University Grenoble Alpes
Master of Science in Industrial and Applied Mathematics
Specialization Statistics

MULTI-CLASS SEMI-SUPERVISED LEARNING THROUGH PSEUDO-LABELLING

Vasilii Feofanov

Research project performed at

Laboratoire d'Informatique de Grenoble

Under the supervision of:

Émilie Devijver, CNRS Researcher
Massih-Reza Amini, Professor

Reviewed by:

Sana Louhichi, Professor

Defended before a jury composed of:

Jean-Baptiste Durand, Assistant Professor
Anatoli Iouditski, Professor

June, 2018

Contents

Introduction

In many real-life applications, the labelling of training observations for learning is costly and sometimes even not realistic. For example, in medical diagnosis or biological data analysis, labelling data may require very expensive tests so that only small labelled data sets may be available. In many other cases, like web oriented applications, huge amount of observations arrive sequentially and there is not enough time to label data for different information needs; while unlabelled data are abundant.

Learning with labelled and unlabelled data, or semi-supervised learning (SSL), has been an active area of research in the machine learning community since the end of nineties [5]. In this case, labelled examples are generally assumed to be very few, leading to an inefficient supervised model, while unlabelled training examples contain valuable information about the prediction problem whose exploitation can lead to an increase of performance.

In SSL we consider a set of labelled training examples identically and independently distributed (i.i.d.) with respect to a fixed yet unknown probability distribution, and a set of unlabelled training examples supposed to be drawn i.i.d. from the marginal distribution. If the unlabelled set is empty, then the problem reduces to supervised learning. The other extreme case is the situation where the set of labelled observations is empty and which corresponds to unsupervised learning. One of the hypotheses under which the learning with labelled and unlabelled data has been studied is the low-density separation. For this, many advances have been made on the algorithmic and theoretical levels. Although real-life applications, for which the design of SSL techniques is attractive, are multi-class by nature; the majority of the theoretical results for semi-supervised learning consider the binary case.

In this study we propose to tackle the problem in a transductive setting [26, pp. 339-371] with the aim of bounding the error of the majority vote classifier over the unlabelled set based on the distribution of its predictions over different classes, and by extending the study of [1] to the multi-class case. In the latter, the authors have considered the distribution of unsigned margins to bound the transductive risk of the voted classifier in the binary case. This bound then leaded to choose automatically a threshold for which the conditional Bayes risk is minimal. In [10], the value of this threshold was empirically tested in a self-training algorithm which iteratively pseudo-labels unlabelled examples based on the prediction scores and the founded threshold.

In our multi-class case, we follow [20] who considered the confusion matrix as an error measure instead of the error rate. The main reason of this choice is that in some applications, the confusion matrix was found to be more informative and more useful than other error measures. Based on this, [20] proposed a PAC-Bayesian generalisation bound on the Gibbs confusion matrix for multi-class classification. The goal of this paper is to continue the study of the aforementioned PAC-Bayesian theory [17, 13] with the objective to the semi-supervised transductive case. As an application, we propose a new semi-supervised approach: the margin-based multi-class self-learning algorithm. The empirical results we obtained demonstrate the efficiency of our algorithm in comparison with the supervised approach.

The paper is organised as follows. In Chapter 1 we introduce the context of our research and recall related works. In Chapter 2 we propose several transductive bounds and discuss the quality. In Chapter 3 we present the application, namely, an extension of the self-learning algorithm as well as perform numerical experiments to validate the approach we propose. The contribution of this work is summarised in Conclusion.

Chapter 1

Introduction to the PAC-Bayesian Theory and Its Application to SSL

1.1 Transductive vs. Inductive Inference

Consider the mono-label classification problem, either binary or multi-class. Let $\mathcal{X} \subset \mathbb{R}^d$ be the input space and $\mathcal{Y}$ be the output space. Designate $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, where $\times$ is the Cartesian product. Consider a set of labelled examples $\mathrm{Z}_\mathcal{L} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$, $\mathrm{Z}_\mathcal{L} \in \mathcal{Z}^l$ as well as a set of unlabelled observations $\mathrm{X}_\mathcal{U} = \{\mathbf{x}'_i\}_{i=l+1}^{l+u}$, $\mathrm{X}_\mathcal{U} \in \mathcal{X}^u$. When one considers the semi-supervised setting, it is assumed that $l \ll u$. We suppose that all pairs $(\mathbf{x}, y)$ from $\mathrm{Z}_\mathcal{L}$ are i.i.d. with a joint probability $\mathcal{D}$, and $\mathbf{x}$ are distributed according to $P_X(\mathbf{x})$ over $\mathcal{X}$. The goal of learning is to predict labels for observations from the set $\mathrm{X}_\mathcal{U}$. For this, we draw a function h from $\mathcal{X}$ to $\mathcal{Y}$ based on the training set $\mathrm{Z}_\mathcal{L}$ and, in the case of semi-supervised learning, the unlabelled set $\mathrm{X}_\mathcal{U}$.

Definition 1.1.1. *We define a* loss function ℓ *as:*

$$\begin{aligned} \ell : \mathcal{Y} \times \mathcal{Y} &\to \mathbb{R} \\ (\hat{y}, y) &\mapsto \ell(\hat{y}, y) \end{aligned}$$

It is used through $(h(\mathbf{x}), y) \mapsto \ell(h(\mathbf{x}), y)$. The loss function characterises the cost to make a mistake, i.e. when h classifies $\mathbf{x}$ to a wrong class.

Example 1.1.2. *The 0/1 loss function $\ell_{0/1}$ is defined as follows:*

$$\ell_{0/1}(h(\mathbf{x}), y) := \mathbb{1}_{h(\mathbf{x}) \neq y},$$

where $\mathbb{1}_{\{\cdot\}}$ is the indicator function.

