MLFP: Un algorithme d'apprentissage de métrique pour la classification de données déséquilibrées

Rémi Viola[1,2], Rémi Emonet [1], Amaury Habrard[1], Guillaume Metzler[1], and Marc Sebban[1]

[1]Laboratoire Hubert Curien UMR 5516, Université de Lyon, Université Jean-Monnet, F-42023, Saint-Etienne, France
[2]Direction Générale des Finances Publiques, Ministère de l'Économie et des Finances, France

Résumé

Apprendre à partir de données déséquilibrées reste une tâche complexe en apprentissage, tant sur le plan théorique que pratique. Dans cet article, nous abordons cette problématique en utilisant une stratégie basée sur l'apprentissage de métrique. Contrairement aux méthodes se basant sur la même approche, notre algorithme apprend une nouvelle métrique qui ne sera utilisée que lorsqu'une (nouvelle) donnée est comparée à un exemple d'apprentissage de la classe minoritaire (ou classe positive). D'un point de vue géométrique, cela revient à rapprocher artificiellement des exemples positifs de cette (nouvelle) donnée sans modifier les caractéristiques de la classe majoritaire. La stratégie mise en œuvre permet d'étendre les frontières de décisions autour des données positives. En terme de performance, cela se traduit par une meilleure F-mesure, critère de performance très souvent employé dans ce contexte, par rapport aux algorithmes de l'état de l'art. Au-delà de cette contribution algorithmique, notre article présente une étude théorique basée sur la *stabilité uniforme*. Cette étude nous donne des garanties de généralisation sur les taux de faux positifs et de faux négatifs. Les expériences, effectuées sur plusieurs ensembles de données déséquilibrées, montrent l'efficacité de notre méthode par rapport aux algorithmes d'apprentissage de métrique existants. Avec la méthode proposée, nous sommes en mesures de rivaliser voire d'obtenir de meilleures performances qu'avec des algorithmes spécifiquement dédiés au traitement de données déséquilibrées.

Mots-clef : Imbalanced Binary Classification, Metric Learning, Nearest-Neighbors, Theoretical Guarantees

1 Introduction

Fraud detection in bank or insurance applications [AMZ16, Sch06], and anomaly identification for medical diagnosis [Agg17] are some societal challenges requiring to address the problem of learning from highly imbalanced data. When dealing with such a setting, one has to face two major issues : (i) the scarcity of the class of interest, only composed of a few positive data, which limits the efficiency of standard margin-based loss functions ; (ii) the scattering of positive examples in the total mass of the training data, which makes the estimation of local densities much more complicated than in balanced scenarios. Several solutions have been proposed in the literature to address these two problems. Most of them consist in applying sampling strategies which aim to balance the dataset by reducing the number of negative examples and/or creating new synthetic positive data [SBK$^+$18, PTMH19]. On the other hand, one can resort to cost-sensitive algorithms [KHB$^+$17] which assign a weight to each class (or even to each example) so that the classifier can focus better on the minority class. Other strategies include the use of ensemble methods [WJS$^+$17, FHSHG18] or the specific adaptation of existing approaches such as deep learning [HLCLT16, DCK18] or kernel methods [MLPC15, DML$^+$18, ZWZ$^+$19].

In this paper, we address the problem of learning from imbalanced data from a metric learning perspective [BHS13, K$^+$13]. Learning a metric specifically designed for the application at hand may present several advantages in the context of imbalanced datasets : (i) the metric can be learned under semantic constraints allowing us to expand the decision boundaries around the positives ; (ii) this framework enables to design optimization problems based on the geometry of the data without suffering from the is-

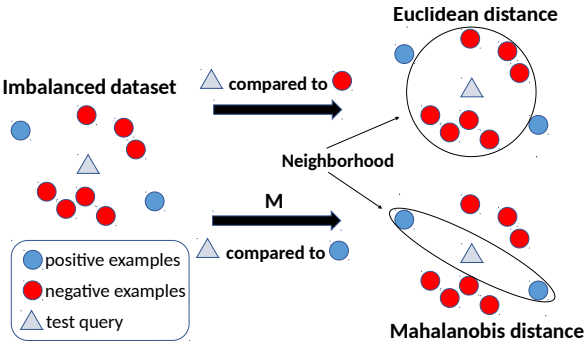

FIGURE 1 – Intuition behind our method **MLFP** : a PSD matrix $\mathbf{M}$ is optimized under constraints, and is used only when a test query is compared to a positive example. The distance to the negative examples is kept unchanged. This allows the learned metric to expand the decision boundaries around the positives and thus to capture more examples of the class of interest.

sues of standard accuracy-based loss functions (*e.g.*, hinge loss for SVMs, exponential loss for boosting, logistic loss for logistic regression) ; (iii) metric learning is a nice setting to derive theoretical guarantees on the learned transformation [BHS15]. Surprisingly, despite these interesting features, metric learning has not received much attention to address the problem of learning from imbalanced data (see, *e.g.*, the recent papers [FWJ+18], [WZJ+18] and [GMHS19]). The goal of this paper is to bridge this gap from both an algorithmic and a theoretical perspective. As illustrated in Figure 1, we propose the algorithm **MLFP** that optimizes a linear transformation (via a *Positive Semi Definite* (PSD) matrix $\mathbf{M}$ of a Mahalanobis distance) only when a test query is compared to a minority training example. A single metric $\mathbf{M}$ is learned for the whole space taking the geometry of the data into account. Unlike the standard metric learning algorithms (see, *e.g.*, **LMNN** [WS09] or **ITML** [DKJ+07]), our method boils down to artificially bringing positive examples closer to the query without challenging the features of the negatives. This has a direct impact on the decision boundaries around the positives allowing us to capture more examples of the class of interest yielding a better F-Measure (see Section 3 for a formal definition). By using the uniform stability framework, we derive theoretical guarantees on the learned matrix $\mathbf{M}$ showing the actual capability of **MLFP** to control the false positive and false negative rates.

The paper is organized as follows. In Section 2, we report some related work on metric learning for imbalanced data classification. Section 3 is dedicated to the presentation of our metric learning algorithm **MLFP**. Section 4 presents a theoretical analysis using the uniform stability framework and Section 5 illustrates the performance of **MLFP** compared to state-of-the-art algorithms.

