weighted function. First, we introduce the definition of our weighted function $\mathbf{a} : \mathbb{R} \to \mathbb{R}^{2L}$ and express the difference of micro F-measure of two error profiles in function of the two error profiles.

In this section, for the sake of creadability, we will set $e_L = \sum_{k=2}^{L} e_{2k-1}$.

Step 1: impact of a change in the error profile

Using the property of pseudo-linearity, we can show that it exists two functions $\mathbf{a} : \mathbb{R} \to \mathbb{R}^{2L}$ and $b : \mathbb{R} \to \mathbb{R}$ defined by:

$$0 = \langle \mathbf{a}(mcF(\mathbf{e})), \mathbf{e} \rangle + b(mcF(\mathbf{e})),$$

where:

$$\mathbf{a}(t) = \begin{cases} 1 + \beta^2 - t & \text{for } e_{2k-1},\ k = 2, ..., L \\ t & \text{for } e_1, \\ 0 & \text{otherwise,} \end{cases} \qquad \text{and} \quad b(t) = (t-1)(1+\beta^2)(1-P_1).$$

From these definitions we can write:

$$\begin{aligned} \langle \mathbf{a}(mcF(\mathbf{e}')), \mathbf{e} - \mathbf{e}' \rangle &= \langle \mathbf{a}(mcF(\mathbf{e}')), \mathbf{e} \rangle + b(mcF(\mathbf{e}')), \\ &\quad \downarrow \text{ using the pseudo-linearity of } mcF \\ &= \underbrace{\langle \mathbf{a}(mcF(\mathbf{e}')) - \mathbf{a}(mcF(\mathbf{e})), \mathbf{e} \rangle}_{} \underbrace{- b(mcF(\mathbf{e})) + b(mcF(\mathbf{e}'))}_{}, \\ &\quad \downarrow \text{ using the definition of both } \mathbf{a} \text{ and } b \\ &= (mcF(\mathbf{e}') - mcF(\mathbf{e}))\underbrace{(1+\beta^2)(1-P_1)}_{} \\ &\quad + (mcF(\mathbf{e}') - mcF(\mathbf{e}))\underbrace{e_1} + (mcF(\mathbf{e}) - mcF(\mathbf{e}'))\underbrace{e_L}, \\ &\quad \downarrow \text{ factorizing} \\ &= (mcF(\mathbf{e}') - mcF(\mathbf{e}))\big((1+\beta^2)(1-P_1) - e_L + e_1\big). \end{aligned}$$

We can now write the difference of micro-F-measure as:

$$\begin{aligned} mcF(\mathbf{e}') - mcF(\mathbf{e}) &= \frac{1}{((1+\beta^2)(1-P_1) - e_L + e_1)} \cdot \langle \mathbf{a}(mcF(\mathbf{e})), \mathbf{e} - \mathbf{e}' \rangle, \\ &= \Phi_{\mathbf{e}} \cdot \langle \mathbf{a}(mcF(\mathbf{e}), \mathbf{e} - \mathbf{e}' \rangle, \end{aligned}$$

where:

$$\Phi_{\mathbf{e}} = \frac{1}{(1+\beta^2)(1-P_1) - e_L + e_1},$$

Step 2: a bound on the micro F-measure $mcF(\mathbf{e})$

Let $t \in [0,1]$. Let us suppose that we have learned a classifier h with the cost function $\mathbf{a}(t)$. This classifier leads to an error profile $\mathbf{e}$ and a micro-F-measure $mcF(\mathbf{e})$. We now imagine a hypothetical classifier that is learned with weights $\mathbf{a}(t')$, and we denote by $\mathbf{e}'$ the error profile of this classifier. For any value of $t' \in [0,1]$, we derive an upper bound on the on the F-measure $mcF(\mathbf{e}')$ that this hypothetical classifier can achieve.

$$\begin{aligned} mcF(\mathbf{e}') - mcF(\mathbf{e}) &= \Phi_{\mathbf{e}} \cdot \langle \mathbf{a}(t'), \mathbf{e} - \mathbf{e}' \rangle, \\ &\quad \downarrow \text{ linearity of the inner product} \\ &= \Phi_{\mathbf{e}} \cdot \Big(\underbrace{\langle \mathbf{a}(t'), \mathbf{e} \rangle} - \langle \mathbf{a}(t'), \mathbf{e}' \rangle \Big), \\ &\quad \downarrow \text{ using } \mathbf{a}(t') = \mathbf{a}(t') + \mathbf{a}(t) - \mathbf{a}(t) \\ &= \Phi_{\mathbf{e}} \cdot \Big(\underbrace{\langle \mathbf{a}(t') - \mathbf{a}(t), \mathbf{e} \rangle} + \langle \mathbf{a}(t), \mathbf{e} \rangle - \langle \mathbf{a}(t'), \mathbf{e}' \rangle \Big), \\ &\quad \downarrow \text{ using the definition of } \mathbf{a} \\ &= \Phi_{\mathbf{e}} \cdot \Big((t'-t)(e_1 - e_L) + \underbrace{\langle \mathbf{a}(t), \mathbf{e} \rangle} - \langle \mathbf{a}(t'), \mathbf{e}' \rangle \Big), \\ &\quad \downarrow \text{ using the sub-optimality of the classifier} \\ &\leq \Phi_{\mathbf{e}} \cdot \Big(\underbrace{\langle \mathbf{a}(t), \mathbf{e}' \rangle + \varepsilon_1 - \langle \mathbf{a}(t'), \mathbf{e}' \rangle} + (t'-t)(e_1 - e_L) \Big), \\ &\quad \downarrow \text{ using the definition of } \mathbf{a} \\ &\leq \Phi_{\mathbf{e}} \cdot \big((t'-t)(e_1 - e_L) + \varepsilon_1 - (t'-t)(e'_1 - e'_L) \big), \\ &\quad \downarrow \text{ rearranging the terms} \\ &\leq \Phi_{\boldsymbol{e}} \varepsilon_1 + \Phi_{\mathbf{e}} \cdot (e_1 - e_L - (e'_1 - e'_L))(t'-t). \end{aligned}$$

In the previous development, we have used the linearity of the inner product and the definition of $\mathbf{a}$. The first inequality uses the sub-optimality of the learned classifier. We then use the definition of the function $\mathbf{a}$.

As in the binary cases, the quantity $e'_1 - e'_L$ remains unknown but we try to optimize this difference according to the sign of $t' - t$ and under the constraint $mcF(\mathbf{e}') > mcF(\mathbf{e})$. So the last inequality becomes:

$$mcF(\mathbf{e}') \leq mcF(\mathbf{e}) + \Phi_{\mathbf{e}} \varepsilon_1 + \Phi_{\mathbf{e}}(e_1 - e_L - M_{\max})(t'-t), \quad \text{if} \quad t' < t,$$

$$mcF(\mathbf{e}') \leq mcF(\mathbf{e}) + \Phi_{\mathbf{e}}\varepsilon_1 + \Phi_{\mathbf{e}}(e_1 - e_L - M_{\min})(t' - t), \quad \text{if} \quad t' > t,$$

where $M_{\max} = \max_{\substack{\mathbf{e}'' \in \mathcal{E}(\mathcal{H}) \\ s.t.\ F(\mathbf{e}'') > F(\mathbf{e})}} (e_2'' - e_1'')$ and $M_{\min} = \min_{\substack{\mathbf{e}'' \in \mathcal{E}(\mathcal{H}) \\ s.t.\ F(\mathbf{e}'') > F(\mathbf{e})}} (e_2'' - e_1'')$.

These two inequalites are the consequence of Lemma 5.1 adapted to the present setting. It remains to compute the values of both $M_{\max}$ and $M_{\min}$. Unfortunately, in the next section, we will see that we can not give an explicit value of these numbers.

A.1.3 Computation of $M_{\max}$ and $M_{\min}$

To compute the value of both $M_{\max}$ and $M_{\min}$, we use the same development as done in the binary setting. We do not write how to derive the new set of constraints. We now search how to modify the vector $\mathbf{e}$ in order to improve the F-Measure and to maximize (or minimize) the difference: $e_1' - e_L'$, where $\mathbf{e}' = \mathbf{e} + \boldsymbol{\alpha}$. Thus, the original optimization problem can be rewritten as an optimization problem on $\boldsymbol{\alpha}$.

Computation of $M_{\max}$

$$\begin{aligned}
\max_{\boldsymbol{\alpha}} \quad & \alpha_1 - \sum_{k=2}^{L} \alpha_{2k-1}, \\
s.t. \quad & \alpha_1 < -\sum_{k=2}^{L} \alpha_{2k-1} \frac{\beta^2(1 - P_1) + e_1}{1 - P_1 - \sum_{k=2}^{L} e_{2k-1}}, \\
& \alpha_1 \in [-e_1, P_1 - e_1], \\
& \alpha_{2k-1} \in [-e_{2k-1}, P_{2k-1} - e_{2k-1}], \ \forall k = 2, ..., L.
\end{aligned}$$

Then we add the quantity $e_1 - e_L$ to this result to have the value M_{max}.

Computaion of $M_{\min}$

$$\begin{aligned}
\min_{\boldsymbol{\alpha}} \quad & \alpha_1 - \sum_{k=2}^{L} \alpha_{2k-1}, \\
s.t. \quad & \alpha_1 < -\sum_{k=2}^{L} \alpha_{2k-1} \frac{\beta^2(1 - P_1) + e_1}{1 - P_1 - \sum_{k=2}^{L} e_{2k-1}}, \\
& \alpha_1 \in [-e_1, P_1 - e_1], \\
& \alpha_{2k-1} \in [-e_{2k-1}, P_{2k-1} - e_{2k-1}], \ \forall k = 2, ..., L.
\end{aligned}$$

Then we add the quantity $e_1 - e_L$ to this result to have the value M_{min}.

In the multi-class, we have to look for a solution of an optimization problem where the set of constraints consists of polyhedron and a hyperplane. Due to the dimensionality of the problem, we can not use the same trick as in the binary setting. However, such optimization problem can easily be solved by any solver.

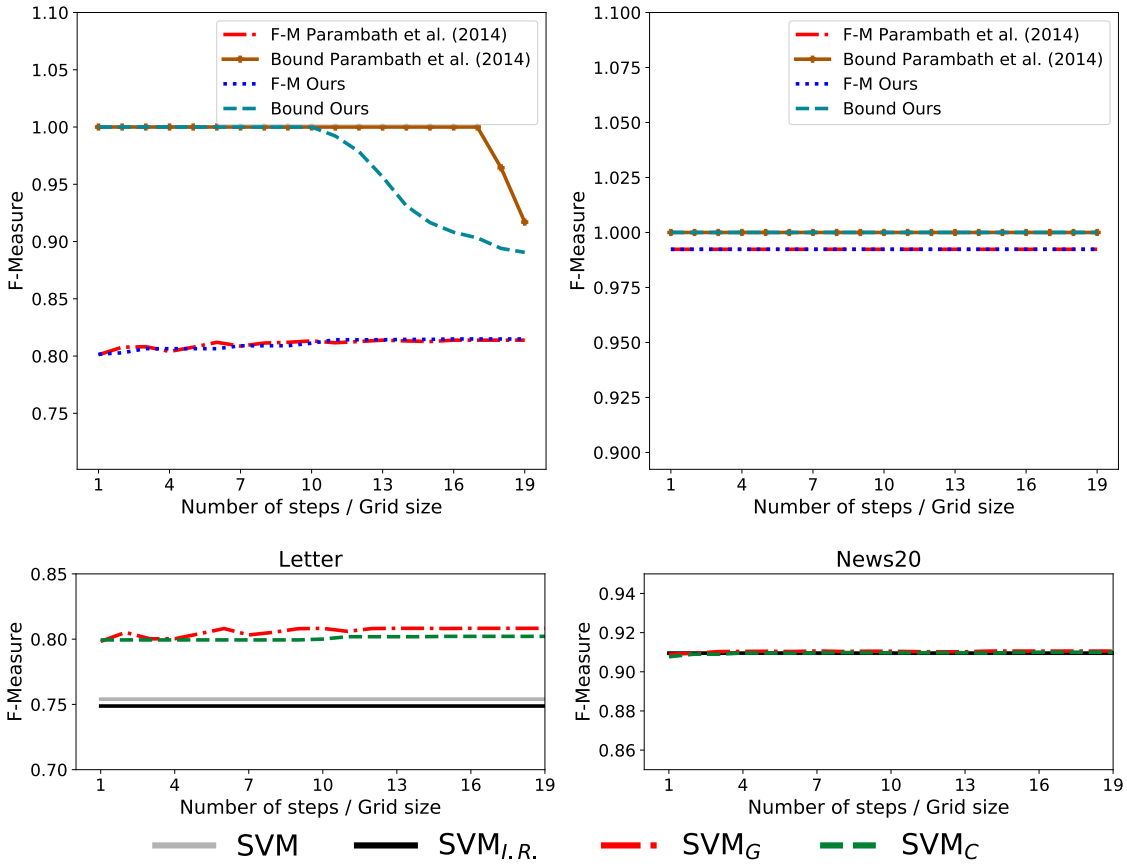

Figure A.1: Comparison of our bound and one the one provided by Parambath et al. (2014) and their respective performance according to the number of steps/grid size (above). Comparson of the performance of different algorithms (below).