Definition 1.1.3. *The* risk functional*, or the* risk*, is defined as the expectation of the loss function:*

$$R(h) := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\, \ell(h(\mathbf{x}), y) = \int \ell(h(\mathbf{x}), y) d\mathcal{D}(\mathbf{x}, y)$$

Further, we consider a fixed class $\mathcal{H}$ of functions from $\mathcal{X}$ to $\mathcal{Y}$, which is called the hypothesis space. In turn, a function h from the class $\mathcal{H}$ is called a hypothesis.

The classical approach in (semi-) supervised learning is to formulate the goal of learning as the task of finding a hypothesis h^* that minimises the risk functional:

$$h^* = \operatorname*{argmin}_{h \in \mathcal{H}} R(h).$$

In practice, the risk can not be computed explicitly as the joint probability $P(\mathbf{x}, y)$ is unknown. Instead, estimation of $R(h)$ (called the *empirical risk*) is done based on the observations from the training set $\mathrm{Z}_{\mathcal{L}}$:

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

It gives an approximation of h^*, $\bar{h}^* = \operatorname{argmin}_{h \in \mathcal{H}} \bar{R}(h)$ that can be used to make a prediction not only for unlabelled observations $\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}$, but for any $\mathbf{x} \in \mathcal{X}$.

This approach of learning is called **inductive** as it aims to induce a general model from the training data, and then predicts labels for the unlabelled set using this model. However, this type of inference is not always the most optimal as it considers a more complex problem of function estimation, instead of estimating this function in u points. There are situations where correct approximation of the function is difficult. For example, in semi-supervised framework the training size is greatly less than the size of the unlabelled set. As a rule, if the training size is small enough, reasonable estimation of the function might not be provided. Besides this, one can notice that in many applications it is required to estimate function values at given points rather than to find the functional dependency.

Thereby, Vapnik suggested in [25, 26] to consider a learning task in a **transductive** setting which implies reasoning from the given training set directly to the unlabelled observations. Instead of inferring a general rule, transductive learning more concentrate on the unlabelled examples to predict them accurately. In this case, the methodology includes the extraction of additional information from the unlabelled set $\mathrm{X}_{\mathcal{U}}$ to perform their predictions.

There is another advantage of the transductive approach. When it is required to provide theoretical guarantees for the error value on the unlabelled set, one way to do it is to find an upper bound of the error that holds with high probability (more information about that is given in Section 1.3). It is reasonable to aim for the tightest bound with respect to the true error. The transductive and inductive settings target different objectives: the transductive approach estimates an error only on the working set, whereas the inductive one estimates an error on the whole space. From this, one can infer that, generally speaking, error bounds in transductive setting would be tighter than error bounds for inductive one [5].

Thus, in transductive learning the following framework is considered (Setting 1 in [26]):

1. A set of *i.i.d.* observations $\mathrm{X}_{\mathcal{N}} = \{\mathbf{x}_i\}_{i=1}^{n} \subset \mathcal{X}$ $(n = l + u)$ is given, and observations are distributed with respect to $P_X(\mathbf{x})$ over $\mathcal{X}$.

2. It is assumed that observations are classified according to the distribution $P_Y(y|\mathbf{x})$.

3. l training examples are chosen uniformly from $\mathrm{X}_{\mathcal{N}}$ *without replacement*. According to the distribution $P_Y(y|\mathbf{x})$, they are labelled as $y_1, \ldots, y_l$. Thus, one forms the training (labelled) set $\mathrm{Z}_{\mathcal{L}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$. The observations not chosen from $\mathrm{X}_{\mathcal{N}}$ form the unlabelled set $\mathrm{X}_{\mathcal{U}} = \{\mathbf{x}'_i\}_{i=l+1}^{l+u}$.

Then, the goal of learning in the transductive setting is formulated as the task of finding a hypothesis h^* based on $\mathrm{Z}_{\mathcal{L}}$ and $\mathrm{X}_{\mathcal{U}}$ that minimises the transductive risk $R_{\mathcal{U}}$. Before we define the notion of the transductive risk, let us give another representation for the inductive risk to show clearer the difference between the inductive and transductive objective.

Proposition 1.1.4. *The inductive risk can be re-written in the following way:*

$$R(h) = \mathbb{E}_{P_X(\mathbf{x})} \mathbb{E}_{P_Y(y|\mathbf{x})} \ell(h(\mathbf{x}), y) = \int \int \ell(h(\mathbf{x}), y)\, dP_x(\mathbf{x})\, dP_Y(y|\mathbf{x}).$$

Finally, we give a definition of the transductive risk.

Definition 1.1.5. *The* transductive risk *is defined as:*

$$R_{\mathcal{U}}(h) := \frac{1}{u} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{E}_{P_Y(y|\mathbf{x}')} \ell(h(\mathbf{x}'), y) = \frac{1}{u} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \int \ell(h(\mathbf{x}'), y)\, dP_Y(y|\mathbf{x}').$$

where $P_Y(y|\mathbf{x}')$ is a posterior distribution of label y.

Assumption 1.1.6. *Further, we assume that there exists a deterministic function $\phi : \mathcal{X} \to \mathcal{Y}$ that labels examples by $y = \phi(x)$. It means that for any $\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}$ there is one and only one possible label y'. It can be noted that ϕ doesn't necessarily belong to the hypothesis space $\mathcal{H}$. Thus, given the function ϕ the transductive risk has the following form:*

$$R_{\mathcal{U}}(h) = \frac{1}{u} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \ell(h(\mathbf{x}'), \phi(\mathbf{x}')) = \frac{1}{u} \sum_{\substack{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}} \\ y' = \phi(\mathbf{x}')}} \ell(h(\mathbf{x}'), y').$$

Further, we denote by y' the label associated to $\mathbf{x}'$.