2 Related work

Most of the metric learning algorithms (see [BHS13, K+13] for a survey) are based on the optimization of the Mahalanobis distance between two points $\mathbf{x}_i$ and $\mathbf{x}_j \in \mathbb{R}^q$:

$$d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j)^2 = (\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{M} (\mathbf{x}_i - \mathbf{x}_j),$$

where $\mathbf{M}$ is a $q \times q$ *Positive Semi Definite* matrix. One can express $\mathbf{M}$ as $\mathbf{L}^T\mathbf{L}$ where $\mathbf{L}$ is a $r \times q$ matrix where r is the rank of $\mathbf{M}$. Thus, this distance can be seen as the Euclidean distance in a new feature space $\mathbf{Lx}$.

A well-known representative of this family of algorithms is the *Large Margin Nearest Neighbor* (**LMNN**) [WS09]. For each example of a training set of size m, the learned metric $\mathbf{M}$ aims to bring closer the neighbors of the same class (called target neighbors) while pushing away the examples of other classes (the impostors). While the number of constraints is in the order of km^2, the authors proposed an efficient subgradient descent algorithm which benefits from the fact that many of these constraints are trivially satisfied. This algorithm has been shown to be very efficient and to scale well with large datasets. However, it is worth noticing that **LMNN** is not designed to take into account some imbalance in the data. Indeed, the similarity constraints constructed from pairs of examples of the same class do not make any difference between the positive and negative examples. Therefore, in imbalanced scenarios, **LMNN** is prone to focus on the majority class and thus is subject to miss the positive examples.

This remark also holds for *Information Theoretic Metric Learning* (**ITML**) [DKJ+07]. We can also cite *Geometric Mean Metric Learning* (**GMML**) [ZHS16] which learn a matrix $\mathbf{M}$ to compute the distance between the similar examples and the matrix $\mathbf{M}^{-1}$ to compute the dissimilar ones. The idea is that if $\mathbf{M}$ brings similar examples close to each other(i.e. all the eigenvalues are less than one), then $\mathbf{M}^{-1}$ will push dissimilar examples away. Build on a geometric intuition, they learn the metric with a convex optimization problem which have a closed form solution by seing the latter as an optimization

problem on the Riemannian manifold of SPD matrices.

The first attempts to address the problem of learning a metric from imbalanced datasets have been proposed very recently. [WZJ+18] introduce an *iterative metric learning* algorithm (**IML**) that aims to define a stable neighborhood used to predict the label of a new test data. The method repeats two main steps : (i) the learning of a linear transformation, *e.g.*, by using **LMNN**, and (ii) a training sample selection given a test example. The procedure is repeated until stabilization of the neighborhood. By repeating the process several times, **IML** is able to locally separate positives from negatives. However, the main issue comes from the algorithmic complexity of the method, which requires to apply **LMNN** and to update the pairs used for the training process at each iteration. Another approach to learn metrics from imbalanced datasets has been recently proposed [GMHS19]. In their *Imbalanced Metric Learning* algorithm (**ImbML**), the authors take into account the nature of the pairwise constraints by using two different sub-losses, one for each label, weighted according to the number of positive and negative examples respectively. This intuitive and natural way to proceed prevents the algorithm from favoring the majority class. However, we will see that applying the learned metric $\mathbf{M}$ to all examples is not necessary, focusing only on the minority class appears to be much more efficient and allows us notably to better control the false negatives. Finally, [FWJ+18] introduce **DMBK** for *Distance Metric by Balancing KL-divergence*. This algorithm resorts to the KL-divergence to represent normalized between-class divergences. Combined with a geometric mean, **DMBK** is able to make these divergences balanced. Thus, this method makes particularly sense for dealing with multi-class classification tasks, but its interest is reduced when used in a binary setting.

Beyond the algorithmic limitations of the previous state-of-the-art algorithms, note that none of them comes with classification guarantees. In this paper, we address this problem by studying the capability of **MLFP** to optimize a metric $\mathbf{M}$ which provides a good compromise between (i) expanding the decision boundaries around the positives which enables to reduce the false negative rate at test time (one of the main issues faced in imbalanced learning) ; (ii) controlling this expansion to prevent the algorithm from detecting too many false alarms (*e.g.* anomalies, frauds), represented by the false positive rate. The theoretical results take the form of guarantees on the learned metric using the uniform stability framework [BE02] which measures the stability of the output of the algorithm when the training set is subject to slight changes.

3 Metric Learning for Imbalanced Data

In this section, we present our algorithm **MLFP**, for *Metric Learning from Few Positives*. In the following, we denote by $S = \{\mathbf{z}_i = (\mathbf{x}_i, y_i)\}_{i=1}^m$ the set of m training examples drawn *i.i.d.* from an unknown joint distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, where $\mathbf{x}_i \in \mathcal{X}$ (here $\mathcal{X} = \mathbb{R}^q$) is a feature vector and $y_i \in \mathcal{Y}$ (here $\mathcal{Y} = \{-1, +1\}$) corresponds to its associated label. The label $+1$ is used to denote the positive or the minority class. We further note $S = S_+ \cup S_-$ with S_+ the set of m_+ positive examples and S_- the set of m_- negative examples, such that $m = m_+ + m_-$.

3.1 Problem Formulation

Mahalanobis distance-based metric learning algorithms typically minimize a function of the form :

$$F(\mathbf{M}) = \frac{1}{m^2} \sum_{(z,z') \in S^2} \ell(\mathbf{M}, \mathbf{z}, \mathbf{z}') + \mu \, Reg(\mathbf{M})$$

where ℓ is a loss function depending on $\mathbf{M}$ and pairs of examples $(\mathbf{z}, \mathbf{z}')$ (some algorithms also use triplets of points), and Reg is a regularization term on $\mathbf{M}$. The parameter μ controls the compromise between the two terms. Note here that each example has the same importance whatever its label. In imbalanced settings, the minority examples correspond to the class of interest (*e.g.*, frauds in bank transactions). Unfortunately, with the above formulation, any margin-based loss function ℓ will be prone to focus on the majority class and will therefore miss most of the positive instances. In order to avoid the pitfall of classic metric learning algorithms that are prone to focus on the majority class, we propose to give more importance to the minority class composed of the positive instances.

In our approach, we use the Euclidean distance when comparing a query point to a majority-class example. The originality comes from the use of an optimized Mahalanobis distance when comparing a query to a minority-class sample. The objective of this strategy is to formulate a metric learning problem leading to a classifier (a kNN here) which is accurate on both classes even in an imbalanced scenario.