The next section is dedicated to the experimental evaluation of the proposed method on multiclass datasets.

A.1.4 Extended Experiments

We do not provide all the graphs, we will simply compare the performance of several algorithms and our bounds with the one provided by Parambath et al. (2014). On both datasets, the performance of our proposed approach is similar to the one based on a grid it is emphasizes on the *News20* dataset. The bounds of both methods are the same, on this dataset, during the training phase due to a value of F-measure closed to 1. However our bound remains more informative after 10 iterations on the *Letter* dataset.

Finally, Tables A.1 provide the F-measure values at different steps/ size of the grid to emphasize the fact that only few iterations are enough to at least better perform than state of the art methods.

Table A.1: Mean F-Measure over 5 experiments and **limiting the number of iterations/grid steps to respectively 2,8 and 15** (standard deviation between brackets).

Datasets	SVM	SVM$_{I.R.}$	SVM$_G$	SVM$_C$	LR	LR$_{I.R.}$	LR$_B$	LR$_G$	LR$_C$
Adult	62.5 (0.2)	64.9 (0.3)	66.4 (0.2)	66.2 (0.3)	63.1 (0.1)	66.0 (0.1)	66.6 (0.1)	66.6 (0.1)	66.2 (0.1)
Abalone10	0.0 (0.0)	30.9 (1.2)	32.6 (1.4)	30.7 (1.1)	0.0 (0.0)	31.9 (1.4)	31.6 (0.6)	31.9 (1.7)	32.4 (1.9)
Satimage	0.0 (0.0)	23.4 (4.3)	6.1 (12.2)	5.9 (11.8)	0.5 (0.9)	24.2 (5.3)	21.4 (4.6)	6.2 (12.3)	6.1 (12.2)
IJCNN	44.5 (0.4)	53.3 (0.4)	60.7 (0.4)	61.6 (0.5)	46.2 (0.3)	51.6 (0.3)	59.2 (0.3)	56.8 (0.3)	58.3 (0.3)
Abalone12	0.0 (0.0)	16.8 (2.7)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	18.0 (3.5)	17.7 (3.7)	2.8 (3.4)	13.3 (3.5)
Pageblocks	48.1 (5.8)	39.6 (4.7)	65.0 (7.6)	63.3 (4.1)	48.6 (3.3)	42.4 (5.2)	55.7 (5.7)	62.7 (7.1)	58.3 (6.8)
Yeast	0.0 (0.0)	29.4 (2.9)	30.9 (17.2)	25.4 (17.5)	2.5 (5.0)	29.0 (3.5)	35.4 (15.6)	27.8 (20.0)	33.0 (18.3)
Wine	0.0 (0.0)	15.6 (5.2)	0.0 (0.0)	11.7 (11.1)	0.0 (0.0)	14.6 (3.2)	18.3 (7.2)	8.7 (11.2)	15.6 (6.7)
Letter	75.4 (0.7)	74.9 (0.8)	80.7 (0.5)	80.4 (0.5)	82.9 (0.3)	82.9 (0.3)	74.9 (0.5)	82.9 (0.2)	82.8 (0.2)
News20	90.9 (0.1)	91.0 (0.2)	90.9 (0.2)	91.0 (0.2)	90.6 (0.1)	90.6 (0.1)	89.4 (0.2)	90.6 (0.2)	90.6 (0.1)
Average	32.1 (0.7)	44.0 (2.3)	43.3 (4.0)	43.6 (4.7)	33.4 (1.0)	45.1 (2.3)	47.0 (3.9)	43.7 (5.6)	45.7 (5.0)

Datasets	SVM	SVM$_{I.R.}$	SVM$_G$	SVM$_C$	LR	LR$_{I.R.}$	LR$_B$	LR$_G$	LR$_C$
Adult	62.5 (0.2)	64.9 (0.3)	66.4 (0.1)	66.5 (0.1)	63.1 (0.1)	66.0 (0.1)	66.6 (0.1)	66.5 (0.1)	66.5 (0.1)
Abalone10	0.0 (0.0)	30.9 (1.2)	32.6 (1.4)	32.6 (1.0)	0.0 (0.0)	31.9 (1.4)	31.6 (0.6)	32.1 (0.8)	31.4 (2.2)
Satimage	0.0 (0.0)	23.4 (4.3)	20.2 (4.7)	20.6 (5.6)	0.5 (0.9)	24.2 (5.3)	21.4 (4.6)	20.3 (5.0)	20.5 (5.0)
IJCNN	44.5 (0.4)	53.3 (0.4)	61.9 (0.7)	61.5 (0.5)	46.2 (0.3)	51.6 (0.3)	59.2 (0.3)	58.0 (0.4)	58.1 (0.3)
Abalone12	0.0 (0.0)	16.8 (2.7)	16.9 (2.9)	18.3 (3.3)	0.0 (0.0)	18.0 (3.5)	17.7 (3.7)	17.5 (3.4)	18.1 (3.7)
Pageblocks	48.1 (5.8)	39.6 (4.7)	65.8 (4.3)	62.8 (3.9)	48.6 (3.3)	42.4 (5.2)	55.7 (5.7)	60.0 (8.8)	59.4 (7.5)
Yeast	0.0 (0.0)	29.4 (2.9)	33.3 (12.2)	39.0 (7.5)	2.5 (5.0)	29.0 (3.5)	35.4 (15.6)	39.4 (8.5)	38.9 (8.6)
Wine	0.0 (0.0)	15.6 (5.2)	19.5 (6.2)	22.4 (6.1)	0.0 (0.0)	14.6 (3.2)	18.3 (7.2)	18.7 (4.5)	21.1 (5.2)
Letter	75.4 (0.7)	74.9 (0.8)	80.6 (0.4)	80.5 (0.4)	82.9 (0.3)	82.9 (0.3)	74.9 (0.5)	82.9 (0.2)	82.9 (0.3)
News20	90.9 (0.1)	91.0 (0.2)	91.0 (0.1)	91.0 (0.2)	90.6 (0.1)	90.6 (0.1)	89.4 (0.2)	90.6 (0.1)	90.6 (0.1)
Average	32.1 (0.7)	44.0 (2.3)	48.8 (3.3)	49.5 (2.9)	33.4 (1.0)	45.1 (2.3)	47.0 (3.9)	48.6 (3.2)	48.8 (3.3)

Datasets	SVM	SVM$_{I.R.}$	SVM$_G$	SVM$_C$	LR	LR$_{I.R.}$	LR$_B$	LR$_G$	LR$_C$
Adult	62.5 (0.2)	64.9 (0.3)	66.4 (0.1)	66.5 (0.1)	63.1 (0.1)	66.0 (0.1)	66.6 (0.1)	66.5 (0.1)	66.5 (0.1)
Abalone10	0.0 (0.0)	30.9 (1.2)	32.6 (1.4)	32.6 (1.0)	0.0 (0.0)	31.9 (1.4)	31.6 (0.6)	32.1 (0.8)	31.4 (2.2)
Satimage	0.0 (0.0)	23.4 (4.3)	20.2 (4.7)	20.6 (5.6)	0.5 (0.9)	24.2 (5.3)	21.4 (4.6)	20.3 (5.0)	20.5 (5.0)
IJCNN	44.5 (0.4)	53.3 (0.4)	61.9 (0.7)	61.5 (0.5)	46.2 (0.3)	51.6 (0.3)	59.2 (0.3)	58.0 (0.4)	58.1 (0.3)
Abalone12	0.0 (0.0)	16.8 (2.7)	16.9 (2.9)	18.3 (3.3)	0.0 (0.0)	18.0 (3.5)	17.7 (3.7)	17.5 (3.4)	18.1 (3.7)
Pageblocks	48.1 (5.8)	39.6 (4.7)	65.8 (4.3)	62.8 (3.9)	48.6 (3.3)	42.4 (5.2)	55.7 (5.7)	60.0 (8.8)	59.4 (7.5)
Yeast	0.0 (0.0)	29.4 (2.9)	33.3 (12.2)	39.0 (7.5)	2.5 (5.0)	29.0 (3.5)	35.4 (15.6)	39.4 (8.5)	38.9 (8.6)
Wine	0.0 (0.0)	15.6 (5.2)	19.5 (6.2)	22.4 (6.1)	0.0 (0.0)	14.6 (3.2)	18.3 (7.2)	18.7 (4.5)	21.1 (5.2)
Letter	75.4 (0.7)	74.9 (0.8)	80.6 (0.4)	80.5 (0.4)	82.9 (0.3)	82.9 (0.3)	74.9 (0.5)	82.9 (0.2)	82.9 (0.3)
News20	90.9 (0.1)	91.0 (0.2)	91.0 (0.1)	91.0 (0.2)	90.6 (0.1)	90.6 (0.1)	89.4 (0.2)	90.6 (0.1)	90.6 (0.1)
Average	32.1 (0.7)	44.0 (2.3)	48.8 (3.3)	49.5 (2.9)	33.4 (1.0)	45.1 (2.3)	47.0 (3.9)	48.6 (3.2)	48.8 (3.3)

Appendix B

French Translations

Table des Matières

B.1 Introduction

L'apprentissage machine est un sous-domaine de l'intelligence artificielle se situant à la frontière entre l'informatique et les mathématiques appliquées (statistiques et optimisation). Cette discipline recoupe aussi en partie la science des données puisqu'elle est basée sur la collecte de

d'informations qui sont analysées et étudiées afin d'en extraire les informations substantielles qui pouvant être utilisées pour des applications spécifiques. Selon l'application, la nature des données peut être multiple : il peut s'agir d'images, de vidéos, de données brutes, de données catégorielles, d'arbres, de graphes, de séries chronologiques, etc.

Une fois que les données sont collectées, et souvent complétées et nettoyées, elles peuvent être utilisées pour plusieurs tâches d'apprentissage telles que la régression quand il s'agit, par exemple, de prédire le prix d'une action ou le prix d'une maison selon ses caractéristiques. Elles peuvent également être exploitées pour des tâches de classification lorsque, par exemple, on vise à distinguer un spam d'un mail "normal" lors de la réception d'un e-mail. Nous pouvons également chercher à identifier si une transaction est frauduleuse ou authentique, à détecter des anomalies dans un examen médical comme un test sanguin. Dans tous les cas, les données sont étiquetées à l'aide d'une variable y que l'on appelle l'étiquette. Lorsque nous avons une transaction authentique ou un courriel classique, l'exemple est généralement étiqueté -1 (aussi appelé exemple négatif) alors qu'il est étiqueté 1 (exemple positif) lorsqu'il s'agit d'un objet d'intérêt comme un spam ou une transaction frauduleuse. Lorsqu'une telle information étiquetée est utilisée dans un algorithme d'apprentissage machine, on parle d'apprentissage supervisé, et d'apprentissage non supervisé le cas échéant.