1.2 Bayesian Learning

In machine learning, one of the classical approach to inference a model is to follow Bayesian reasoning. The idea is to consider a probability distribution over the hypothesis space $\mathcal{H}$. We suppose that there is a prior belief on what is the probability to choose a hypothesis $h \in \mathcal{H}$. Then, after observing the data we can update the probability according to the Bayes rule.

More specifically, let h be distributed *a priori* with a probability $P_H(h)$. This prior distribution might be based on some experience or domain knowledge. If it is not available or hard to interpret, the prior distribution is set as uniform. Next, let $\mathrm{Z}_{\mathcal{L}}$ be distributed according to $P_{\mathrm{Z}}(\mathrm{Z}_{\mathcal{L}})$ that is defined over $\mathcal{Z}^l$. In other words, the probability to observe (unconditionally) the training set $\mathrm{Z}_{\mathcal{L}} \in \mathcal{Z}^l$ is defined as $P_{\mathrm{Z}}(\mathrm{Z}_{\mathcal{L}})$. This probability is often called the *data likelihood* [19]. Since we assume that training examples are i.i.d., $P_{\mathrm{Z}} \equiv \mathcal{D}^l$. Further, $P_{\mathrm{Z}}(\mathrm{Z}_{\mathcal{L}}|h)$ denotes the data likelihood given information that h holds for this data. Finally, we define $P_H(h|\mathrm{Z}_{\mathcal{L}})$, the *posterior* probability of a hypothesis h. It corresponds to the update of the probability after observing $\mathrm{Z}_{\mathcal{L}}$. This can be computed by applying the Bayes rule:

$$P_H(h|\mathrm{Z}_{\mathcal{L}}) = \frac{P_H(h) P_{\mathrm{Z}}(\mathrm{Z}_{\mathcal{L}}|h)}{\sum_{h' \in \mathcal{H}} P_H(h') P_{\mathrm{Z}}(\mathrm{Z}_{\mathcal{L}}|h')}, \tag{1.1}$$

Based on the reasoning described above, different classification procedures are designed. For instance, an apparent technique is to choose the hypothesis that is *maximum a posteriori* (MAP), i.e. the hypothesis with the largest posterior probability:

$$h_{\mathrm{MAP}} := \operatorname*{argmax}_{h \in \mathcal{H}} P_H(h|\mathrm{Z}_{\mathcal{L}}) = \operatorname*{argmax}_{h \in \mathcal{H}} P_H(h) P_{\mathrm{Z}}(\mathrm{Z}_{\mathcal{L}}|h).$$

This approach works well when the algorithm yields a significantly large posterior probability to only one hypothesis. But this is not always the case. Firstly, there would not be given enough data to determine a significant difference between the two or more maximal a posteriori hypotheses. Secondly, it would exist a situation when the true function ϕ does not belong to the hypothesis space $\mathcal{H}$. Thirdly, from the practical point of view, there might be required to consider a large number of hypotheses, so finding the most appropriate one would be computationally expensive. All of these problems could be prevent with the ensemble learning approach [9]. Indeed, we can combine classifiers from the hypothesis space to get better performance. For each new instance, we compute the most expected label by performing a vote among the hypotheses weighted by the posterior probabilities. This approach is called the *Bayes optimal classifier*.

Definition 1.2.1. *The* Bayes optimal classifier *is defined in the following way:*

$$O(\mathbf{x}) := \underset{c\in\mathcal{Y}}{\operatorname{argmax}}\, \mathbb{E}_{P_H(h|\mathrm{Z}_\mathcal{L})} \mathbb{1}_{h(\mathbf{x})=c} = \underset{c\in\mathcal{Y}}{\operatorname{argmax}} \sum_{h\in\mathcal{H}} \mathbb{1}_{h(\mathbf{x})=c}\, P_H(h|\mathrm{Z}_\mathcal{L}), \qquad \forall \mathbf{x} \in \mathcal{X}. \tag{1.2}$$

Given a prior distribution and a fixed hypothesis space this classifier is optimal and other classification algorithms would be worse on average. However, one can be noticed that the estimation of the posterior distribution could be practically infeasible. That is why for application it would be better to consider the *Q-weighted majority vote classifier* that, in contrast, regards the posterior distribution Q over $\mathcal{H}$ that is not necessarily a posterior distribution in the Bayesian sense[1]. Algorithms as the *AdaBoost* [11], the *Random Forest* [2] and the *Support Vector Machine* [25] can be considered as the Q-weighted majority vote. In [11] it was shown that under certain conditions the AdaBoost is considered as approximation of the Bayes optimal classifier. Thus, the study of the majority vote classifier gives a tool for analysing of widely used in practice classification methods. Further, we will take into consideration the binary classification framework with $\mathcal{Y} = \{-1, +1\}$ until the opposite is said.

Definition 1.2.2. *We define the* Q-weighted majority vote classifier *(also called the Bayes classifier) as follows:*

$$B_Q(\mathbf{x}) := \operatorname{sign}[\mathbb{E}_{h\sim Q} h(\mathbf{x})], \qquad \forall \mathbf{x} \in \mathcal{X}. \tag{1.3}$$

Remark 1.2.3. *We could also define the Q-weighted majority vote classifier in the same way as the optimal classifier in* (1.2)*. In fact, these two definitions are equivalent. Indeed, the classifier chooses, for an observation, the label that got more votes with respect to the distribution Q. The sign function gives +1 (resp. -1) when the argument is positive (resp. negative), which implies the win of the class with the label +1 (resp. -1). However, Definition 1.2.2 is applicable only for the case when $\mathcal{Y} = \{-1, +1\}$, whereas the* $\operatorname{argmax}_{y\in\mathcal{Y}}$ *view works for the multi-class framework too.*