Our algorithm **MLFP** tries to control the false positive (FP) and false negative (FN) rates thanks to the

following constrained optimization problem :

$$\min_{\mathbf{M}\in\mathbb{S}^+} \frac{1}{m^3}\left((1-\alpha)\sum_{\substack{(\mathbf{x}_i,\mathbf{x}_j,\mathbf{x}_k)\\ y_i=y_j=1\neq y_k}} \ell_{\mathrm{FN}}(\mathbf{M},\mathbf{z}_i,\mathbf{z}_j,\mathbf{z}_k) + \alpha \sum_{\substack{(\mathbf{x}_i,\mathbf{x}_j,\mathbf{x}_k)\\ y_i=y_j=-1\neq y_k}} \ell_{\mathrm{FP}}(\mathbf{M},\mathbf{z}_i,\mathbf{z}_j,\mathbf{z}_k)\right) + \mu\|\mathbf{M}-\mathbf{I}\|_{\mathcal{F}}^2,$$

$$\text{such that } \lambda_{\max}(\mathbf{M}) \leq 1. \tag{1}$$

where $\mathbb{S}^+$ is the set of PSD matrices, $\lambda_{\max}(\mathbf{M})$ is the largest eigenvalue of the PSD matrix $\mathbf{M}$, ℓ_{FN} and ℓ_{FP} are defined by :

$$\ell_{\mathrm{FN}}(\mathbf{M},\mathbf{z}_i,\mathbf{z}_j,\mathbf{z}_k) = [1-c+d_{\mathbf{M}}(\mathbf{x}_i,\mathbf{x}_j)^2 - d(\mathbf{x}_i,\mathbf{x}_k)^2]_+,$$
$$\ell_{\mathrm{FP}}(\mathbf{M},\mathbf{z}_i,\mathbf{z}_j,\mathbf{z}_k) = [1-c+d(\mathbf{x}_i,\mathbf{x}_j)^2 - d_{\mathbf{M}}(\mathbf{x}_i,\mathbf{x}_k)^2]_+,$$

where α is the positive rate $\frac{m_+}{m}$ and $\mu\|\mathbf{M}-\mathbf{I}\|_{\mathcal{F}}^2$ is a regularization term which penalizes a large deviation from the Euclidean distance. The hyper-parameter c controls the margin we want to preserve between pairs of dissimilar examples according to the Euclidean space and the learned one.

Problem (1) is composed of two terms where triplets are involved. Unlike standard metric learning algorithms, our method takes into account both the Euclidean distance d and the metric learned d_M. More precisely : the first term ℓ_{FN} aims to gather the minority class examples with respect to the learned metric such that the distance between two positives (using $\mathbf{M}$) is less than the distance to a negative example (using the Euclidean distance). This subloss can be seen as a way to prevent the model from generating false negatives (FN). The second term ℓ_{FP} works in a similar manner. The only difference lies in the fact that the query $\mathbf{x}_i$ is a negative example. Thus, we learn $\mathbf{M}$ such that the positive queries $\mathbf{x}_k$ are not bringing too close to $\mathbf{x}_i$, *i.e.* the Euclidean distance between two negatives $\mathbf{x}_i$ and $\mathbf{x}_j$ (with respect to the Euclidean distance) is lower than the distance between $\mathbf{x}_i$ and $\mathbf{x}_k$ (with respect to $\mathbf{M}$). This subloss can be seen as a way to prevent the model from generating false positives (FP).

Both FN and FP are important terms to optimize measures that are more suited to deal with imbalanced settings, such as the F-Measure [Rij79] defined as follows :

$$F_1 = \frac{2(m_+ - FN)}{2m_+ - FN + FP}.$$

Minimizing the F-Measure boils down to finding a good trade-off between FP and FN. However, in a highly imbalanced setting, where m_+ is very low, missing only a few positives leads to a dramatic decrease of the F-Measure. The constraint over the largest eigenvalue $\lambda_{\max}(\mathbf{M})$ of the learned matrix $\mathbf{M}$ aims to pay more attention to the positive class. In the next section, we provide a formal explanation of its use.

3.2 On the Impact of the Constraint

We study the impact of the $\lambda_{\max}(\mathbf{M})$ value on both FN and FP and, thus the influence of the constraint of our optimization problem.

Proposition 1. *Let $\mathbb{P}[FN_{\mathbf{M}}(\mathbf{x})]$ (resp. $\mathbb{P}[FP_{\mathbf{M}}(\mathbf{x})]$) be the probability of a positive query (resp. a negative query) $\mathbf{x}$ of being a false negative (resp. a false positive) using the 1NN algorithm with the learned matrix $\mathbf{M}$ and $\mathbb{P}[FN(\mathbf{x})]$ (resp. $\mathbb{P}[FP(\mathbf{x})]$) the same probability using the Euclidean distance.*
Then, if $\lambda_{\max}(\mathbf{M}) \leq 1$, we have :

$$\mathbb{P}[FN_{\mathbf{M}}(\mathbf{x})] \leq \mathbb{P}[FN(\mathbf{x})] \text{ and } \mathbb{P}[FP_{\mathbf{M}}(\mathbf{x})] \geq \mathbb{P}[FP(\mathbf{x})].$$

Sketch of proof. Let ε be the distance from $\mathbf{x}$ to its nearest neighbor $N_{\mathbf{x}}$. The example $\mathbf{x}$ is a false negative if $N_{\mathbf{x}} \in S_-$, that is, all positives $\mathbf{x}' \in S_+$ are outside an ellipsoid $\mathcal{E}_{\varepsilon,\mathbf{M}^{-1}}(\mathbf{x})$, defined by ε and $\mathbf{M}$. Therefore, we have :

$$\mathbb{P}[FN_{\mathbf{M}}(\mathbf{x})] = (1 - \mathbb{P}\left[\mathbf{x}' \in \mathcal{E}_{\varepsilon,\mathbf{M}^{-1}}(\mathbf{x})\right])^{m_+}. \tag{2}$$

When the Euclidean distance is used, we deal with a standard sphere $\mathcal{S}_\varepsilon$ of radius ε, and we get :

$$\mathbb{P}[FN(\mathbf{x})] = (1 - \mathbb{P}\left[\mathbf{x}' \in \mathcal{S}_\varepsilon(\mathbf{x})\right])^{m_+}. \tag{3}$$