Portés par l'application à la détection de fraude, nous nous intéresserons, au sein de ce document, plus particulièrement à la classification binaire. Notre objectif est d'apprendre un classifieur h, également appelé hypothèse, en utilisant un ensemble de données étiquetées $\mathbf{x}_i, y_i{}_{i=1}^{m}$ afin de classer de nouveaux exemples, où $\mathbf{x}_i$ désigne l'ensemble des caractéristiques de l'individu i.

Cette thèse CIFRE a été effectué en collaboration avec l'entreprise Blitz Business Services, une PME travaillant sur la détection de fraude bancaire par chèque. Lorsqu'un client souhaite régler ses achats en magasin, il dispose de plusieurs moyens de paiements comme le chèque, l'espèce ou la carte bancaire. L'entreprise Blitz offre à ses clients (principalement les enseignes de la Grande Distribution) un moyen de sécuriser les transactions par chèques en détectant des transactions dites frauduleuses qui peuvent être de deux natures différentes: elles peuvent correspondre à l'utilisation d'un faux-chèques, i.e. un chèque qui n'est pas issu d'un établissement bancaire; ou une transaction impayée, lorsque que le chèque est encaissé mais que le propriétaire du chèque ne dispose pas de suffisamment de fonds sur son compte bancaire. L'entreprise Blitz n'a qu'un rôle de conseil et la décision finale reste à la charge de ses clients.

La difficulté de ce type de tache est double: la première consiste à considérer et traiter un nombre important de transactions chaque année et de fournir une réponse (i.e. un conseil) à ses clients en un temps très limité (de l'ordre de la dizaine de millisecondes. Deuxièmement, une fraude constitue ce que l'on appelle un évènement rare: ces fraudent représentent uniquement 0.4% des transactions alors que cela représente plus de 1% du chiffre d'affaire de la grande distribution.

Cette thèse s'inscrit donc dans un contexte d'apprentissage dans un domaine déséquilibré. En d'autres termes, lorsque la classe qui intéresse l'utilisateur est sous représenté par rapport

aux autres classes. Dans un tel contexte, la plupart des algorithmes de classification, essentiellement basés sur la minimisation du taux d'erreurs, conduisent à des solutions triviales et prédisent tous les exemples comme étant négatifs ou non frauduleux. Ainsi, en raison du fort déséquilibre présent dans nos données, un moyen aisé de parvenir à de très bonnes performances (plus de $99,6\%$) consiste à prédire toutes les transactions comme étant non frauduleuses, ce qui est inconcevable car nous passons totalement à côté du problème que l'on s'est posé.

Plus généralement, apprendre dans un contexte déséquilibré revêt d'un enjeu majeur dans la communauté en Apprentissage Machine à cause de son large champ applicatif dans le contexte bancaire, médical ou encore dans le milieu des assurances. Cela représente également un grand challenge pour la communauté car il nécessite de revoir complètement les algorithmes qui existent pour qu'ils puissent répondre à notre tâche. Plusieurs techniques ont été développé en ce sens comme l'apprentissage de nouvelles représentations des données qui permettent de classer plus facilement les données, l'attribution de poids aux différentes erreurs effectuées par l'algorithme ou l'utilisation de méthodes d'échantillonnages pour rééquilibrer les différentes classes en présence.

L'objectif de cette thèse est de proposer de nouvelles stratégies qui permettent de traiter le problème d'apprentissage dans un contexte déséquilibré et de les appliquer à la problématique de détection de fraudes. On se propose de développer de nouveaux algorithmes adaptés à ce contexte et qui permettent d'améliorer le modèle de classification utilisé par l'entreprise Blitz. Une partie de cette thèse est également dédiée à des contributions plus théoriques dans le cadre de l'apprentissage dans un domaine déséquilibré en proposant de nouvelles façon d'optimiser des mesures de performances adaptées à notre contexte.

Les contributions de cette thèse se divisent deux parties. La première partie se concentre sur des approches dites géométriques dans lesquelles des algorithmes d'apprentissage de représentations sont développées. Cette nouvelle représentation permettant une meilleure classification des données à l'aide d'hypothèses simples. Dans un deuxième temps nous étudions une approche par pondérations des erreurs afin d'améliorer l'optimisation de mesures de performances adaptées à ce contexte et nous proposons également une application concrète de ce type de méthodes en développant de nouvelles approches permettant d'améliorer le système de l'entreprise Blitz, i.e. en proposant un algorithme permettant d'optimiser la marge de la grande enseigne.

Contexte de la thèse. Cette thèse a été effectué au sein de l'équipe Data Intelligence du Laboratoire Hubert Curien UMR CNRS 5516, affilié à l'Université Jean-Monnet à Saint-Etienne et à l'Université de Lyon. Elle a également été conduite au sein de l'entreprise Blitz Business Services se situant à Villefontaine, France. Ce travail a été effectué sous un contrat CIFRE financé par l'ANRT (Agence Nationale pour la Recherche Technologique).

A propos de l'entreprise Blitz Business Services. La société Blitz Business Services est située à Villefontaine, France. Son activité principale est la détection de fraude par

chèque et ses clients sont la Grande Distribution. Elle travaille également sur l'optimisation du passage en caisse ou sur l'octroi de facilités de paiement, comme l'accord d'un paiement en plusieurs mensualités sur des sites de vente en ligne par exemple. Enfin, Blitz développe une activité qui consiste à optimiser l'utilisation des lettres de rappel et des appels lorsqu'une contravention a été établi pour une personne ne disposant pas d'un titre de transport valable dans les transports publics (SNCF).

Contributions

Ce manuscrit présente différentes contributions à la fois théoriques et pratiques pour aborder les questions d'apprentissage dans un contexte déséquilibré. Le corps principal de cette thèse contient les contributions de cette thèse. Les extensions et les expériences supplémentaires sont exposées en annexe pour une meilleure lisibilité.

La première partie se décompose en deux chapitres:

Chapter 1. Dans ce premier chapitre, nous présentons le domaine de l'apprentissage statistique: de l'apprentissage d'un modèle jusqu'à sa capacité à bien fonctionner sur des données non visualisées mais issues de la même distribution. Il présente également les définitions et les notations utiles pour le reste du document. Nous terminons ce chapitre en introduisant le problème de l'apprentissage à partir de données déséquilibrées et en illustrant le comportement de certains algorithmes d'apprentissage machine lorsqu'ils sont confrontés à de telles données.

Chapter 2. Le deuxième chapitre se concentre sur l'état de l'art des techniques utilisées pour aborder la question de l'apprentissage déséquilibré. Il introduit des généralités sur la détection des anomalies et des fraudes dans les scénarios d'apprentissage très déséquilibrés. Il est également consacré à la présentation des indicateurs de performance pertinents utilisés dans un tel contexte.

Dans la deuxième partie de cette thèse, nous présentons deux contributions basées sur des approches géométriques pour traiter des données déséquilibrées. Les deux visent à apprendre les régions d'influence autour des exemples positifs, en utilisant l'approche d'apprentissage métrique (i) dans le chapitre 3 et (ii) une distance ajustée par rapport aux données positives dans le chapitre 4.

Chapter 3. Partant de l'hypothèse que les fraudes sont localement concentrées les unes aux autres [1]. Notre première contribution propose une nouvelle stratégie basée sur les Support Vector Data Description, une variante des SVM qui vise à rassembler des exemples ensembles.

[1] Même si les fraudes sont rares, un fraudeur donné agit généralement rapidement avec la même stratégie, conduisant à quelques exemples positifs dans une même région de l'espace.

Nous proposons d'en modifier la formulation standard afin de créer la plus grande surface autour de chaque instance positive que l'on appellera zone à risques. La surface de cette zone est optimisée à l'aide d'une stratégie d'apprentissage de métrique et conduit à des ellipsoïdes dans notre espace initial, pour lequel la taille et l'orientation sont contrôlées par des termes de régularisations. En résolvant la formulation duale de notre problème d'optimisation, nous montrons que, comparé à la plupart des techniques d'apprentissage de métrique, le caractère semi définie positif de la métrique est obtenu "gratuitement". Nous obtenons également des garanties de généralisation sur la méthode proposée en utilisant le cadre de stabilité uniforme et montrons son efficacité, en termes de F-Mesure, par rapport aux algorithmes d'apprentissage machine standard combinés à des méthodes d'échantillonnage. Sur l'ensemble de données privées de Blitz, nous montrons qu'il est possible de contrôler soit le rappel, soit la précision du modèle en jouant sur les hyper-paramètres.

Chapter 4. Les exemples positifs de l'ensemble d'entraînement ont un rôle important lorsqu'il s'agit de détecter de nouveaux positifs en phase de test. Lorsque nous travaillons avec des algorithmes basés sur la distance, comme l'algorithme k- plus proches voisins, la distance par rapport à ces points positifs est essentielle. Au lieu d'appliquer une transformation linéaire des données comme c'est habituellement le cas, nous montrons qu'il suffit de modifier la distance d'une nouvelle requête uniquement par rapport exemples positifs pour améliorer significativement cet algorithme. Notre méthode a aussi la capacité de contrôler le taux de faux positifs et de faux négatifs. De plus, nous montrons, sur plusieurs ensembles de données, que la performance de l'algorithme proposé peut aussi être améliorée lorsqu'il est combiné avec des stratégies d'échantillonnages et atteint des performances au moins égales à des approches d'apprentissage de métrique ou une autres variantes du k-NN.

La troisième et dernière partie de cette thèse se concentre sur deux contributions sur les méthodes sensibilisation aux coûts. Dans le chapitre 5, nous optimisons la F-mesure et fournissons des bornes sur sa valeur maximale. Le chapitre 6 utilise cette approche pour une application pratique : l'optimisation des bénéfices des de la Grande Distribution.

Chapter 5. L'utilisation de mesures spécifiques telles que la F-mesure s'avère plus pertinente dans le contexte d'un apprentissage déséquilibré. Mais son optimisation reste une tâche difficile en raison de sa non-linéarité et de sa non convexité. Cependant, en raison d'une propriété de la F-mesure, nous sommes en capacité de faire le lien entre son optimisation et l'apprentissage par pondération des erreurs, c'est-à-dire l'apprentissage en assignant des coûts différents à chaque classe. Partant de ce constat, nous proposons de nouvelles bornes sur la différence de F-mesure entre deux hypothèses. Ainsi, nous sommes en mesure de donner des limites sur la mesure F-mesure optimale atteignable. Nous fournissons également une interprétation géométrique de ces bornes sous forme de cônes asymétriques. Ce dernier point permet de dériver un algorithme itératif qui choisit les bons coûts à affecter à chaque classe afin d'optimiser la mesure de performance étudiée. Nous montrons que la méthode

présentée est au moins tout aussi efficace que ses concurrents et ne nécessite qu'un petit nombre d'itérations pour parvenir a de meilleurs résultats. D'un point de vue théorique, nous montrons également que les bornes proposées sont beaucoup plus fines, précises, que celles proposées dans la littérature.