Definition 1.2.4. *The* transductive risk of the Q-weighted majority vote classifier *is defined by:*

$$R_\mathcal{U}(B_Q) := \frac{1}{u} \sum_{\mathbf{x}'\in \mathrm{X}_\mathcal{U}} \ell(B_Q(\mathbf{x}'), y'),$$

where y denotes the unique true label of an observation $\mathbf{x}$. For the 0/1 loss function, the risk is represented as:

$$R_\mathcal{U}(B_Q) = \frac{1}{u} \sum_{\mathbf{x}\in \mathrm{X}_\mathcal{U}} \mathbb{1}_{B_Q(\mathbf{x})\neq y}.$$

In the study of the majority vote classifier another algorithm is also considered, the *Gibbs classifier.*

Definition 1.2.5. *The* Gibbs classifier *is a stochastic learning algorithm that chooses randomly a hypothesis $h \in \mathcal{H}$ according to the distribution Q and then predicts for $\mathbf{x} \in \mathcal{X}$ a label as $h(\mathbf{x})$.*

While the Gibbs classifier is stochastic, we determine its risk as the expected risk over all hypotheses from $\mathcal{H}$.

Definition 1.2.6. *The* transductive risk of the Gibbs classifier *is defined by:*

$$R_\mathcal{U}(G_Q) := \frac{1}{u} \sum_{x'\in \mathrm{X}_\mathcal{U}} \mathbb{E}_{h\sim Q}[\ell(h(\mathbf{x}'), y')], \tag{1.4}$$

where y' represents the true label of the observation $\mathbf{x}'$.

[1] In this case, by posterior distribution we call any distribution over $\mathcal{H}$ that has been obtained after observing data.

Particularly, for the 0/1 loss function, (1.4) is rewritten as:

$$R_{\mathcal{U}}(G_Q) = \frac{1}{u} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{E}_{h \sim Q}(\mathbb{1}_{[h(\mathbf{x}') \neq y']}) = \frac{1}{u} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{P}_{h \sim Q}(h(\mathbf{x}') \neq y).$$

In the remainder of this paper, we focus only on the 0/1 loss function.

The following relation between the transductive risk of the Gibbs and the Q-weighted majority vote classifiers holds.

Claim 1.2.7 (Lemma 4.1 in [14] for the inductive risk)**.** *It is true that* $R_{\mathcal{U}}(B_Q) \leq 2R_{\mathcal{U}}(G_Q)$.

Proof. Without loss of generality, consider a situation when the transductive set consists of only one example: $\mathrm{X}_{\mathcal{U}} = \{\mathbf{x}'\}$. Denote the true label for this example as y'.

- $B_Q(\mathbf{x}') \neq y'$. Then, $R_{\mathcal{U}}(B_Q) = 1$. Let's designate the posterior probabilities $\{P(h|\mathrm{Z}_{\mathcal{L}})\}_{h \in \mathcal{H}}$ as $\{\frac{m_h}{n}\}_{h \in \mathcal{H}}$, $\sum_{h \in \mathcal{H}} m_h = n$. From $B_Q(\mathbf{x}') \neq y'$ we derive that with a probability at least $\frac{n/2}{n} = \frac{1}{2}$ the Gibbs classifier will mistake and the loss will be equal to 1. Then, we deduce that $2R_{\mathcal{U}}(G_Q) \geq 2 \cdot (1 \cdot \frac{1}{2} + 0 \cdot \frac{1}{2}) = 1 = R_{\mathcal{U}}(B_Q)$.
- $B_Q(\mathbf{x}') = y'$. From the definition of the risk we immediately infer: $R_{\mathcal{U}}(B_Q) = 0 \leq R_{\mathcal{U}}(G_Q) \leq 2R_{\mathcal{U}}(B_Q)$.

□

As it can be seen, this claim gives us an upper bound of the transductive risk for the Q-weighted majority vote when we are able to obtain an upper bound for the Gibbs risk. However, this upper bound mostly does not have any practical use, since it appears to be not tight. To deduce a more reasonable bound, one can use information extracted from the unlabelled set. For instance, we can compute unsigned margin values for unlabelled observations.

Definition 1.2.8. *Let Q be a posterior distribution over the hypothesis space $\mathcal{H}$. The* unsigned margin*, or the* margin*, is defined as follows:*

$$m_Q(\mathbf{x}) := \left| \mathbb{E}_{h \sim Q} h(\mathbf{x}) \right|.$$

For an observation, its margin can be treated as an indicator of confidence to be correctly classified. It lies in $[0, 1]$. The margin that is equal 1 states that all hypotheses vote for one label, either +1 or -1, which can be considered as the confident prediction. If the margin is 0, then the observation will be correctly classified no better than a chance. We assume that the majority vote classifier makes its errors in most cases on observations with low margins. Then, it would be also interesting to take into consideration a transductive risk function that computes error only on unlabelled observation with the margin more than some $\theta > 0$.

Definition 1.2.9. *Let $\theta > 0$. The* joint Bayes risk *is defined as follows:*

$$R_{\mathcal{U} \wedge \theta}(B_Q) := \frac{1}{u} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}') \neq y' \,\wedge\, m_Q(\mathbf{x}') > \theta}, \tag{1.5}$$

where y' denotes the unique true label of an observation $\mathbf{x}'$.

1.3 PAC-Bayesian Framework

One of the most important field in machine learning is the theory of prediction. Given a learning algorithm and a training set, what the prediction error on the target set we may expect? Typically, we can consider a confidence interval for the target that holds with probability at least $1 - \delta$, and states that the target error is bounded by some function of δ and error on the

training set [13]. This type of formulation is named as *Probably Approximately Correct (PAC)* learning [24]. It does not make any assumption on data distribution and hypotheses' priors. The PAC-Bayesian framework [18, 17] uses this idea for Bayesian learning. More specifically, the PAC-Bayesian theorem aims to provide this kind of guarantees for the Gibbs classifier.