Having $\lambda_{\max}(\mathbf{M}) \leq 1$ implies Eq. (2) $\leq$ Eq. (3). Indeed $\lambda_{\max}(\mathbf{M}) \leq 1$ implies that the sphere $\mathcal{S}_\varepsilon$ is included in the ellipsoid $\mathcal{E}_{\varepsilon,\mathbf{M}^{-1}}$ as illustrated in Figure 2. By this choice, we expand the decision boundaries around positives and thus capture more minority class examples. Using a similar scheme, we can prove the second inequality of Proposition 1. When $\mathbf{x}$ is negative and $N_{\mathbf{x}} \in S_+$, we have

$$\mathbb{P}[FP_{\mathbf{M}}(\mathbf{x})] = (1 - \mathbb{P}\left[\mathbf{x}' \in \mathcal{E}_{\varepsilon,\mathbf{M}^{-1}}(\mathbf{x})\right])^{m_-}, \tag{4}$$

and

$$\mathbb{P}[FP(\mathbf{x})] = (1 - \mathbb{P}\left[\mathbf{x}' \in \mathcal{S}_\varepsilon(\mathbf{x})\right])^{m_-}. \tag{5}$$

□

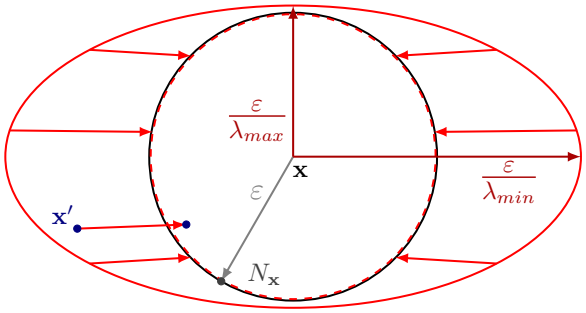

FIGURE 2 – Illustration of the constraint $\lambda_{\max}(\mathbf{M}) \leq 1$. Without learning the matrix $\mathbf{M}$, the Euclidean distance is used both to compare a query $\mathbf{x}$ to a negative $N_{\mathbf{x}}$ and to a positive $\mathbf{x}'$. The isodistance curves are thus spherical and identical (one in solid black for $N_{\mathbf{x}}$, one in dashed red for $\mathbf{x}'$). By learning the matrix $\mathbf{M}$, we virtually change the distance of the query to the positive examples. The isodistance curves for the positives are now ellipses, like the one represented in red. In the example, the positive $\mathbf{x}'$, that is outside the sphere, is inside the ellipse and will thus be considered closer, with the constraint $\lambda_{\max}(\mathbf{M}) \leq 1$, than the negative $N_{\mathbf{x}}$ that lies on the black sphere. With this same constraint, we are sure that the ellipse is enclosing the circle (*i.e.* $\frac{\varepsilon}{\lambda_{max}} \geq \varepsilon$) and so that all positives will be brought closer to the query. In the end, this constraint ensures that we increase the influence of the positives and thus leads to the decrease of FN.

From Equations (2) and (4), we can note that they are both exponentially decreasing *w.r.t.* to the number of positives and negatives respectively. However, in imbalanced scenarios, the number of negatives is supposed to be much higher than the number of positives. Thus, the probability of having a false positive is decreasing faster than the probability of having a false negative. We then choose to learn a matrix $\mathbf{M}$ under the constraint $\lambda_{\max}(\mathbf{M}) \leq 1$, so that our algorithm will focus first on reducing FN. An illustration of the impact of this constraint in terms of decision boundaries is shown in Figure 3. The experiments in Section 5 will confirm that the use of this constraint is very relevant from an F-Measure perspective and is able to reduce the number of FN at test time.

4 Theoretical Analysis

In this section, we provide generalization guarantees about the learned metric $\mathbf{M}$ using the uniform stability framework [BE02] adapted to metric learning [BHS15]. Then, we use this result to derive classification guarantees over a 1-Nearest Neighbor (1NN) classifier making use of this metric. Note that the whole study is conducted under the constraint $\lambda_{\max}(\mathbf{M}) \leq 1$ as used in Problem (1) .

First, we denote by ℓ the weighted combination of ℓ_{FN} and ℓ_{FP} as defined in Problem (1) and F_S the objective function to optimize over the training set $S = \{\mathbf{z}_i\}_{i=1}^m$. We have

$$F_S = \frac{1}{m^3} \sum_{i,j,k=1}^{m} \ell(\mathbf{M}, (\mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k)) + \mu \|\mathbf{M} - \mathbf{I}\|_{\mathcal{F}}^2.$$

Let $\mathcal{R}_S$ be the associated empirical risk over S defined as

$$\mathcal{R}_S = \frac{1}{m^3} \sum_{i,j,k=1}^{m} \ell(\mathbf{M}, (\mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k)),$$

and $\mathcal{R}$ be the corresponding expected true risk defined as

$$\mathcal{R} = \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[\mathcal{R}_S] = \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}\left[\frac{1}{m^3} \sum_{i,j,k=1}^{m} \ell(\mathbf{M}, (\mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k))\right] = \mathop{\mathbb{E}}_{\mathbf{z},\mathbf{z}',\mathbf{z}''\sim\mathcal{D}}[\ell(\mathbf{M}, (\mathbf{z}, \mathbf{z}', \mathbf{z}''))] .$$

The last equality is due to the *i.i.d.* aspect of the expectation. We also suppose that for all $\mathbf{x}$, we have $\|\mathbf{x}\| \leq K$.

4.1 Uniform Stability

Intuitively, an algorithm is stable if its output, in terms of loss, does not change significantly under a small modification of the training sample. The supremum of this change must be bounded in $\mathcal{O}(1/m)$. In this section, we assume that the space is bounded, i.e. there exsists $0 < K < \infty$ such that, for all $\mathbf{x} \in \mathcal{X}$, $\|x\| \leq K$.

Definition 1. *A learning algorithm $\mathcal{A}$ has a uniform stability in $\frac{\kappa}{m}$ with respect to a loss function ℓ and parameter set θ, with κ a positive constant if :*

$$\forall S,\ \forall i,\ 1 \leq i \leq m,\ \sup_Z |\ell(\theta_S, Z) - \ell(\theta_{S^i}, Z)| \leq \frac{\kappa}{m},$$

where S is a learning sample of size m, $Z = (\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3) = ((\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_3, y_3))$ is a triplet of labeled examples, 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 $\mathbf{z}_i$ from S by another example $\mathbf{z}'_i$ independent from S and drawn from $\mathcal{D}$. Finally, $\ell(\theta_S, Z)$ is the loss suffered at Z.