Chapter 6. Les approches par pondération des erreurs sont pertinentes pour régler la question du déséquilibre de la classification. Dans ce chapitre, nous proposons de combiner ces approches à des algorithmes basés sur des arbres dans un cadre applicatif pour la société Blitz : l'optimisation des bénéfices pour les distributeurs. Le montant d'une transaction est une information pertinente dans le cadre de la classification d'une transaction. Une plus grande attention doit être accordée aux transactions avec un fort montant, ce que les modèles précédents ne prenaient pas en compte. De plus, une classification erronée sur ce type d'opérations a des conséquences plus lourdes que celle d'un petit montant. Nous proposons matrice de pondération des résultats de la classification et, par conséquent, une fonction de perte qui vise à optimiser les bénéfices de la Grande Distribution. Nous proposons également différentes méthodes fondées sur des arbres pour optimiser cette marge. Les expériences réalisées dans ce chapitre montrent qu'il est possible d'augmenter considérablement la marge dela grande enseigne en utilisant une approche par pondération des erreurs et que l'algorithme de *Gradient Tree Boosting* proposé donne de meilleurs résultats que les algorithmes fondés sur les forêts aléatoires.

Appendice A Nous fournissons une extension au contexte multi-classe du travail présenté dans le chapitre 5. Cette extension inclut les preuves et des expériences supplémentaires.

B.2 Résumé des Chapitres

B.2.1 Résumé du Chapitre 1

Ce premier chapitre introduit permet de présenter le contexte dans lequel s'inscrit cette thèse, à savoir l'apprentissage statistiques. Plus précisément l'apprentissage de modèle prédictif permettant la résolution de certaines tâches, comme de la classification d'images ou encore l'estimation de grandeurs statistiques ou réelles. Après avoir présenté les différentes tâches que l'on peut rencontrer dans l'apprentissage statistique, nous présentons son formalisme de façon plus concise ainsi que ses fondements. A savoir (i) comment apprendre un algorithme capable de répondre à des besoins précis à l'aide sur les données disponibles, (ii) s'assurer que le modèle appris par notre algorithme est capable de généraliser, i.e. d'être performant sur des données non connues lors de l'apprentissage du modèle mais issues de la même distribution. Pour répondre à ces deux points, nous commençons par introduire la notion de risque qui est au coeur du domaine de l'apprentissage machine. Nous poursuivons notre introduction en abordant la question du choix de l'espace des hypothèses que nous nous autorisons pour

résoudre le problème que l'on se pose. Bien définir l'espace des hypothèses va permettre à l'algorithme de se focaliser sur des hypothèses à la fois performante sur les données dites d'entraînements mais également sur les données futures. Nous présentons ensuite la théorie PAC-Bayésienne qui permet de dériver des garanties en généralisation sur l'hypothèse apprise vis-à-vis du risque que l'on a cherché à optimiser. Nous en développons trois grands axes différents qui permettent de répondre à la question de la performance en généralisation.

Après avoir introduit le formalisme de l'apprentissage statistique, nous présentons quelques modèles simples ainsi que des fonctions de coûts, attachées à la notion de risque, adaptées aux contextes de classification et de régression. Nous présentons également différents modèles de classification que nous serons amenés à rencontrer tout au long des différents chapitres. Finalement, nous terminons ce premier chapitre en introduisant le contexte de cette thèse: l'apprentissage dans un contexte déséquilibré, nous montrons que les outils classiques d'apprentissage s'avèrent inefficaces en négligeant l'information importante que représente la classe minoritaire. Une définition ainsi que différents exemples viennent souligner la difficulté que représente l'apprentissage machine dans un tel contexte.

B.2.2 Résumé du Chapitre 2

Dans ce chapitre, nous présentons l'apprentissage dans un contexte déséquilibré et plus précisément de la détection de fraudes. Nous commençons par discuter de la distinction entre anomalies et fraudes, bien que partageant une caractéristique commune, à savoir une sous-représentativité dans les données, il est important de les distinguer car les techniques permettant d'aborder la détection de fraude sont différentes de celle de la détection d'anomalies. Nous poursuivons la description de la notion de fraude en montrant que cette dernière est présente dans de nombreux contextes industriels. Pour aborder le problème de détection de fraudes et donc d'apprentissage dans un contexte déséquilibré, nous avons vu qu'il est nécessaire de définir de nouveaux outils plus approprié à ce contexte mais également de nouvelles techniques

Cela commence par la présentation de critères de mesures de performances qui vont permettre de mesurer la capacité qu'à un algorithme à effectuer un nombre minimal d'erreurs tout en étant capable de retrouver des exemples de la classe minoritaire. Nous présentons un choix non exhaustif de ces mesures de performance en s'attachant à souligner leurs différences.

Dans un second temps, nous nous concentrons sur les données et sur les techniques permettant de traiter le problème de déséquilibre : (i) les méthodes d'échantillonnage et (ii) la pondération des différentes classes. Nous verrons que les méthodes d'échantillonnage permettent de rééquilibrer les données par le biais de la création, duplication ou encore simple suppression de données. Les algorithmes d'échantillonnage sont nombreux et nous présentons les plus connus d'entre eux. Il est également possible d'affecter un poids aux différentes classes en présence. Ces deux méthodes ont pour objectif de contraindre les algorithmes à se concentrer sur les exemples de la classe minoritaire.

D'un point de vue algorithmique, ces derniers restent les mêmes que dans un contexte

d'apprentissage classique dit équilibré. Nous verrons que le simple fait de coupler des algorithmes classiques avec des méthodes d'échantillonnage permet de traiter le problème de déséquilibre. Nous présentons un large panel d'algorithmes présentés dans l'état de l'art et qui se proposent de résoudre le problème de l'apprentissage dans un contexte déséquilibré: l'apprentissage de représentation, l'utilisation de la notion de distance, le boosting, les SVM et leurs variantes, les réseaux de neurones, etc ...

Nous finissons ce chapitre en présentant les différents axes abordés dans les contributions de cette thèse.

B.2.3 Résumé du Chapitre 3

Ce premier chapitre des contributions de la thèse s'inscrit dans le cadre du développement d'une approche géométrique pour traiter le problème du déséquilibre. Les fraudes, si elles sont générées par une même personne et dans une courte période de temps, présentent des caractéristiques très similaires. D'un point de vue géométrique, elles partagent donc des caractéristiques semblables. Cela se traduit par la présence de petits groupes de fraudes que l'on peut retrouver dans tout l'espace.

Nous commençons par introduire un problème simple de détection d'anomalies qu'est celui de l'apprentissage de la plus petit sphère d'inclusion. Nous montrons que ce problème d'optimisation est très proche de celui des One Class SVM, un algorithme de classification non supervisé. Nous expliquons ensuite comment nous partons de ces modèles existants pour proposer un modèle qui sera construit autour de chaque fraude connue et qui sera en mesure des capturer de nouvelles fraudes localisées à proximité et prenant la forme de zone à risque. Afin de mieux appréhender la géométrie des données et de construire des zones à risque les plus grandes possibles, nous couplons notre méthode avec un apprentissage de métrique. L'apprentissage de métrique peut nécessiter de nombreuses opérations de calcul afin de garantir le côté semi défini positif de la matrice. Cependant, nous montrons qu'en cherchant à résoudre le problème dual, dont nous détaillons le développement, le caractère semi défini positif peut-être obtenu sans effort.

Une analyse théorique vient compléter cette première contribution en se basant sur la notion de stabilité. Cette analyse permet de garantir que la métrique M tend à se stabiliser lorsque le nombre de données augmente, ce qui permet de garantir une certaine performance du modèle sur de nouvelles données. En effet, nous montrons que nous sommes en mesure de contrôler le nombre de faux positifs de notre modèle. D'un point de vue expérimentale, nous montrons que la méthode proposée est capable d'être tout aussi performante que des algorithmes d'apprentissage classiques même couplés avec des stratégies d'échantillonnage.

B.2.4 Résumé du Chapitre 4

Le chapitre précédent a montré à quel point la distance à une donnée de la classe minoritaire (i.e. une fraude) était importante pour nous permettre de retrouver de nouvelles fraudes. Poursuivant dans cette même optique, nous proposons, dans ce chapitre, une version modifiée

de l'algorithme des plus proches voisins. Cette nouvelle contribution a pour but de modifier la façon dont est calculée la distance à un exemple positif afin de la rapprocher pour permettre de retrouver de nouvelles fraudes. Cependant, le modèle appris est contraint à ne pas générer trop d'erreurs, i.e. trop de faux positifs. Pour cela, les distance aux données positives sont pondérées par un facteur γ qui est optimisé de façon à maximiser la F-mesure.

Une étude théorique montre que, dans un contexte déséquilibré, ce paramètre γ peut-être choisi plus petit que 1 et que nous avons tout intérêt à rapprocher de toutes nouvelles données tests, les positifs existants. Des résultats expérimentaux sur des jeux de données issus de la communauté viennent illustrer ce résultat théorique.

Finalement, nous montrons qu'en dépit de sa simplicité, ce modèle est plus efficace que les méthodes de l'état de l'art reposants sur les plus proches voisins mais aussi sur l'apprentissage de représentation comme l'apprentissage de métriques. En outre, l'algorithme proposé s'avère particulièrement efficace lorsqu'il est combiné à des stratégies d'échantillonnage et montre que les points générés par ces méthodes d'échantillonnage s'apparentent majoritairement à du bruit plus qu'à une réelle information permettant d'améliorer les résultats en classification.

B.2.5 Résumé du Chapitre 5

Ce chapitre s'inscrit dans le deuxième axes des contributions de cette thèse : l'approche par pondération des erreurs. Ce cinquième chapitre de cette thèse présente des contributions à la fois théoriques et pratiques sur l'optimisation d'une mesure de performance dans un contexte déséquilibré la F-mesure. Cette mesure présente plusieurs aspects qui rendent son optimisation complexe: (i) elle est non linéaire contrairement à des mesures comme le taux d'erreur et (ii) elle est non convexe ce qui complexifie la tâche des algorithmes d'optimisation.

Cependant, nous montrons qu'il est possible de réduire le problème d'optimisation de la F-mesure en un problème de minimisation d'une version pondérée du risque empirique, i.e. en pondérant les erreurs effectués sur chaque classe. Nous montrons le lien entre les deux problèmes en établissant la propriété de pseudo-linéarité de la F-mesure. La principale difficulté reste alors la recherche des coûts optimaux, i.e. de la pondération optimale des erreurs. Pour cela, nous commençons par établir des résultats théoriques entre la F-mesure obtenue lorsqu'un modèle est appris avec une certaines pondération des erreurs et la plus grande F-mesure que l'on puisse obtenir si nous avions appris un modèle avec un autre paramètre de pondération des erreurs. La borne obtenue dépend de l'écart entre le paramètre courant de pondération et une autre valeur quelconque. Ce résultat permet également de mesurer l'écart à la F-mesure optimale. D'un point de vue géométrique, nous montrons que la borne obtenue peut être interprétée sous la forme d'un cône asymétrique dans l'espace défini par le paramètre de pondération et de la F-mesure. De premières expériences montrent que la borne obtenue est beaucoup plus fine que des résultats existants et permet de largement élagué l'espace de recherche des coûts optimaux.

Dans un deuxième temps, nous proposons une contribution algorithmique. En effet, on se propose d'utiliser cette interprétation géométrique pour construire un algorithme itératif

qui va permettre de rechercher les coûts optimaux permettant d'optimiser la F-mesure. L'algorithme proposé se montre (i) tout aussi efficace que d'autres algorithmes d'optimisation de la F-mesure et (ii) nécessite un faible nombre d'itérations pour converger.

B.2.6 Résumé du Chapitre 6

Dans ce dernier chapitre de la thèse, nous nous proposons d'appliquer des méthodes par pondération des erreurs à une application industrielle concrète : l'optimisation de la marge d'une enseigne de la grande distribution. Ce chapitre se propose de modifier le modèle de détection de fraudes employé par l'entreprise Blitz Business Services qui repose uniquement sur une performance en terme de classification des erreurs et se fonde la combinaison d'arbres de décisions.