Classical works study PAC-Bayesian framework for the inductive inference, where the goal is to bound the risk function (generalisation error). Suppose that P and Q are prior and posterior distributions over the hypothesis space $\mathcal{H}$ respectively. In the inductive case, the true and the empirical Gibbs risk are defined as follows:

$$R(G_Q) := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\mathbb{E}_{h\sim Q}\mathbb{1}_{h(\mathbf{x})\neq y},$$
$$R_{\mathcal{L}}(G_Q) := \frac{1}{l}\sum_{i=1}^{l}\mathbb{E}_{h\sim Q}\mathbb{1}_{h(\mathbf{x}_i)\neq y_i}.$$

Then, we recall the PAC-Bayesian theorem that derives a generalisation bound for the Gibbs classifier.

Theorem 1.3.1 (Theorem 5.1 in [13]). *For all $\mathcal{D}$, for any choice of P, for any $\delta \in (0,1]$ it is true that:*

$$\mathbb{P}_{\mathrm{Z}_{\mathcal{L}}\sim\mathcal{D}^l}\left(\forall Q \text{ on } \mathcal{H} : kl\left[R_{\mathcal{L}}(G_Q), R(G_Q)\right] \leq \frac{1}{l}\left[KL(Q \parallel P) + \ln\frac{l+1}{\delta}\right]\right) \geq 1-\delta,$$

where $KL(Q \parallel P)$ is the Kullback-Leibler divergence between Q and P:

$$KL(Q \parallel P) = \mathbb{E}_{h\sim Q}\left[\ln\frac{dQ(h)}{dP(h)}\right],$$

and $kl(q,p)$ is the the Kullback-Leibler divergence overload:

$$kl(q,p) = \begin{cases} q\log\frac{q}{p} + (1-q)\log\frac{1-q}{1-p} & \text{if } p > q, \\ 0 & \text{otherwise.} \end{cases}$$

An advantage of the PAC-Bayes theorem is that this bound holds for any choice of Q. Therefore, Q is not necessarily the classical Bayesian posterior. Thus, the result can be also connected with Q-weighted majority vote classifier through Lemma 4.1 from [14]. Despite the fact that the prior distribution P should be predefined, Theorem 1.3.1 holds for any choice of P.

A PAC-Bayes bound has also been got for the transductive inference. Based on Vapnik's work [25], a first explicit bound was proposed by [7]. After, this result was refined by [3]. The latter work proposes a bound for the Gibbs risk on the whole set:

$$R_{\mathcal{N}}(G_Q) = \frac{1}{n}\sum_{\mathbf{x}\in\mathrm{X}_{\mathcal{N}}}\mathbb{E}_{h\sim Q}\mathbb{1}_{h(\mathbf{x})\neq y}.$$

To get the bound for the transductive risk (which is the target functional we want to minimise) one can notice that:

$$R_{\mathcal{N}}(G_Q) = \frac{1}{n}(lR_{\mathcal{L}}(G_Q) + uR_{\mathcal{U}}(G_Q)).$$

Theorem 1.3.2 (Corollary 7 in [3]). *Following Vapnik's Setting 1, for any set $\mathrm{X}_{\mathcal{N}}$ of $n = l+u \geq 40$ examples, for any hypothesis space $\mathcal{H}$, for any $\delta \in (0,1]$, with probability at least $1-\delta$ over the choices $\mathrm{Z}_{\mathcal{L}}$ ($l \in [20, n-20]$), we have:*

$$R_{\mathcal{N}}(G_Q) \leq R_{\mathcal{L}}(G_Q) + \sqrt{\frac{n-l}{2ln}\left[KL(Q \parallel P) + \ln\frac{t(l,n)}{\delta}\right]},$$

where $t(l,n) := 3\ln(l)\sqrt{l(1-\frac{l}{n})}$.

As a disadvantage one can note that the Gibbs classifier is a stochastic method that is not used in practice. Although we are able to deduce a bound for the majority vote classifier using Claim 1.2.7, this bound would be not tight enough. In the next section we give a review of [1], where a transductive bound has been proposed for the majority vote classifier based on the margin distribution of the unlabelled set.

1.4 A Transductive Bound for the Majority Vote Classifier in the Binary Case

Further, π and μ denote respectively probability and expectation operators with respect to the uniform distribution over $\mathrm{X}_{\mathcal{U}}$. For instance, the mean margin value for the unlabelled set $\mathrm{X}_{\mathcal{U}}$ is $\frac{1}{u}\sum_{\mathbf{x}'\in\mathrm{X}_{\mathcal{U}}} m_Q(\mathbf{x}') = \mu\{m_Q(\mathbf{x}')\}$, and the proportion of observations in $\mathrm{X}_{\mathcal{U}}$ that have a margin higher than some θ is $\frac{1}{u}\sum_{\mathbf{x}'\in\mathrm{X}_{\mathcal{U}}} \mathbb{1}_{m_Q(\mathbf{x}')>\theta} = \pi(m_Q(\mathbf{x}') > \theta)$. Thus, the aforementioned notations are introduced just for sake of simplicity.

As it was mentioned in the previous section, the risk of the majority vote classifier is no greater than twice the Gibbs risk. To derive tighter guarantee, in [1] it was proposed to take into account the margin distribution of the unlabelled set, besides a given Gibbs risk bound.