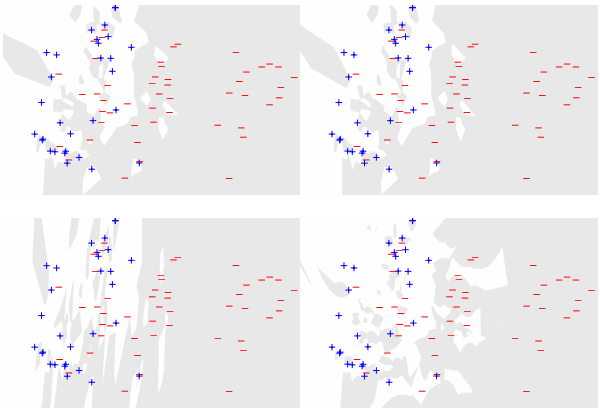

FIGURE 3 – Illustration of the impact of the constraint $\lambda_{\max}(\mathbf{M}) \leq 1$ in **MLFP** (bottom right) compared to k**NN** (top left), **LMNN** (top right), **ImbML** (bottom left) on the *autompg* dataset with a 1NN classifier. We perform a PCA, keeping the two most relevant dimensions, and plot the test set on a mesh grid of the space. In light grey (*resp.* white), areas classified as negative (*resp.* positive).

In this definition, S^i represents the notion of small modification of the training sample. The next definition aims to study the evolution of the loss function according to the considered triplets Z and Z'.

Definition 2. *A loss function ℓ is said to be γ-admissible, w.r.t. the distance metric $\mathbf{M}$ if (i) it is convex w.r.t. its first argument and (ii) if the following condition holds :*

$$\forall Z, Z' \quad |\ell(\mathbf{M}, Z) - \ell(\mathbf{M}, Z')| \leq \gamma,$$

where $Z = (\mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k)$ and $Z' = (\mathbf{z}'_i, \mathbf{z}'_j, \mathbf{z}'_k)$ are two triplets from a sample S and drawn from $\mathcal{D}$.

From the two above definitions, we can state the following generalization bound.

Theorem 1. *Let $\delta > 0$ and $m > 2$. Let S be a sample of m randomly selected training examples. Let $\mathbf{M}$ be the matrix learned from Problem (1) which has a uniform stability in $\frac{\kappa}{m}$. The loss function ℓ as defined above is γ-admissible. With probability $1-\delta$, the following bound on the true risk $\mathcal{R}$ of ℓ holds :*

$$\mathcal{R} \leq \mathcal{R}_S + 2\frac{\kappa}{m} + (2\kappa + 2\gamma)\sqrt{\frac{\ln(2/\delta)}{2m}},$$

where

$$\kappa = \frac{12}{\mu} \times ((1-\alpha)K^2)^2 \text{ and } \gamma = (1-\alpha)(1-c+4K^2).$$

Proof 1. *(Sktech of proof) The proof is adapted from [BHS15] but requires a direct extension to the use of triplets of examples instead of pairs. Notice that ℓ is convex since it is linear w.r.t. $\mathbf{M}$. The proof resorts to three main steps : (i) we show that the loss function is γ-admissible, (ii) then we prove that ℓ is k-lipschitz, (iii) then we can bound the regularization term and apply Hoeffding's inequality to get the result. All the details are given in the Supplementary Material.*

The derived bound provides guarantees on the generalization performances of the learned metric on the distribution $\mathcal{D}$ *w.r.t.* to the loss ℓ. We now make use of this bound to provide classification guarantees of a 1NN making use of the learned metric $\mathbf{M}$.

4.2 Classification Guarantees

We derive here generalization guarantees on the FP and FN rates for a 1NN classifier making use of the metric $\mathbf{M}$ learned by **MLFP**. Let S be the learning sample of size m used by a nearest-neighbor classifier. Let us define the empirical risks for FP and FN :

$$\mathcal{R}_{FP}(S) = \mathbb{E}_{\mathbf{z}=(\mathbf{x},y)\sim D} \mathbb{1}_{\{d_{\mathbf{M}}(\mathbf{x},\mathbf{x}_p)^2 \leq d(\mathbf{x},\mathbf{x}_n)^2\}} \times \mathbb{1}_{\{y=-1\}}.$$

where $\mathbf{x}_p, \mathbf{x}_n \in S$ are respectively the nearest positive and negative neighbors of $\mathbf{x}$ in S. Symmetrically, we have :

$$\mathcal{R}_{FN}(S) = \mathbb{E}_{\mathbf{z}=(\mathbf{x},y)\sim D} \mathbb{1}_{\{d(\mathbf{x},\mathbf{x}_n)^2 \leq d_{\mathbf{M}}(\mathbf{x},\mathbf{x}_p)^2\}} \times \mathbb{1}_{\{y=1\}}.$$

We consider then the expected true risks averaged over all the training samples of size m :

$$\mathcal{R}_{FP} = \mathbb{E}_{S\sim D^m} \mathcal{R}_{FP}(S) \text{ and } \mathcal{R}_{FN} = \mathbb{E}_{S\sim D^m} \mathcal{R}_{FN}(S).$$

We can no introduce our main result.

Theorem 2. *Let $\delta > 0$ and $m > 0$. Let S be a training sample of size m i.i.d. from a distribution D, z a new instance i.i.d. from D, and let $\mathbf{M}$ be the learned matrix from Problem (1) which has a uniform stability in $\frac{\kappa}{m}$ with respect to the loss ℓ. Considering that the loss function ℓ is γ-admissible, let us denote by $\mathcal{R}_S$ its empirical risk. With probability $1-\delta$, we have the following bounds for the FP and FN rates :*

$$\mathcal{R}_{FP} \leq \frac{1}{\alpha}\left[\mathcal{R}_{S\cup\{z\}} + \frac{2\kappa}{m+1} + (2\kappa+2\gamma)\sqrt{\frac{\ln(2/\delta)}{2(m+1)}}\right],$$

$$\mathcal{R}_{FN} \leq \frac{1}{1-\alpha}\left[\mathcal{R}_{S\cup\{z\}} + \frac{2\kappa}{m+1} + (2\kappa+2\gamma)\sqrt{\frac{\ln(2/\delta)}{2(m+1)}}\right].$$

Proof 2. *(Sktech of proof) The proof is based on the following two steps :* (i) *we bound the indicator function of FP and FN by the hinge loss ;* (ii) *we make the true risk* $\mathcal{R}$ *to appear on the right hand side and benefit from Theorem 1 to get the result.*

By comparing these two bounds, one can observe that when the class imbalance becomes important, *i.e.* when α takes a low value, the guarantees on the FN rate become better than the guarantees on FP. This result provides a theoretical confirmation that our approach - thanks to the constraint $\lambda_{\max}(\mathbf{M}) \leq 1$ - is able to focus more on reducing FN. An illustration of this phenomenon will be shown in the next section.