Nous commençons par présenter la matrice de coûts que nous allons employer dans le cadre de notre travail. Cette dernière présente la particularité d'attribuer un poids à chaque terme de la matrice de confusion et non uniquement aux erreurs effectuées par l'algorithme. Ces poids ont également la particularité d'être exemple dépendants. Nous montrons ensuite comment inclure cette nouvelle information dans l'apprentissage d'arbres de décisions pour générer des modèles qui vont prendre en compte le montant du chèque, et donc d'une éventuelle fraude. Afin de pouvoir se comparer au modèle existant, plusieurs arbres sont construits et plusieurs règles d'attributions des étiquettes aux feuilles des arbres (ou encore à la combinaison des arbres) sont testées afin de pouvoir maximiser la marge de l'enseigne.

Dans une deuxième partie nous nous proposons d'employer un autre modèle à base d'arbres et fondé sur le gradient boosting. Après avoir rappelé la fonction de coût utilisée, nous présentons comment l'injecter dans ce type de modèle afin qu'elle soit directement optimisée et montrons que la seule optimisation d'une version exponentielle de notre loss est efficace pour résoudre le problème posé. Finalement nous montrons qu'il est nécessaire de fournir la gradient et le hessien de notre fonction objective pour notre problème d'optimisation.

Des expériences sur des données privées viennent montrer l'intérêt de cette modification à l'échelle de l'entreprise. Elles montrent qu'il est possible d'augmenter fortement la marge des enseignes tout en leur laissant une certaine flexibilité dans le choix des coefficients attribués à notre matrice de coûts. Flexibilité dont le client ne disposait avec l'ancien modèle et dont le problème d'optimisation se révélait, à leurs yeux, peu significatif.

B.3 Conclusions

Dans cette thèse, nous avons abordé le problème de l'apprentissage à partir de données déséquilibrées dans le contexte de la détection des fraudes. Le travail réalisé au cours de cette thèse se voulait éclectique et explorait différentes façons d'aborder le problème de l'apprentissage dans ce contexte non trivial. Nos contributions sont principalement réparties en deux catégories : les contributions géométriques et les contributions par pondérations des erreurs. Dans le premier cas, nous avons appris avec ME^2 une nouvelle représentation

des données autour de chaque fraude connue afin de construire des zones à risque. Nous modifions également la distance par rapport aux fraudes existantes (ou données positives) afin d'augmenter les cellules de Voronoi et donc la capacité du modèle à capturer de nouvelles fraudes. Dans ce dernier cas, nous attribuons des poids à chaque classe d'exemples afin d'optimiser une mesure de performance adaptée aux scénarios déséquilibrés, c'est-à-dire la F-mesure, mais aussi d'améliorer le modèle actuel d'algorithme de détection de fraude par chèque utilisé par la société Blitz. Les contributions de cette thèse sont à la fois algorithmiques et théoriques et ont donné lieu à la publication de plusieurs articles dont trois dans des conférences internationales (*ICTAI*, *AISTATS* et *IDA*) et une publications dans une revue internationale (*Pattern Recognition Letter*) ainsi que des communications dans une conférence française (CAp).

Résumé des contributions

Il existe de nombreuses techniques pour aborder le problème d'apprentissage dans un contexte déséquilibré. Cela peut impliquer (i) des méthodes d'échantillonnage pour équilibrer les différentes classes, (ii) l'apprentissage d'une nouvelle représentation des données dans laquelle le problème de classification est rendu plus aisé ; (iii) explorer l'optimisation de nouvelles fonctions de coûts et de nouveaux indicateurs de performance plus adaptés à ce type de contexte.

Une méthode d'apprentissage de métrique a constitué le premier axe de recherche de cette thèse. Nous avons étudié la notion de zones à risque autour des fraudes avérées et avons dérivé des garanties théoriques sur la projection linéaire apprise. Cette première contribution, essentiellement théorique, visait à mesurer la stabilité des zones de risque, c'est-à-dire les garanties qu'une opération non frauduleuse est susceptible d'être frauduleuse au moment du test. Ce travail a montré l'importance de la distance par rapport aux exemples positifs (ou aux fraudes existantes) lorsqu'on apprend à partir de données déséquilibrées. Cette observation a conduit à notre deuxième contribution pour laquelle nous avons proposé une distance ajustée pour l'algorithme du plus proche voisin. Une brève étude théorique a expliqué pourquoi il est nécessaire de comparer un exemple de test avec une donnée positive et d'en modifier la façon dont est calculée la distance si l'on veut augmenter les performances de notre modèle en terme de F-mesure. De nombreuses expériences ont viennent confirmer cette étude théorique.

L'utilisation d'approche par pondération des erreurs a constitué le cœur de notre deuxième axe de recherche. L'attribution d'un poids plus important aux exemples de la classe minoritaire s'est avérée pertinente dans un tel contexte. Cependant, nous avons montré que cela doit être fait avec beaucoup de prudence. Notre troisième contribution est dédiée à la conception d'un algorithme qui sélectionne automatiquement les poids optimaux à attribuer à chaque classe pour un ensemble donné de mesures de performance. L'algorithme proposé est basé sur une étude théorique approfondie fournissant des zones de pondération potentielles où il est possible de maximiser la F-mesure. Mais l'utilisation de méthodes sensibles aux coûts présente aussi des applications plus concrètes telles que la mise en œuvre et l'optimisation de fonctions

de coûts qui ont une réelle signification pour le monde industriel. Notre dernière contribution utilisait ce type de modèle pour améliorer le système de détection des fraudes de Blitz. Nous avons proposé une nouvelle fonction de coût pour optimiser les profits d'une enseigne de la Grande Distribution. Ce nouveau modèle tient compte du montant d'une transaction dans la prise de décision, ce qui n'était pas le cas jusqu'à présent dans l'entreprise.

Perspectives

Au-delà des perspectives présentées dans les différents chapitres, nous aimerions fournir quelques extensions possibles des contributions de cette thèse.

Dans le chapitre 3, une partie de la contribution était de fournir des garanties de généralisation sur les domaines de décision appris. Une première perspective consisterait à dériver directement des garanties de généralisation sur la F-mesure. La principale difficulté de la mesure F vient de sa non-linéarité. Nous souhaitons exploiter le lien entre la fonction d'erreur pondérée et la performance en terme de F-mesure développée au chapitre 5. En utilisant le cadre de stabilité uniforme (Bousquet and Elisseeff, 2002) sur la fonction d'erreur pondérée, nous visons à établir le lien entre la F-mesure observée pendant l'entraînement et la mesure F prévue au moment du test. En raison de sa non-linéarité, une autre possibilité pour étudier la performance en termes de F-mesure serait d'utiliser la théorie d'estimation bootstrap (Efron, 1979; Xu and Goodacre, 2018). Nous pourrions fournir des intervalles de confiance sur l'estimation et donc sur la F-mesure espérée au moment du test. Il pourrait également être intéressant d'étudier la convergence de l'algorithme proposé au chapitre 5 qui n'a été démontrée que dans la pratique.

En ce qui concerne l'entreprise, plusieurs améliorations peuvent être apportées au système de détection de fraudes actuellement utilisé. Le modèle est mis à jour chaque mois, en supposant que le changement dans la distribution de la fraude est significatif après un mois. Cependant, il serait intéressant de mettre en place une méthode de détection de dérive pour automatiser la mise à jour du modèle (Baena-Garcıa et al., 2006; de Lima Cabral and de Barros, 2018) qui pourrait potentiellement conduire à une amélioration du système utilisé. Cela ouvre la porte au développement de nouvelles méthodes d'adaptation de domaines.

La société travaille actuellement sur l'ajout de nouvelles informations dans le modèle qui peut être fourni par les enseignes : les *familles de produits* achetées par ses clients. Ces informations peuvent être utilisées pour détecter des comportements suspects car certaines de ces familles sont plus risquées que d'autres. Cependant, cette information est très parcimonieuse et prend la forme de très gros vecteurs de caractéristiques avec beaucoup de zéros. Pour tirer profit de cette information, nous devons étudier des méthodes d'apprentissage parcimonieuses. Dans le même but, nous cherchons également à développer de nouveaux types de modèles qui prennent en compte l'historique des clients de la Grande Enseigne. Cette information sur l'historique des clients est actuellement contenu dans des variables. Afin de s'affranchir de ces variables et de disposer d'une information plus "pure", une perspective intéressante consisterait à utiliser les Long Short Term Memory Networks (LSTM) (Hochre-

iter and Schmidhuber, 1997) afin de rendre compte plus précisément de l'historique.

Bibliography

V. M. Tikhomirov A. N. Kolmogorov. ε-entropy and ε-capacity of sets in function spaces. *Uspekhi Mat. Nauk*, 14:3–86, 1959. 29

Aisha Abdallah, Mohd Aizaini Maarof, and Anazida Zainal. Fraud detection system: A survey. *Journal of Network and Computer Applications*, 68:90–113, 2016. 44, 45

Charu C. Aggarwal. *Outlier Analysis*. Springer International Publishing, 2017. 43, 45, 59

Rehan Akbani, Stephen Kwek, and Nathalie Japkowicz. Applying support vector machines to imbalanced datasets. In *European conference on machine learning*, pages 39–50. Springer, 2004. 55

Mennatallah Amer, Markus Goldstein, and Slim Abdennadher. Enhancing one-class support vector machines for unsupervised anomaly detection. In *Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description*, pages 8–15. ACM, 2013. 56

Manuel Artís, Mercedes Ayuso, and Montserrat Guillén. Detection of automobile insurance fraud with discrete choice models and misclassified claims. *Journal of Risk and Insurance*, 69(3):325–340, 2002. 45

Ozlem Asian, Olcay Taner Yildiz, and Ethem Alpaydin. Calculating the VC-dimension of decision trees. In *2009 24th International Symposium on Computer and Information Sciences*. IEEE, sep 2009. 26

Meriem El Azami, Carole Lartizien, and Stéphane Canu. Robust outlier detection with L0-SVDD. In *22th European Symposium on Artificial Neural Network (ESANN-14)*, 2014. 82

Francis R Bach, David Heckerman, and Eric Horvitz. Considering cost asymmetry in learning classifiers. *Journal of Machine Learning Research*, 7(Aug):1713–1741, 2006. 55

Manuel Baena-Garcıa, José del Campo-Ávila, Raúl Fidalgo, Albert Bifet, R Gavalda, and R Morales-Bueno. Early drift detection method. In *Fourth international workshop on knowledge discovery from data streams*, volume 6, pages 77–86, 2006. 144, 167

Alejandro Correa Bahnsen, Aleksandar Stojanovic, Djamila Aouada, and Björn Ottersten. Cost sensitive credit card fraud detection using bayes minimum risk. In *Proceedings of the 2013 12th International Conference on Machine Learning and Applications - Volume 01*, ICMLA '13, pages 333–338, Washington, DC, USA, 2013. IEEE Computer Society. ISBN 978-0-7695-5144-9. doi: 10.1109/ICMLA.2013.68. URL `http://dx.doi.org/10.1109/ICMLA.2013.68`. 53, 54

Alejandro Correa Bahnsen, Aleksandar Stojanovic, Djamila Aouada, and Björn Ottersten. *Improving Credit Card Fraud Detection with Calibrated Probabilities*, pages 677–685. 2014. 47

Ricardo Barandela, José Salvador Sánchez, Vicente García, and E. Rangel. Strategies for learning in class imbalance problems. *Pattern Recognition*, 36(3):849–851, 2003. 83, 92, 98

Peter L. Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, March 2003. ISSN 1532-4435. 26, 27

Kevin Bascol, Rémi Emonet, Elisa Fromont, Amaury Habrard, Guillaume Metzler, and Marc Sebban. Un algorithme d'optimisation de la F-mesure par pondération des erreurs de classification. In *Conférence francophone sur l'Apprentissage Automatique (CAp-18)*, 2018.

Kevin Bascol, Rémi Emonet, Elisa Fromont, Amaury Habrard, Guillaume Metzler, and Marc Sebban. From cost-sensitive classification to tight F-measure bounds. In *In Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics (AISTATS-19)*, 2019.