In [1] a transductive bound for the joint Bayes risk is obtained first, then the corresponding bound for the Bayes risk is obtained as a corollary. Indeed, one can notice that those two risk functions are connected in the following way:

$$R_{\mathcal{U}}(B_Q) = R_{\mathcal{U}\wedge 0}(B_Q) + \pi(B_Q(\mathbf{x}') \neq y' \ \wedge \ m_Q(\mathbf{x}') = 0) \leq R_{\mathcal{U}\wedge 0}(B_Q) + \pi(m_Q(\mathbf{x}') = 0).$$

Lemma 1.4.1 (Lemma 3 in [1]). *Let* $\Gamma = \{\gamma |\ \exists\ \mathbf{x}' \in \mathrm{X}_{\mathcal{U}} : \gamma = m_Q(\mathbf{x}');\ \gamma > 0\}$. *Let enumerate its elements such that they form an ascending order:*

$$\gamma_1 < \gamma_2 < \cdots < \gamma_N,$$

where $N = |\Gamma|$. *Denote* $b_i = \pi(B_Q(\mathbf{x}') \neq y' \wedge m_Q(\mathbf{x}') = \gamma_i) = \frac{1}{u}\sum_{\mathbf{x}'\in\mathrm{X}_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}')\neq y' \wedge m_Q(\mathbf{x}')=\gamma_i}$. *Then,*

$$R_{\mathcal{U}}(G_Q) = \sum_{i=1}^{N} b_i\gamma_i + \frac{1}{2}(1 - \mu\{m_Q(\mathbf{x}')\}) \tag{1.6}$$

$$\forall\theta \in [0,1],\ R_{\mathcal{U}\wedge\theta}(B_Q) = \sum_{i=k+1}^{N} b_i \ \textit{with}\ k = \max\{i|\gamma_i \leq \theta\}. \tag{1.7}$$

This lemma provides a new way to connect the Bayes and Gibbs risks. It states that the Gibbs risk can be represented as the sum of two terms: 1) $\sum_{i=1}^{N} b_i\gamma_i$, the probability to make a mistake multiplied by the expected margin on misclassified examples, 2) $0.5(1 - \mu\{m_Q(\mathbf{x}')\})$, a constant term. In the following theorem, a transductive bound is obtained based on the solution of a linear program, in which we target to maximise $\sum_{i=1}^{N} b_i\gamma_i$.

Theorem 1.4.2 (Theorem 1 in [1]). *Consider* B_Q *as in Eq.* (1.3). *Suppose that an upper bound of the transductive Gibbs risk* $R_u^{\delta}(G_Q)$ *is given. Then for any* Q *and for all* $\delta \in (0,1]$, $\forall\theta \geq 0$, *with probability at least* $1-\delta$, *the following bounds hold:*

$$R_{\mathcal{U}}(B_Q) \leq \inf_{\gamma\in(0,1]} \left\{ \pi(m_Q(\mathbf{x}') < \gamma) + \frac{1}{\gamma}\left\lfloor K_u^{\delta}(Q) - M_Q^{<}(\gamma) \right\rfloor_+ \right\} \tag{1.8}$$

$$R_{\mathcal{U}\wedge\theta}(B_Q) \leq \inf_{\gamma\in(\theta,1]} \left\{ \pi(\theta < m_Q(\mathbf{x}') < \gamma) + \frac{1}{\gamma}\left\lfloor K_u^{\delta}(Q) + M_Q^{\leq}(\theta) - M_Q^{<}(\gamma) \right\rfloor_+ \right\}, \tag{1.9}$$

where

- $K_u^\delta(Q) = R_u^\delta(G_Q) + \frac{1}{2}(\mu\{m_Q(\mathbf{x}')\} - 1)$
- $M_Q^\triangleleft(t) = \mu\{m_Q(\mathbf{x}')\mathbb{1}_{m_Q(\mathbf{x}')\triangleleft t}\},\ (\triangleleft \in \{<, \leq\})$.
- $\lfloor x \rfloor_+ = x \cdot \mathbb{1}_{x>0}$.

Thus, Theorem 1.4.2 states that the bound of the Bayes risk is determined by $\pi(m_Q(\mathbf{x}') < \gamma) + \frac{1}{\gamma}\left\lfloor K_u^\delta(Q) - M_Q^<(\gamma)\right\rfloor_+$, where γ is the solution of the linear program. The following proposition indicates conditions under which the bound is tight.

Proposition 1.4.3 (Proposition 2 in [1]). *Assume that for all* $\mathbf{x}' \in \mathrm{X}_\mathcal{U}, m_Q(\mathbf{x}') > 0$ *and that there exists* $C \in (0, 1]$ *such that for all* $\gamma > 0$*:*

$$\pi(B_Q(\mathbf{x}') \neq y' \ \wedge \ m_Q(\mathbf{x}') = \gamma) \neq 0 \Rightarrow \\ \pi(B_Q(\mathbf{x}') \neq y' \ \wedge \ m_Q(\mathbf{x}') < \gamma) \geq C \cdot \pi(m_Q(\mathbf{x}') < \gamma). \quad (1.10)$$

Then, with probability at least $1 - \delta$ *it is true that:*

$$F_u^\delta(Q) - R_\mathcal{U}(B_Q) \leq \frac{1 - C}{C} R_\mathcal{U}(B_Q) + \frac{R_u^\delta(G_Q) - R_\mathcal{U}(G_Q)}{\gamma^*},$$

where

- $\gamma^* = \sup\{\gamma | \pi(B_Q(\mathbf{x}') \neq y' \ \wedge m_Q(\mathbf{x}') = \gamma) \neq 0\}$.
- $F_u^\delta(Q) = \inf_{\gamma \in (0,1]} \left\{\pi(m_Q(\mathbf{x}') < \gamma) + \frac{1}{\gamma}\left\lfloor K_u^\delta(Q) - M_Q^<(\gamma)\right\rfloor_+\right\}$.

This proposition states that if Condition (1.10) holds, the difference between the Bayes risk and its upper bound does not exceed an expression depending on the constant C. If we assume that the Gibbs risk bound is as tight as possible and the majority vote classifier makes most of its mistake on observations with low margin, we obtain that Condition (1.10) accepts high value C (close to 1), and the bound becomes tight. From theoretical point of view it is reasonable to assume that the majority vote classifier mistakes mostly on low margin region, since we suppose that in our hypothesis space we are able to approximate the true function ϕ in a good way.