5 Experiments

In this section, we compare **MLFP** to other metric learning algorithms, focusing on (highly) imbalanced datasets. For all experiments, we use a 3-Nearest Neighbor classifier as done in both [WS09] and [WZJ+18].

5.1 Experimental Setup

We use several public datasets from the UCI [1] and KEEL [2] repositories. These datasets are diverse in terms of imbalance ratio (IR, number of majority examples per positive example), dimension, number of examples, as shown in Table 1. In addition, we use eight datasets provided by the French Ministry of the Economy and Finances (DGFiP). It corresponds to the tax returns of French companies and are used for fraud detection. These frauds may correspond to overvalued charges, voluntary reductions in profits or, local or international, VAT fraud. As the DGFiP can only control a small part of the 3,000,000 companies, it is essential to optimise the selection of these companies in order to reduce errors as much as possible. Considering the control process, it is less costly for the DGFiP to quickly control a non-fraudulent company than to let a big fraudster pass. It is therefore important for it to reduce the number of FN, even if it means increasing the number of FP a little. All the datasets are standardized by substracting the mean and dividing by the standard deviation.

We use the F-Measure as the performance criterion to compare the different methods.

Furthermore, 80% of the dataset is randomly selected in order to train the model and 20% to test it. The different hyper-parameters are tuned with a 10-fold-cross-validation over the training set. The sampling of the test set is repeated 5 times and we report the average results in terms of F-Measure (F_1).

TABLE 1 – Properties of the considered datasets, sorted by imbalance ratio (IR) which corresponds to the ratio m_-/m_+. The first part refers to the public datasets, the second one describes the private DGFiP datasets.

DATASETS	SIZE	DIM	IR	DATASETS	SIZE	DIM	IR
BALANCE	625	4	1.2	LIBRAS	360	90	14
AUTOMPG	392	7	1.7	REDWINEQUALITY4	1599	11	29.2
IONOSPHERE	351	34	1.8	YEAST6	1484	8	41.4
PIMA	768	8	1.9	ABALONE17	4177	10	71.0
WINE	178	13	2	ABALONE20	4177	10	159.7
GLASS	214	9	2.1				
GERMAN	1000	23	2.3	DGFIP 9 2	440	173	3
VEHICLE	846	18	3.3	DGFIP 4 2	255	82	3.8
HAYES	132	4	3.4	DGFIP 8 1	1028	255	4.6
SEGMENTATION	2310	19	6	DGFIP 8 2	1031	254	4.6
ABALONE8	4177	10	6.4	DGFIP 9 1	409	171	5.1
YEAST3	1484	8	8.1	DGFIP 4 1	240	76	5.2
PAGEBLOCKS	5473	10	8.8	DGFIP 16 1	789	162	8.7
SATIMAGE	6435	36	9.3	DGFIP 16 2	786	164	9.1

For our **MLFP** method, the hyper-parameters μ for the regularization and c for the margin are both tuned in the range $[0, 1]$, using a Bayesian optimization with 400 calls. The Bayesian optimization is done with the Scikit-Optimize library [3]. As the matrix $\mathbf{M}$ can be expressed as $\mathbf{L}^T\mathbf{L}$ (Cholesky decomposition), we directly learn a diagonal matrix $\mathbf{L}$. Since we are not particularly interested, in this paper, in low rank matrices, we do not impose any constraint on the dimension of $\mathbf{L}$. At each iteration of the optimization process, the spectral radius of the matrix $\mathbf{L}$ is constrained to be less than one so that $\mathbf{M} = \mathbf{L}^T\mathbf{L}$ has its largest value less than one.

We compare **MLFP** with several methods : The 3-Nearest Neighbor algorithm **(3NN)**, as a baseline. **LMNN**, where the hyper-parameter μ, which controls the trade-off between the two parts of the loss (see [WS09] for more details), is tuned in $[0, 1]$ using a Bayesian optimization with 20 calls. **ITML** [DKJ+07]. **GMML** [ZHS16], where the parameter t is tuned in $[0, 1]$ also with 20 calls of a Bayesian optimization. **IML** [WZJ+18] where we select $5k$ points for the sampling selection and we also tune the μ parameter of the **LMNN** algorithm in $[0, 1]$. We used 0.8 for the ratio of matching as suggested in the paper. **ImbML** [GMHS19] where the parameter m is tuned in $\{1, 10, 100, 1000, 10000\}$, the parameter λ in $\{0, 0.01, 0.1, 1, 10\}$ and the parameter a in $[0, 1]$. We also use a Bayesian optimization with 400 calls.

1. https://archive.ics.uci.edu/ml/datasets.html
2. https://sci2s.ugr.es/keel/datasets.php
3. https://scikit-optimize.github.io/

5.2 Results

The main results are reported in Table 2. Unsurprisingly, all metric learning methods perform better than a **3NN**. Furthermore, in terms of F-Measure, those which were designed to deal with imbalanced scenarios perform better than **LMNN**, **ITML** or **GMML**. However, the most competitive method is **MLFP** : the F-Measure is increased on average by 1.6 points compared to the second best method (**ImbML**). More precisely, our **MLFP** outperforms all the other methods on 8 (over 19) datasets. The fact that **MLFP** works better than **ImbML** shows the advantage of learning a specific metric when computing distances to positive examples. So, it is not strictly necessary to take all terms of the F-measure into account in the loss function. Focusing on False Negative and False Positive Rate, as we propose here, is enough. Furthermore, as it can be seen on Figure 3, both **ImbML** and **MLFP** focuses on the minority class, but they perform this task in a different way. Our method tries to reduce the number of FN by increasing the decision boundaries around each of positive example and then reduces the impact of each negatives by surrounding them. In **ImbML**, the possibility of having very large margins in the learned space has the disadvantage of creating larger areas of negative classification and this potentially increases the risk of FN.

For the DGFiP datasets, the results are available in the Table 3. We can see that our method outperforms all other metric learning methods for these datasets. As our algorithm tends to focus on the positives, and therefore the fraudsters, it is obviously more suitable for them. This also comes from the fact that we know that, in this kind of context, fraudsters often try to mimic the behaviour of non-frauders in order to stay undetected. By artificially increasing the decision boundaries around known positives, it is easier to capture fraudsters in hiding.