Gustavo EAPA Batista, Ana LC Bazzan, and Maria Carolina Monard. Balancing training data for automated annotation of keywords: a case study. In *WOB*, pages 10–18, 2003. 53

Rukshan Batuwita and Vasile Palade. Fsvm-cil: fuzzy support vector machines for class imbalance learning. *IEEE Transactions on Fuzzy Systems*, 18(3):558–571, 2010. 55

Rukshan Batuwita and Vasile Palade. Class imbalance learning methods for support vector machines. *Imbalanced learning: Foundations, algorithms, and applications*, 83, 2013. 55

Aseem Behl, CV Jawahar, and M Pawan Kumar. Optimizing average precision using weakly supervised data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1011–1018, 2014. 49

Aurélien Bellet, Amaury Habrard, and Marc Sebban. A survey on metric learning for feature vectors and structured data. *arXiv preprint arXiv:1306.6709*, 2013. 59

Aurélien Bellet, Amaury Habrard, and Marc Sebban. Metric learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 9(1):1–151, 2015. 60, 91

Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, September 1975. ISSN 0001-0782. doi: 10.1145/361002.361007. URL `http://doi.acm.org/10.1145/361002.361007`. 33

Alina Beygelzimer, Elad Hazan, Satyen Kale, and Haipeng Luo. Online gradient boosting. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2458–2466. 2015. 58

Richard J Bolton and David J Hand. Statistical fraud detection: A review. *Statistical science*, pages 235–249, 2002. 47

Richard J Bolton, David J Hand, et al. Unsupervised profiling methods for fraud detection. *Credit Scoring and Credit Control VII*, pages 235–255, 2001. 44

Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152. ACM, 1992. 30, 33

M. El Boujnouni, M. Jedra, and N. Zahid. New decision function for support vector data description. In *Snd Int. Conf. on Innovative Computing Technology (INTECH 2012)*, pages 305–310, 2012. 67, 68

Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002. ISSN 1532-4435. 27, 28, 76, 144, 167

Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. *Introduction to Statistical Learning Theory*, pages 169–207. Springer Berlin Heidelberg, 2004. 129

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004. ISBN 0521833787. 35, 54, 109

Paula Branco, Luís Torgo, and Rita P. Ribeiro. A survey of predictive modeling on imbalanced domains. *ACM Comput. Surv.*, 49(2):31:1–31:50, August 2016. 43, 46, 49

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001. 57, 134

Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. *Classification and regression trees*. The Wadsworth statistics/probability series. Wadsworth and Brooks/Cole Advanced Books and Software, Monterey, CA, 1984. 38

Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *ACM sigmod record*, volume 29, pages 93–104. ACM, 2000. 59

Andreas Buja, Werner Stuetzle, and Yi Shen. Loss functions for binary class probability estimation and classification: Structure and applications," manuscript, available at www-stat.wharton.upenn.edu/ buja, 2005. 136, 137

Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 475–482. Springer, 2009. 52

Róbert Busa-Fekete, Balázs Szörényi, Krzysztof Dembczynski, and Eyke Hüllermeier. Online f-measure optimization. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 595–603. Curran Associates, Inc., 2015. 48, 108

Alberto Cambini and Laura Martein. *Generalized Convexity and Optimization: Theory and Applications*. Lecture Notes in Economics and Mathematical Systems 616. Springer-Verlag Berlin Heidelberg, 1 edition, 2009. 112, 113

Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 2009. 45, 108

Olivier Chapelle. Training a support vector machine in the primal. *Neural Computation*, 19: 1155–1178, 2007. 35

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16 (1):321–357, June 2002. 52, 82, 93, 99

Nitesh V. Chawla, Nathalie Japkowicz, and Aleksander Kotcz. Editorial: Special issue on learning from imbalanced data sets. *SIGKDD Explor. Newsl.*, 6(1):1–6, June 2004. ISSN 1931-0145. 49, 50

Sanjay Chawla and Aristides Gionis. k-means–: A unified approach to clustering and outlier detection. In *Proceedings of the 2013 SIAM International Conference on Data Mining*, pages 189–197. SIAM, 2013. 43

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794. ACM, 2016. ISBN 978-1-4503-4232-2. 134, 135

Tianqi Chen and Tong He. Xgboost: extreme gradient boosting. 58

David A Cieslak and Nitesh V Chawla. Learning decision trees for unbalanced data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 241–256. Springer, 2008. 57

Gilles Cohen, Mélanie Hilario, Hugo Sax, Stéphane Hugonnet, and Antoine Geissbuhler. Learning from imbalanced data in surveillance of nosocomial infection. *Artificial intelligence in medicine*, 37(1):7–18, 2006. 48

Alejandro Correa Bahnsen, Sergio Villegas, Djamila Aouada, and Bjorn Ottersten. Fraud detection by stacking cost-sensitive decision trees. *Data Science for Cyber-Security (DSCS), London 25-27 September*, 2017. 53, 57, 132, 133

Corinna Cortes and Mehryar Mohri. Auc optimization vs. error rate minimization. In *Advances in neural information processing systems*, pages 313–320, 2004. 48

T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, January 1967. 31, 32, 91

David R. Cox. The regression analysis of binary sequences (with discussion). *Journal of the Royal Statistical Society*, 20:215–242, 1958. 31, 36

J.S. Cramer. The origins of logistic regression. *SSRN Electronic Journal*, 2003. 36

Andrea Dal Pozzolo, Giacomo Boracchi, Olivier Caelen, Cesare Alippi, and Gianluca Bontempi. Credit card fraud detection and concept-drift adaptation with delayed supervised information. In *Neural Networks (IJCNN), 2015 International Joint Conference on*, pages 1–8. IEEE, 2015. 44

Jason V Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th international conference on Machine learning*, pages 209–216. ACM, 2007. 91, 92

Danilo Rafael de Lima Cabral and Roberto Souto Maior de Barros. Concept drift detection based on fisher's exact test. *Information Sciences*, 442:220–234, 2018. 144, 167

Stijn Decubber, Thomas Mortier, Krzysztof Dembczynski, and Willem Waegeman. Deep f-measure maximization in multi-label classification: A comparative study. page 16, 2018. 108

Maria José del Jesus, Alberto Fernández, Salvador Garcıa, and Francisco Herrera. A first study on the use of fuzzy rule based classification systems for problems with imbalanced data sets. In *Symposium on Fuzzy Systems in Computer Science (FSCS06). Magdeburg, Germany*, pages 63–72, 2006. 51

Krzysztof Dembczyński, Wojciech Kotłowski, Oluwasanmi Koyejo, and Nagarajan Natarajan. Consistency analysis for binary classification revisited. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 961–969, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. 108

Krzysztof J Dembczynski, Willem Waegeman, Weiwei Cheng, and Eyke Hüllermeier. An exact algorithm for f-measure maximization. In *NIPS*, 2011. 108

David J Dittman, Taghi M Khoshgoftaar, Randall Wald, and Amri Napolitano. Comparison of data sampling approaches for imbalanced bioinformatics data. In *FLAIRS Conference*, 2014. 50

Georgios Douzas and Fernando Bacao. Effective data generation for imbalanced learning using conditional generative adversarial networks. *Expert Systems with applications*, 91: 464–471, 2018. 61

Lixiang Duan, Mengyun Xie, Tangbo Bai, and Jinjiang Wang. A new support vector data description method for machinery fault diagnosis with unbalanced datasets. *Expert Systems with Applications*, 64:239 – 246, 2016. 68

S. A. Dudani. The distance-weighted k-nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(4):325–327, April 1976. 33, 92, 98

B. Efron. Bootstrap methods: Another look at the jackknife. *Ann. Statist.*, 7(1):1–26, 01 1979. doi: 10.1214/aos/1176344552. URL `https://doi.org/10.1214/aos/1176344552`. 144, 167

Charles Elkan. The foundations of cost-sensitive learning. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence - Volume 2*, pages 973–978, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. 54, 136

Charles Elkan. Using the triangle inequality to accelerate k-means. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, ICML'03, pages 147–153. AAAI Press, 2003. 33

D. Jack Elzinga and Donald W. Hearn. The minimum covering sphere problem. *Management Science*, 19(1):96–104, 1972. 67

Andrew Estabrooks, Taeho Jo, and Nathalie Japkowicz. A multiple resampling method for learning from imbalanced data sets. *Computational intelligence*, 20(1):18–36, 2004. 50

Wei Fan, Salvatore J Stolfo, Junxin Zhang, and Philip K Chan. Adacost: misclassification cost-sensitive boosting. In *Icml*, pages 97–105, 1999. 56

Lin Feng, Huibing Wang, Bo Jin, Haohao Li, Mingliang Xue, and Le Wang. Learning a distance metric by balancing kl-divergence for imbalanced datasets. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2018. 60

Alberto Fernández, Salvador Garcia, Francisco Herrera, and Nitesh V Chawla. Smote for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary. *Journal of Artificial Intelligence Research*, 61:863–905, 2018. 53, 93

César Ferri, José Hernández-Orallo, and R Modroiu. An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, 30(1):27–38, 2009. 49

Jordan Frery, Amaury Habrard, Marc Sebban, Olivier Caelen, and Liyun He-Guelton. Efficient top rank optimization with gradient boosting for supervised anomaly detection. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 20–35. Springer, 2017. 49

Yoav Freund and Robert E. Schapire. A short introduction to boosting. In *In Proceedings of the Sixteenth IJCAI*, pages 1401–1406. Morgan Kaufmann, 1999. 58

Yoav Freund, Robert Schapire, and Naoki Abe. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612, 1999. 56

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, 28(2):337–407, 04 2000. 31, 137

Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000. 58, 137

Mikel Galar, Alberto Fernandez, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera. A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(4):463–484, 2012. 57

E. A. Garcia and H. He. Learning from imbalanced data. *IEEE Transactions on Knowledge & Data Engineering*, 21:1263–1284, 12 2008. ISSN 1041-4347. 49

V. García, J. S. Sánchez, and R. A. Mollineda. On the effectiveness of preprocessing methods when dealing with different levels of class imbalance. *Know.-Based Syst.*, 25(1):13–21, February 2012. 41

Sunder Gee. *Fraud and fraud detection: a data analytics approach*. John Wiley & Sons, 2014. 47

Marc G. Genton. Classes of kernels for machine learning: A statistics perspective. *Journal of Machine Learning Research*, 2:299–312, March 2002. 36

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 61, 90

Zahra Hajizadeh, Mohammad Taheri, and Mansoor Zolghadri Jahromi. Nearest neighbor classification with locally weighted distance for imbalanced data. *International Journal of Computer and Communication Engineering*, 3(2):81, 2014. 92

Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International Conference on Intelligent Computing*, pages 878–887. Springer, 2005. 52, 93, 99

James A Hanley and Barbara J McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, 1982. 48

P. Hart. The condensed nearest neighbor rule. *IEEE Transactions on Information Theory*, 14(3):515–516, May 1968. 33, 50

Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pages 1322–1328. IEEE, 2008. 53, 93, 99

Q. P. He and J. Wang. Principal component based k-nearest-neighbor rule for semiconductor process fault detection. In *2008 American Control Conference*, pages 1606–1611, June 2008. 33

Katherine A Heller, Krysta M Svore, Angelos D Keromytis, and Salvatore J Stolfo. One class support vector machines for detecting anomalous windows registry accesses. In *Proc. of the workshop on Data Mining for Computer Security*, volume 9, 2003. 56

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9 (8):1735–1780, 1997. 145, 167

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963. 25

M. Jansche. Maximum expected f-measure training of logistic regression models. In *EMNLP*, 2005. 108

László A Jeni, Jeffrey F Cohn, and Fernando De La Torre. Facing imbalanced data–recommendations for the use of performance metrics. In *Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on*, pages 245–251. IEEE, 2013. 49