1.5 Margin Based Binary Self-Learning Algorithm

In Section 1.4 one was discussed a transductive bound for Q-weighted majority vote classifier that was proposed by [1]. The main characteristic of this bound is that it makes use of information derived from the unlabelled set, namely, the margin value for each observation. Proposition 1.4.3 states that the bound is tight enough under certain conditions, so it can be considered as a good approximation of the majority vote risk.

Consequently, based on Theorem 1.4.2, [1] proposed the *margin-based self-learning algorithm* (further simply called the *self-learning algorithm*) that can be applied for a task with semi-supervised context. The main principle is to learn a model in ordinary supervised mode based on the training set that is augmented by pseudo-labelled observations. We find unlabelled examples that have the margin greater than a fixed threshold θ^*, pseudo-label them using a classifier and move them to the training set. Then, a new classifier is learnt using the augmented labelled set. The process is repeated until all observations will be pseudo-labelled or there will not be instances greater that θ^*. The most important point in this algorithm is how to determine the threshold θ^*. We would like to find θ^* that let pseudo-label some observations on the one hand, but tends to not add too much noise. In other words, we aim to minimise the Bayes risk conditionally given the margin is greater than a threshold $\theta > 0$.

Definition 1.5.1. *Given a threshold $\theta > 0$, the conditional Bayes error is defined as:*

$$R_{\mathcal{U}|\theta}(B_Q) := \pi(B_Q(\mathbf{x}') \neq y' | m_Q(\mathbf{x}') > \theta) = \frac{R_{\mathcal{U}\wedge\theta}(B_Q)}{\pi(m_Q(\mathbf{x}') > \theta)}. \tag{1.11}$$

Thus, at each step we find a threshold that minimises the conditional Bayes error, which can be viewed as labelling only observations with "good enough" confidence level. The self-learning algorithm is summarised in Algorithm 1. Generally speaking, the algorithm gains the performance of a basis classifier H as it complements the training set by additional examples that are pseudo-labelled with high confidence. If there are no such examples in the unlabelled set, the algorithm works the same as the initial classifier. It can be noticed that the SLA is a de facto inductive approach. However, information derived from the unlabelled set is used for learning. Moreover, the conditional Bayes error is determined in transductive manner; therefore, the algorithm is more oriented to find a solution for a transductive learning task.

Remark 1.5.2. *In Equation (*) we evaluate the joint Bayes risk according to Theorem 1.4.2 by:*

$$R_{\mathcal{U}\wedge\theta}(B_Q) \leq \inf_{\gamma\in(0,1]} \left\{ \pi(\theta < m_Q(\mathbf{x}') < \gamma) + \frac{1}{\gamma} \Big\lfloor K_u^{\delta}(Q) + M_{\overline{Q}}^{\leq}(\theta) - M_Q^{<}(\gamma) \Big\rfloor_+ \right\}.$$

Algorithm 1 Self-learning algorithm (SLA)

Input:
Labelled dataset $\mathrm{Z}_{\mathcal{L}}$
Unlabelled observations $\mathrm{X}_{\mathcal{U}}$
Initialisation:
A set of pseudo-labelled instances, $Z_{\mathcal{U}} \leftarrow \emptyset$
A classifier H trained on $\mathrm{Z}_{\mathcal{L}}$
repeat

1. Compute the margin threshold θ^* that minimises the conditional Bayes error:

$$\theta^* = \operatorname*{argmin}_{\theta\in(0,1]} R_{\mathcal{U}|\theta}(B_Q) = \operatorname*{argmin}_{\theta\in(0,1]} \frac{R_{\mathcal{U}\wedge\theta}(B_Q)}{\pi(m_Q(\mathbf{x}') > \theta)}. \tag{*}$$

2. $S \leftarrow \{(\mathbf{x}', y') | \mathbf{x}' \in \mathrm{X}_{\mathcal{U}}; [m_Q(\mathbf{x}') \geq \theta^*] \wedge [y' = sign(H(\mathbf{x}'))]\}$
3. $Z_{\mathcal{U}} \leftarrow Z_{\mathcal{U}} \cup S$, $\mathrm{X}_{\mathcal{U}} \leftarrow \mathrm{X}_{\mathcal{U}} \setminus S$
4. Learn a classifier H with a loss function:

$$\mathcal{L}(H, \mathrm{Z}_{\mathcal{L}}, Z_{\mathcal{U}}) = \frac{1}{l} \sum_{\mathbf{x}\in\mathrm{Z}_{\mathcal{L}}} e^{-yH(\mathbf{x})} + \frac{1}{|Z_{\mathcal{U}}|} \sum_{x'\in Z_{\mathcal{U}}} e^{-y'H(\mathbf{x}')}$$

until $\mathrm{X}_{\mathcal{U}}$ or S are $\emptyset$
Output: The final classifier H

In practice, to find an optimal θ^* and γ that provides infimum in Eq. 1.9 a grid search method is performed. It is a variant of the exhaustive search that looks for a minimum over the grid of values within the interval of interest. For optimisation of the conditional Bayes risk the interval is $(0, 1]$, while γ is optimised on $(\theta, 1]$.

Before going to the next section, we would mention that the self-learning algorithm can be considered as the improvement of the "classical" self-learning algorithm described in [23]. In the classical approach, the θ keeps to be a fixed value, which is less optimal strategy according to the results of the numerical experiments performed in [1].