In the theoretical part of this paper, we have proved that learning a matrix $\mathbf{M}$ under the constraint $\lambda_{\max}(\mathbf{M}) \leq 1$ allows our algorithm to focus first on reducing FN. An illustration of the impact of this contraint in terms of false negatives is shown in Figure 4 on the 19 datasets. This figure reports the percentage of false negatives **at test time** generated by the **3NN** algorithm and **MLFP** with or without the constraint. The results show that, compared to a **3NN** algorithm, **MLFP** systematically reduces the number of false negatives and thus has the desired effect. When comparing **MLFP** with and without the constraint, we can note that on 14 datasets out of 19, the use of the constraint $\lambda_{\max}(\mathbf{M}) \leq 1$ leads at test time to a smaller number of false negatives.

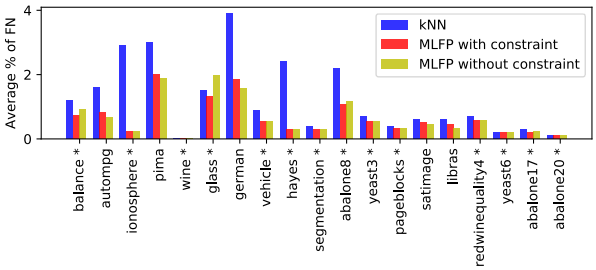

FIGURE 4 – Average percentage of false negatives for each dataset at test time (see Section 5 for more details), for k**NN** and **MLFP** with or without the constraint on λ_{max}. On 14 datasets (with $*$) over 19, the number of FN is lower for the version with the constraint. Note that the number of FN is always lower with **MLFP** compared to k**NN**.

6 Conclusion

In this paper, we have proposed a new metric learning algorithm to deal with imbalanced datasets. In this setting, finding the good compromise between the false negative and false positive rates is still an open problem. The original contribution of this paper comes from the optimization in our algorithm **MLFP** of a Mahalanobis distance which is *only* used to compare a new query to positive examples, while the Euclidean distance is still used when for comparing that query to negative samples. A constraint on the maximum eigenvalue of the learned matrix is introduced and has been shown to be provably efficient to reduce the false negative rate. Our paper is supported by a theoretical study and an extensive experimental evaluation showing that **MLFP** outperforms state-of-the-art metric-learning methods.

This work opens the door to two promising lines of research. First, in **MLFP** we learn a linear projection of the data. One interesting perspective would consist in kernelizing our metric learning algorithm or designing a deep learning version allowing us to capture non linearity. A simpler solution might also consist in learning different local metrics for different regions of the input space as done in [ZES16]. Second, as initiated in [SBK$^+$18], combining a Mahalanobis distance with a sampling strategy might lead to a new family of imbalanced learning methods.

TABLE 2 – Mean results in terms of F-Measure over 5 experiments for the different Metric Learning methods, with **3NN** as final classifier, on datasets *sorted by imbalance ratio*. The *mean* over all datasets among ML methods is given and the best results are in *bold*.

DATASETS	**3NN**	**LMNN**	**ITML**	**GMML**	**IML**	**ImbML**	**MLFP** (OURS)
BALANCE	0.880(0.018)	0.874(0.019)	0.931(0.032)	0.888(0.025)	0.886(0.029)	**0.960(0.019)**	0.874(0.003)
AUTOMPG	0.780(0.054)	0.792(0.031)	0.801(0.018)	**0.823(0.034)**	0.785(0.021)	0.790(0.044)	0.805(0.021)
IONOSPHERE	0.745(0.015)	0.803(0.049)	0.831(0.054)	0.764(0.056)	0.823(0.044)	0.786(0.053)	**0.923(0.026)**
PIMA	0.601(0.042)	0.591(0.037)	0.583(0.022)	0.579(0.035)	0.591(0.037)	0.575(0.026)	**0.635(0.032)**
WINE	0.968(0.016)	0.992(0.016)	0.992(0.016)	0.992(0.016)	0.992(0.016)	**0.992(0.016)**	0.961(0.041)
GLASS	0.735(0.049)	0.710(0.064)	**0.759(0.051)**	0.750(0.032)	0.710(0.064)	0.716(0.043)	0.747(0.034)
GERMAN	0.407(0.049)	0.358(0.029)	0.430(0.073)	0.407(0.030)	0.352(0.029)	0.388(0.043)	**0.511(0.006)**
VEHICLE	0.850(0.045)	0.928(0.024)	0.931(0.019)	0.936(0.013)	0.933(0.026)	**0.937(0.014)**	0.859(0.037)
HAYES	0.581(0.210)	0.824(0.089)	0.829(0.071)	0.876(0.091)	0.824(0.089)	0.908(0.083)	**0.930(0.109)**
SEGMENTATION	0.882(0.031)	0.888(0.011)	0.866(0.029)	0.870(0.029)	0.895(0.020)	**0.909(0.028)**	0.882(0.024)
ABALONE8	0.223(0.025)	0.220(0.040)	0.213(0.025)	0.210(0.038)	0.228(0.021)	0.200(0.023)	**0.336(0.018)**
YEAST3	0.719(0.028)	0.734(0.020)	0.742(0.034)	**0.747(0.031)**	0.717(0.032)	0.723(0.023)	0.725(0.022)
PAGEBLOCKS	0.855(0.027)	0.844(0.027)	0.850(0.023)	0.864(0.022)	0.842(0.027)	**0.865(0.021)**	0.860(0.022)
SATIMAGE	0.688(0.034)	0.707(0.038)	0.710(0.024)	0.682(0.028)	0.710(0.039)	**0.731(0.030)**	0.697(0.030)
LIBRAS	0.694(0.188)	0.725(0.105)	0.722(0.204)	0.667(0.272)	0.690(0.120)	**0.729(0.157)**	0.694(0.066)
REDWINEQUALITY4	0.062(0.075)	0.057(0.114)	0.027(0.053)	0.055(0.068)	0.000(0.000)	0.031(0.062)	**0.083(0.039)**
YEAST6	0.560(0.205)	0.578(0.246)	0.523(0.205)	0.458(0.307)	**0.629(0.244)**	0.606(0.148)	0.527(0.152)
ABALONE17	0.000(0.000)	0.000(0.000)	0.029(0.057)	0.000(0.000)	0.000(0.000)	**0.073(0.000)**	0.053(0.033)
ABALONE20	0.000(0.000)	0.000(0.000)	0.000(0.000)	0.000(0.000)	0.044(0.089)	0.000(0.093)	**0.078(0.029)**
MEAN	0.591	0.612	0.619	0.609	0.613	0.627	**0.643**
AVERAGE RANK	4.7	3.9	3.5	3.9	4.1	3.1	**3**

TABLE 3 – Mean results in terms of F-Measure over 5 experiments for the different Metric Learning methods, with **3NN** as final classifier, on private DGFiP datasets *sorted by imbalance ratio*. The *mean* over all datasets among ML methods is given and the best results are in *bold*.