J. L. W. V. Jensen. Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta Mathematica*, 30:175–193, 1906. 40

T. Joachims. A support vector method for multivariate performance measures. In *ICML*, 2005. 108

D. Kang, J. Park, and J. C. Principe. Binary classification based on svdd projection and nearest neighbors. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6, July 2010. 68

Taghi M Khoshgoftaar, Moiz Golawala, and Jason Van Hulse. An empirical study of learning from imbalanced data using random forest. In *Tools with Artificial Intelligence, 2007. ICTAI 2007. 19th IEEE international conference on*, volume 2, pages 310–317. IEEE, 2007. 57

Edwin M Knox and Raymond T Ng. Algorithms for mining distancebased outliers in large datasets. In *Proceedings of the international conference on very large data bases*, pages 392–403. Citeseer, 1998. 58

Vladimir Koltchinskii and Dmitriy Panchenko. Rademacher processes and bounding the risk of function learning. In Evarist Giné, David M. Mason, and Jon A. Wellner, editors, *High Dimensional Probability II*, pages 443–457. Birkhäuser Boston, 2000. 26

Aryeh Kontorovich and Roi Weiss. A bayes consistent 1-nn classifier. In *Artificial Intelligence and Statistics*, pages 480–488, 2015. 91

Aryeh Kontorovich, Sivan Sabato, and Ruth Urner. Active nearest-neighbor learning in metric spaces. In *Advances in Neural Information Processing Systems*, pages 856–864, 2016. 91

Ender Konukoglu and Melanie Ganz. Approximate false positive rate control in selection frequency for random forest. *CoRR*, abs/1410.2838, 2014. 46

Yufeng Kou, Chang-Tien Lu, Sirirat Sirwongwattana, and Yo-Ping Huang. Survey of fraud detection techniques. In *Networking, sensing and control, 2004 IEEE international conference on*, volume 2, pages 749–754. IEEE, 2004. 44, 45

Oluwasanmi O Koyejo, Nagarajan Natarajan, Pradeep K Ravikumar, and Inderjit S Dhillon. Consistent binary classification with generalized performance metrics. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS-14)*, pages 2744–2752. Curran Associates, Inc., 2014. 108, 109, 125, 127, 130

Miroslav Kubat, Stan Matwin, et al. Addressing the curse of imbalanced training sets: one-sided selection. In *Icml*, volume 97, pages 179–186. Nashville, USA, 1997. 50

Jorma Laurikkala. Improving identification of difficult small classes by balancing class distribution. In *Conference on Artificial Intelligence in Medicine in Europe*, pages 63–66. Springer, 2001. 50

Trung Le, Dat Tran, and Wanli Ma. Fuzzy multi-sphere support vector data description. In *17th Pacific-Asia Conference (PAKDD), Part II*, pages 570–581. Springer, 2013. 68

Bertrand Lebichot, Yann-Aël Le Borgne, Liyun He-Guelton, Frédéric Oblé, and Gianluca Bontempi. Deep-learning domain adaptation techniques for credit cards fraud detection. In *INNS Big Data and Deep Learning conference*, pages 78–88. Springer, 2019. 60

Guillaume Lemaître, Fernando Nogueira, and Christos K Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *The Journal of Machine Learning Research*, 18(1):559–563, 2017. 99

Ping Li, Christopher J. C. Burges, and Qiang Wu. Mcrank: Learning to rank using multiple classification and gradient boosting. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, NIPS'07, pages 897–904, 2007. 58

Chun-Fu Lin and Sheng-De Wang. Fuzzy support vector machines. *IEEE transactions on neural networks*, 13(2):464–471, 2002. 55

Wei Liu and Sanjay Chawla. Class confidence weighted knn algorithms for imbalanced data sets. In Joshua Zhexue Huang, Longbing Cao, and Jaideep Srivastava, editors, *Advances in Knowledge Discovery and Data Mining*, pages 345–356, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. 33

Xu-Ying Liu and Zhi-Hua Zhou. The influence of class imbalance on cost-sensitive learning: An empirical study. In *null*, pages 970–974. IEEE, 2006. 54

Yi Liu and Y.F. Zheng. Minimum enclosing and maximum excluding machine for pattern description and discrimination. In *18th IEEE International Conference on Pattern Recognition (ICPR06)*, 2006. 67

Susan Lomax and Sunil Vadera. A survey of cost-sensitive decision tree induction algorithms. *ACM Computing Surveys (CSUR)*, 45(2):16, 2013. 57

Victoria López, Alberto Fernández, Jose G Moreno-Torres, and Francisco Herrera. Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. open problems on intrinsic data characteristics. *Expert Systems with Applications*, 39(7):6585–6608, 2012. 54

Victoria Lopez, Alberto Fernandez, Salvador Garcia, Vasile Palade, and Francisco Herrera. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences*, 250:113 – 141, 2013. 108

Ulrike von Luxburg and Olivier Bousquet. Distance-based classification with lipschitz functions. *Journal of Machine Learning Research*, 5(Jun):669–695, 2004. 91

Larry M Manevitz and Malik Yousef. One-class svms for document classification. *Journal of machine Learning research*, 2(Dec):139–154, 2001. 56

Hamed Masnadi-Shirazi and Nuno Vasconcelos. Risk minimization, probability elicitation, and cost-sensitive svms. In *ICML*, pages 759–766. Citeseer, 2010. 55

Colin McDiarmid. *On the method of bounded differences*, pages 148–188. London Mathematical Society Lecture Note Series. Cambridge University Press, 1989. 25

J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 209(441-458):415–446, jan 1909. 35

Charles E Metz. Basic principles of roc analysis. In *Seminars in nuclear medicine*, volume 8, pages 283–298. Elsevier, 1978. 48

Guillaume Metzler, Xavier Badiche, Brahim Belkasmi, Elisa Fromont, Amaury Habrard, and Marc Sebban. Apprentissage de sphères maximales d'exclusion avec garanties théoriques. In *Conférence francophone sur l'Apprentissage Automatique (CAp-17)*, 2017.

Guillaume Metzler, Xavier Badiche, Brahim Belkasmi, Elisa Fromont, Amaury Habrard, and Marc Sebban. Learning maximum excluding ellipsoids from imbalanced data with theoretical guarantees. In *Pattern Recognition Letters*, volume 112, pages 310–316. Elsevier BV, 2018a.

Guillaume Metzler, Xavier Badiche, Brahim Belkasmi, Elisa Fromont, Amaury Habrard, and Marc Sebban. Tree-based cost sensitive methods for fraud detection. In *International Symposium on Intelligent Data Analysis(IDA- 2018)*, pages 213–224. Springer International Publishing, 2018b.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. The MIT Press, 2012. 31

Ajinkya More. Survey of resampling techniques for improving classification performance in unbalanced datasets. *arXiv preprint arXiv:1608.06048*, 2016. 50

Katharina Morik, Peter Brockhausen, and Thorsten Joachims. Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, pages 268–277, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc. ISBN 1-55860-612-2. URL `http://dl.acm.org/citation.cfm?id=645528.657612`. 55

David R Musicant, Vipin Kumar, Aysel Ozgur, et al. Optimizing f-measure with support vector machines. In *FLAIRS*, 2003. 108

Harikrishna Narasimhan, Purushottam Kar, and Prateek Jain. Optimizing non-decomposable performance measures: A tale of two classes. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 199–208, 2015a. 108

Harikrishna Narasimhan, Harish Ramaswamy, Aadirupa Saha, and Shivani Agarwal. Consistent multiclass algorithms for complex performance measures. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, volume 37 of *Proceedings of Machine Learning Research*, pages 2398–2407, Lille, France, 07–09 Jul 2015b. PMLR. 54, 108, 109, 126, 127, 130

Nikolaos Nikolaou, Narayanan Edakunni, Meelis Kull, Peter Flach, and Gavin Brown. Cost-sensitive boosting algorithms: Do we really need them? *Machine Learning*, 104(2):359–384, Sep 2016. 138

Sarwat Nizamani, Nasrullah Memon, Mathies Glasdam, and Dong Duong Nguyen. Detection of fraudulent emails by employing advanced feature abundance. *Egyptian Informatics Journal*, 15(3):169–174, 2014. 44

Shameem Puthiya Parambath, Nicolas Usunier, and Yves Grandvalet. Optimizing f-measures by cost-sensitive classification. In *Advances in Neural Information Processing Systems (NIPS-14)*, pages 2123–2131, 2014. 54, 108, 109, 110, 113, 115, 121, 124, 125, 126, 127, 128, 129, 130, 153

Hamid Parvin, Behrouz Minaei-Bidgoli, and Hamid Alinejad-Rokny. A new imbalanced learning and dictions tree method for breast cancer diagnosis. *Journal of Bionanoscience*, 7(6): 673–678, 2013. 57

Eric J. Pauwels and Onkar Ambekar. One class classification for anomaly detection: Support vector data description revisited. In *Industrial Conference on Data Mining*, pages 25–39, 2011. 66, 82

Clifton Phua, Vincent Lee, Kate Smith, and Ross Gayler. A comprehensive survey of data mining-based fraud detection research. *arXiv preprint arXiv:1009.6119*, 2010. 43

S.K. Shevade P.M. Chinta, P. Balamurugan and M.N. Murty. Optimizing f-measure with non-convex loss and sparse linear classifiers. In *IJCNN*, 2013. 108

J. R. Quinlan. Induction of decision trees. *Mach. Learn.*, 1(1):81–106, March 1986. 38

J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993. ISBN 1-55860-238-0. 70

T. Rapcsák. On pseudolinear functions. *European Journal of Operational Research*, 50(3): 353–360, feb 1991. 111

Gunnar Rätsch, Takashi Onoda, and K-R Müller. Soft margins for adaboost. *Machine learning*, 42(3):287–320, 2001. 56

C. J. Van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 2nd edition, 1979. ISBN 0408709294. 46

L. Rokach and O. Maimon. Top-down induction of decision trees classifiers - a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 35 (4):476–487, Nov 2005. 38

S. R. Safavian and D. Landgrebe. A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(3):660–674, May 1991. 38

Yusuf Sahin, Serol Bulkan, and Ekrem Duman. A cost-sensitive decision tree approach for fraud detection. *Expert System Application*, 40(15):5916–5923, November 2013. 134

Mehdi SM Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing generative models via precision and recall. In *Advances in Neural Information Processing Systems*, pages 5228–5237, 2018. 90

Mayu Sakurada and Takehisa Yairi. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, page 4. ACM, 2014. 60

Mark Sanderson. Word sense disambiguation and information retrieval. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '94, pages 142–151, New York, NY, USA, 1994. Springer-Verlag New York, Inc. URL `http://dl.acm.org/citation.cfm?id=188490.188548`. 46

Jörg Schiller. The impact of insurance fraud detection systems. *Journal of Risk and Insurance*, 73(3):421–438, 2006. 45

Bernhard Scholkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001. 55, 56, 67

Clayton Scott. Calibrated asymmetric surrogate losses. *Electron. J. Statist.*, 6:958–992, 2012. doi: 10.1214/12-EJS699. URL `https://doi.org/10.1214/12-EJS699`. 109

John Shawe-Taylor, Keith Howker, and Peter Burge. Detection of fraud in mobile telecommunications. *Information Security Technical Report*, 4(1):16 – 28, 1999. ISSN 1363-4127. doi: https://doi.org/10.1016/S1363-4127(99)80003-X. URL `http://www.sciencedirect.com/science/article/pii/S136341279980003X`. 45

J.J. Sylvester. A question in the geometry of situation. *Quaterly Journal of Pure and Apllied Mathemtatics*, 1857. 66

Yuchun Tang, Yan-Qing Zhang, Nitesh V Chawla, and Sven Krasser. Svms modeling for highly imbalanced classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(1):281–288, 2009. 55

David M. J. Tax and Robert P. W. Duin. Data domain description using support vectors. In *Proceedings of the European Symposium on Artificial Neural Networks*, pages 251–256, 1999. 66

David M. J. Tax and Robert P. W. Duin. Support vector data description. *Machine Learning Journal*, 54(1):45–66, 2004. 66, 67, 82

Nguyen Thai-Nghe, Zeno Gantner, and Lars Schmidt-Thieme. Cost-sensitive learning methods for imbalanced data. In *Neural Networks (IJCNN), The 2010 International Joint Conference on*, pages 1–8. IEEE, 2010. 55

Ivan Tomek. Two modifications of cnn. *IEEE Trans. Systems, Man and Cybernetics*, 6: 769–772, 1976. 50, 93, 99

Luís Torgo and Elsa Lopes. Utility-based fraud detection. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, pages 1517–1522, 2011. 46

Luis Torgo and Rita Ribeiro. Utility-based regression. In *Knowledge Discovery in Databases: PKDD 2007*, pages 597–604. Springer Berlin Heidelberg, 2007. 46

Luís Torgo, Rita P. Ribeiro, Bernhard Pfahringer, and Paula Branco. Smote for regression. In Luís Correia, Luís Paulo Reis, and José Cascalho, editors, *Progress in Artificial Intelligence*, pages 378–389, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. 43

Luís Torgo, Paula Branco, Rita P. Ribeiro, and Bernhard Pfahringer. Resampling strategies for regression. *Expert Systems*, 32(3):465–476, aug 2014. doi: 10.1111/exsy.12081. URL `https://doi.org/10.1111/exsy.12081`. 43

Reba A. Umberger, Linda A. Hatfield, and Patricia M. Speck. Understanding negative predictive value of diagnostic tests used in clinical practice. *Dimensions of Critical Care Nursing*, 36(1):22–29, 2017. 46

L. G. Valiant. A theory of the learnable. *Communication of the ACM*, 27(11):1134–1142, 1984. 25

Aad W. van der Vaart and Jon A. Wellner. *Weak Convergence and Empirical Processes*. Springer New York, 1996. 29

C. J. van Rijsbergen. Further experiments with hierarchic clustering in document retrieval. *Information Storage and Retrieval*, 10(1):1–14, 1974. 108

V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, 1971. 25, 26

V. Vapnik and A. Chervonenkis. Necessary and sufficient conditions for the uniform convergence of means to their expectations. *Theory of Probability & Its Applications*, 26(3): 532–553, 1982. 25

Vladimir Vapnik and Corinna Cortes. Support-vector networks. *Machine Learning*, 20:273–297, 1995. 30, 33

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin, Heidelberg, 1995. 23

Nakul Verma and Kristin Branson. Sample complexity of learning mahalanobis distance metrics. In *Advances in Neural Information Processing Systems (NIPS-15)*, 2015. 81

Rémi Viola, Rémi Emonet, Amaury Habrard, Guillaume Metzler, Sébastien Riou, and Marc Sebban. An adjusted nearest neighbor algorithm maximizing the f-measure from imbalanced data. In *In Proceedings of the 31st International Conference on Tools with Artificial Intelligence (ICTAI-2019)*, 2019a.

Rémi Viola, Rémi Emonet, Amaury Habrard, Guillaume Metzler, Sébastien Riou, and Marc Sebban. Une version corrigée de l'algorithme des plus proches voisins pour l'optimisation de la f-mesure dans un contexte déséquilibré. In *Conférence francophone sur l'Apprentissage Automatique (CAp-19)*, 2019b.

Benjamin X Wang and Nathalie Japkowicz. Boosting support vector machines for imbalanced data sets. *Knowledge and information systems*, 25(1):1–20, 2010. 56

Haixun Wang, Wei Fan, Philip S Yu, and Jiawei Han. Mining concept-drifting data streams using ensemble classifiers. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 226–235. AcM, 2003. 44

Nan Wang, Xibin Zhao, Yu Jiang, Yue Gao, and KLISS BNRist. Iterative metric learning for imbalance data classification. In *IJCAI*, pages 2805–2811, 2018. 60

Senzhang Wang, Zhoujun Li, Wenhan Chao, and Qinghua Cao. Applying adaptive oversampling technique based on data density and cost-sensitive svm to imbalanced learning. In *Neural Networks (IJCNN), The 2012 International Joint Conference on*, pages 1–8. IEEE, 2012. 55

Zhe Wang, Daqi Gao, and Zhisong Pan. An effective support vector data description with relevant metric learning. In *7th International Symposium on Neural Networks (ISNN), Part II*, pages 42–51. Springer, 2010. 68

Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244, June 2009. ISSN 1532-4435. 33, 59, 60, 91, 92, 98

Graham Williams, Rohan Baxter, Hongxing He, Simon Hawkins, and Lifang Gu. A comparative study of rnn for outlier detection in data mining. In *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*, pages 709–712. IEEE, 2002. 60

Dennis L. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-2(3):408–421, jul 1972. doi: 10.1109/tsmc.1972.4309137. URL `https://doi.org/10.1109/tsmc.1972.4309137`. 50, 93, 99

Xiaoyun Wu and Rohini K Srihari. New\'{\i}-support vector machines and their sequential minimal optimization. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 824–831, 2003. 55

Huan Xu and Shie Mannor. Robustness and generalization. In *Conference on Learning Theory (COLT-10)*, pages 503–515, 2010. 29, 30

Huan Xu and Shie Mannor. Robustness and generalization. *Machine Learning*, 86(3):391–423, Mar 2012. 29

Yun Xu and Royston Goodacre. On splitting training and validation set: A comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning. *Journal of Analysis and Testing*, 2(3):249–262, 2018. 144, 167

Chan-Yun Yang, Jr-Syu Yang, and Jian-Jun Wang. Margin calibration in svm class-imbalanced learning. *Neurocomputing*, 73(1-3):397–411, 2009. 54

Nan Ye, Kian Ming Adam Chai, Wee Sun Lee, and Hai Leong Chieu. Optimizing f-measure: A tale of two approaches. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, 2012. 108

Mohd Izhan Mohd Yusoff, Ibrahim Mohamed, and Mohd Rizam Abu Bakar. Fraud detection in telecommunication industry using gaussian mixed model. In *Research and Innovation in Information Systems (ICRIIS), 2013 International Conference on*, pages 27–32. IEEE, 2013. 45

Bernard Zenko, L jupco Todorovski, and Saso Dzeroski. A comparison of stacking with meta decision trees to bagging, boosting, and stacking with other methods. In *icdm*, page 669. IEEE, 2001. 57

Ke Zhang, Marcus Hutter, and Huidong Jin. A new local distance-based outlier detection approach for scattered real-world data. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 813–822. Springer, 2009. 59

L. Zhang, Xingning Lu, Bangjun Wang, and Shuping He. Similarity learning based on multiple support vector data description. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7, July 2015. 68

Ming-Jie Zhao, Narayanan Unny Edakunni, Adam Craig Pocock, and Gavin Brown. Beyond fano's inequality: bounds on the optimal f-score, ber, and cost-sensitive risk and their implications. *Journal of Machine Learning Research*, 14:1033–1090, 2013. 48, 108

Chong Zhou and Randy C Paffenroth. Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 665–674. ACM, 2017. 60

Abstract. Fraud and anomaly detection, or more generally learning in an imbalanced context, is a task very often encountered in industrial applications. Detecting these anomalies is a major challenge in today's society due to its potential economic consequences. BLITZ Business Services is confronted with this type of problem in the context of the fight against check fraud. These frauds represent 0.4% of the transactions but millions of euros of losses per year for its customers. Dealing with fraud data, and more generally with imbalanced data, is a complex task for most current learning algorithms because of the under-representation of frauds over non-frauds. The techniques are also as diverse and varied as the nature of the frauds encountered and range from sampling strategy, representation learning, optimization of measures appropriate to an imbalanced context or the construction of classification algorithms combining the advantages of several of the former. This thesis is intended to be eclectic, in the same way as the techniques present in the state of the art and is divided into two main axes: (i) a so-called geometric approach in which we propose metric learning algorithms for data classification and (ii) a cost-sensitive approach that we use for both theoretical and practical purposes. Our first contribution is based on learning local models around known frauds in order to build risky areas. It is based on the assumption that a new fraud is very likely to occur in the neighborhood of a known fraud. A theoretical study accompanies this algorithm to ensure that the number of false positives generated by the algorithm remains controlled. In our second contribution, we propose a version the $k-$Nearest Neighbors algorithm adapted to the imbalanced context. In this study, we propose to analyze how the distance from a new query to a fraud should be modified in order to optimize a measure adapted to this context: the F-measure, through cross validation. This measure is at the heart of our third contribution, which is mainly theoretical. We propose to derive a bound on the optimal F-measure from the pseudo-linearity property of this measure, the errors made by the hypotheses learned and a cost-sensitivity approach. The theoretical bounds obtained are then used to build an iterative algorithm for optimizing the F-measure, algorithm that is at least as efficient as its competitors. Our fourth and final contribution is industrial and aims to combine the use of tree-based models and cost sensitivity to improve BLITZ existing system by offering a profit optimization system for its customers.

Résumé. La détection de fraudes et d'anomalies, ou plus généralement l'apprentissage dans un contexte déséquilibré, est une tâche très souvent rencontrée dans de nombreuses applications industrielles. Détecter ces anomalies revêt un enjeu majeur dans notre société actuelle de par ses conséquences économiques. La société BLITZ Business Services est confrontée à ce type de problématique dans le cadre de la lutte contre la fraude par chèques. Ces fraudes représentent 0.4% des transactions pour ses clients mais des millions d'euros de pertes par an. Les données de fraudes sont difficiles pour la plupart des algorithmes actuels de par cette sous-représentativité des fraudes par rapport aux non-fraudes. Les techniques d'analyse sont aussi diverses et variées que la nature des fraudes rencontrées et vont de la stratégie de ré-échantillonnage, d'apprentissage de représentation, l'optimisation de mesures appropriées à un contexte déséquilibré ou encore la construction d'algorithmes de classification combinant plusieurs autres algorithmes. Cette thèse se veut éclectique, à l'image des techniques présentes dans l'état de l'art, et se divise en deux grands axes : (i) une approche dite géométrique dans laquelle nous proposons des algorithmes d'apprentissage de métrique pour la classification de données et (ii) une approche par sensibilité aux coûts que nous utilisons à la fois dans un but théorique mais aussi pratique. Notre première contribution repose sur l'apprentissage de modèles locaux autour de fraudes avérées afin de construire des zones à risque. Elle part du postulat qu'une nouvelle fraude a de très grandes chances d'apparaître à proximité d'une fraude connue. Une étude théorique accompagne cet algorithme permettant d'assurer que le nombre de faux positifs générés par l'algorithme reste contrôlé. Dans notre deuxième contribution, nous proposons une version de l'algorithme des k plus proches voisins plus adaptée au contexte déséquilibré. Dans cette étude, essentiellement expérimentale, nous nous proposons d'étudier la façon dont doit être modifiée la distance d'un nouvel exemple à une fraude afin d'optimiser une mesure adaptée à ce contexte : la F-mesure, par le biais de la validation croisée. Cette mesure est au centre de notre troisième contribution qui se veut principalement théorique. Nous proposons de dériver une borne sur la F-mesure optimale à partir de la propriété de pseudo-linéarité de cette mesure, des erreurs effectuées par l'hypothèse apprise et d'une approche par sensibilité aux coûts. Ces bornes théoriques obtenues sont ensuite utilisées pour construire un algorithme itératif d'optimisation de la F-mesure, algorithme qui est tout aussi performant que ces concurrents. Notre quatrième et dernière contribution est industrielle et a pour but de combiner l'utilisation de modèles à base d'arbres et de sensibilité aux coûts pour améliorer le système existant de la société BLITZ en proposant un système d'optimisation des bénéfices de ses clients.