DATASETS	**3NN**	**LMNN**	**ITML**	**GMML**	**IML**	**ImbML**	**MLFP** (OURS)
DGFIP 9 2	0.231(0.047)	0.266(0.063)	0.232(0.102)	0.186(0.070)	0.306(0.053)	0.187(0.057)	**0.391(0.026)**
DGFIP 4 2	0.049(0.059)	0.076(0.097)	0.114(0.097)	0.136(0.025)	0.099(0.089)	0.071(0.094)	**0.307(0.067)**
DGFIP 8 1	0.167(0.047)	0.164(0.041)	0.102(0.048)	0.131(0.070)	0.203(0.047)	0.137(0.058)	**0.308(0.004)**
DGFIP 8 2	0.175(0.044)	0.161(0.064)	0.132(0.067)	0.168(0.021)	0.129(0.031)	0.161(0.047)	**0.304(0.006)**
DGFIP 9 1	0.067(0.087)	0.064(0.053)	0.140(0.059)	0.103(0.092)	0.150(0.096)	0.184(0.109)	**0.290(0.027)**
DGFIP 4 1	0.094(0.078)	0.029(0.057)	0.000(0.000)	0.055(0.068)	0.000(0.000)	0.033(0.067)	**0.262(0.025)**
DGFIP 16 1	0.058(0.048)	0.139(0.123)	0.065(0.080)	0.159(0.105)	**0.282(0.076)**	0.119(0.102)	0.207(0.016)
DGFIP 16 2	0.076(0.039)	0.035(0.044)	0.078(0.040)	0.164(0.086)	0.107(0.070)	0.164(0.164)	**0.206(0.015)**
MEAN	0.115(0.067)	0.117(0.081)	0.108(0.067)	0.138(0.042)	0.160(0.101)	0.132(0.055)	**0.284(0.060)**
AVERAGE RANK	4.8	4.9	5.1	3.9	3.6	4.3	**1.1**

Références

[Agg17] Charu C. Aggarwal. *Outlier Analysis*. Springer International Publishing, 2017.

[AMZ16] Aisha Abdallah, Mohd Aizaini Maarof, and Anazida Zainal. Fraud detection system : A survey. *Journal of Network and Computer Applications*, 68 :90–113, 2016.

[BE02] Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of machine learning research*, 2(Mar) :499–526, 2002.

[BHS13] Aurélien Bellet, Amaury Habrard, and Marc Sebban. A survey on metric learning for feature vectors and structured data. *CoRR*, abs/1306.6709, 2013.

[BHS15] Aurélien Bellet, Amaury Habrard, and Marc Sebban. *Metric Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2015.

[DCK18] Sri Harsha Dumpala, Rupayan Chakraborty, and Sunil Kumar Kopparapu. A novel data representation for effective learning in class imbalanced scenarios. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 2100–2106, 7 2018.

[DKJ+07] 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.

[DML+18] Shuya Ding, Bilal Mirza, Zhiping Lin, Jiuwen Cao, Xiaoping Lai, Tam V. Nguyen, and Jose Sepulveda. Kernel based online learning for imbalance multiclass classification. *Neurocomputing*, 277 :139–148, 2018.

[FHSHG18] Jordan Frery, Amaury Habrard, Marc Sebban, and Liyun He-Guelton. Non-linear gradient boosting for class-imbalance learning. In *2nd International Workshop on Learning with Imbalanced Domains : Theory and Applications*, pages 38–51, 2018.

[FWJ+18] 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*, PP :1–12, 01 2018.

[GMHS19] Léo Gautheron, Emilie Morvant, Amaury Habrard, and Marc Sebban. Metric learning from imbalanced data. In *arXiv*. 1909.01651, 2019.

[HLCLT16] Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. Learning deep representation for imbalanced classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5375–5384, 2016.

[K+13] Brian Kulis et al. Metric learning : A survey. *Foundations and Trends® in Machine Learning*, 5(4) :287–364, 2013.

[KHB+17] Salman H. Khan, Munawar Hayat, Mohammed Bennamoun, Ferdous A. Sohel, and Roberto Togneri. Cost-sensitive learning of deep feature representations from imbalanced data. *IEEE transactions on neural networks and learning systems*, 29(8) :3573–3587, 2017.

[MLPC15] Josey Mathew, Ming Luo, Chee Khiang Pang, and Hian Leng Chan. Kernel-based smote for svm classification of imbalanced datasets. In *IECON 2015-41st Annual Conference of the IEEE Industrial Electronics Society*, pages 001127–001132. IEEE, 2015.

[PTMH19] María Pérez-Ortiz, Peter Tiňo, Rafal Mantiuk, and César Hervás-Martínez. Exploiting synthetically generated data with semi-supervised learning for small and imbalanced datasets. *CoRR*, abs/1903.10022, 2019.

[Rij79] Cornelis J. Van Rijsbergen. *Information Retrieval*. 1979.

[SBK+18] Shiven Sharma, Colin Bellinger, Bartosz Krawczyk, Osmar R. Zaïane, and Nathalie Japkowicz. Synthetic oversampling with the majority class : A new perspective on handling extreme imbalance. *IEEE International Conference on Data Mining*, pages 447–456, 2018.

[Sch06] Jörg Schiller. The impact of insurance fraud detection systems. *Journal of Risk and Insurance*, 73(3) :421–438, 2006.

[WJS+17] Fei Wu, Xiao-Yuan Jing, Shiguang Shan, Wangmeng Zuo, and Jing-Yu Yang. Multiset feature learning for highly imbalanced data classification. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 1583–1589. AAAI Press, 2017.

[WS09] Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb) :207–244, 2009.

[WZJ+18] Nan Wang, Xibin Zhao, Yu Jiang, Yue Gao, and KLISS BNRist. Iterative metric learning for imbalance data classification. In *27th International Joint Conference on Artificial Intelligence*, pages 2805–2811, 2018.

[ZES16] Valentina Zantedeschi, Rémi Emonet, and Marc Sebban. Metric learning as convex combinations of local models with generalization guarantees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1478–1486, 2016.

[ZHS16] Pourya Zadeh, Reshad Hosseini, and Suvrit Sra. Geometric mean metric learning. 07 2016.

[ZWZ+19] Xiaogang Zhang, Dingxiang Wang, Yicong Zhou, Hua Chen, Fanyong Cheng, and Min Liu. Kernel modified optimal margin distribution machine for imbalanced data classification. *Pattern Recognition Letters*, 125 :325–332, 2019.