1.6 Multi-class Framework and Confusion Matrix

In Section 1.3 and Section 1.4 it was given a review of PAC-Bayesian theory for the binary classification. In fact, the binary case is well studied in the literature, whereas there are just few works about multi-class classification. In this section and Section 1.7 we survey some of these works. We consider the same framework described in Section 1.1 for the inductive case, except the fact that the output space is $\mathcal{Y} = \{1, \dots, K\}$ now. In addition, we sat that a label y is drawn from the marginal distribution $P_Y(y)$ over $\mathcal{Y}$. We denote the vector of prior probabilities as $\mathbf{p} = (\mathrm{p}_1, \dots, \mathrm{p}_K)$, where $\mathrm{p}_k = P_Y(k),\ k = \{1, \dots, K\}$. It is natural to suppose that $l_k \geq 1$, where l_k is the number of observations in the training set that have the true label $k \in \{1, \dots, K\}$, and $\sum_{k=1}^{K} l_k = l$. We formulate a problem of multi-class classification as a task of minimising a generalisation error measure. In this paper, two error measures are considered, namely, error rate and confusion matrix.

Definition 1.6.1. *The* error rate *(also called the* misclassification error*) for a given hypothesis $h \in \mathcal{H}$ is defined as:*

$$\mathtt{E}(h) := \mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}(h(\mathbf{x}) \neq y) = \sum_{k=1}^{K} \mathrm{p}_k \mathbb{P}_{(\mathbf{x},k)\sim\mathcal{D}}(h(\mathbf{x}) \neq k).$$

Remark 1.6.2. *One can notice that the error rate is the multi-class extension of the risk function for the binary classification. We deliberately call it as the error rate in order to not mix up with the notion of conditional risk that will be defined further.*

The minimisation of the error rate is a classical approach in multi-class framework. However, it is not always the best error measure. There are many applications where it is important to take into account not only a fact of misclassification, but also a predicted label. For instance, it could be an application where misclassification between two classes i and $j\,(j \neq i)$ has a crucial impact on performance. In this case, it could be necessary to set a higher error weight each time when a classifier predicts the class j given the true class i. Another example, which takes place often in real applications, is imbalanced classification: a situation when the prior distribution $P_Y(y)$ is far from the uniform one. For all of them, it would be more appropriate to consider a confusion matrix as the error measure.

In [20] and [12] it is proposed to consider a confusion matrix that contains zero values on the main diagonal and misclassification conditional probabilities elsewhere.

Definition 1.6.3. *Given a classifier $h \in \mathcal{H}$, the* confusion matrix $\mathbf{C}_h = (c_{ij})_{i,j=\{1,\dots,K\}^2}$ *is defined as:*

$$\forall (i,j),\ c_{ij} := \begin{cases} 0 & \text{if } i = j \\ \mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}(h(\mathbf{x}) = j | y = i) & \text{otherwise.} \end{cases}$$

Thus, for qualitative prediction it would be reasonable to aim for a confusion matrix that is close to zero matrix as much as possible. Thereby, the goal of learning can be formulated as minimisation of the confusion matrix's norm. In [12] it is proposed a boosting algorithm that minimises the confusion matrix spectral norm. Experiment results show that for unbalanced data this method outperforms other boosting approaches.

Remark 1.6.4. *Sometimes, in literature, the confusion matrix is defined in a slightly different way. Namely, instead of zero values, the main diagonal contains conditional probabilities to correctly classify an element. Then, the confusion matrix $\mathbf{D}_h = (d_{ij})_{i,j=\{1,\dots,K\}^2}$ is written as:*

$$\forall (i,j) :\ d_{ij} := \mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}(h(\mathbf{x}) = j | y = i).$$

Definition 1.6.5. *Given a classifier $h \in \mathcal{H}$ and the training set $Z_{\mathcal{L}}$, the* empirical confusion matrix $\hat{\mathbf{C}}_h = (\hat{c}_{ij})_{i,j=\{1,\dots,K\}^2}$ *is defined as follows:*

$$\forall (i,j),\ \hat{c}_{ij} := \begin{cases} 0 & \text{if } i = j \\ \sum_{z=1}^{l} \frac{1}{l_{y_z}} \mathbb{1}_{h(\mathbf{x}_z)=j} \mathbb{1}_{y_z=i} & \text{otherwise,} \end{cases}$$

where l_{y_z} designates the number of training examples from the class that the observation z belongs to.

Now, we remind the definition of an operator matrix norm. Consider the normed vector spaces $\mathbb{R}^m$ and $\mathbb{R}^n$ with the norm $\|\cdot\|_p$, $p \geq 1$. Let $\mathbf{A}$ be a linear mapping $\mathbb{R}^m \to \mathbb{R}^n$ seen as a $n \times m$ matrix.

Definition 1.6.6. *Let $\mathbf{A}$ be an $n \times m$ matrix. Its operator norm of degree p is defined as:*

$$\|\mathbf{A}\|_p := \sup_{\substack{\mathbf{x}\in\mathbb{R}^m \\ \|\mathbf{x}\|_p=1}} \|\mathbf{A}\mathbf{x}\|_p = \sup_{\mathbf{x}\in\mathbb{R}^m} \frac{\|\mathbf{A}\mathbf{x}\|_p}{\|\mathbf{x}\|_p}$$

Proposition 1.6.7. *When $p = 2$, the matrix operator norm is called the* spectral norm. *In this case the norm of $\mathbf{A}$ corresponds to the matrix's largest singular value, which, in turn, is the square root of the largest eigenvalue for the matrix $\mathbf{A}^\intercal\mathbf{A}$:*

$$\|\mathbf{A}\|_2 = \sqrt{\lambda_{\max}(\mathbf{A}^\intercal\mathbf{A})}.$$

The following claim shows how the error rate can be represented through the confusion matrix. This leads to the fact that from "more general" structure such as the confusion matrix we can always come back to the more simple one, which is the error rate.

Claim 1.6.8. *For all $h \in \mathcal{H}$ the error rate can be expressed through the confusion matrix from Definition 1.6.3 in the following way:*

$$\mathtt{E}(h) = \left\|\mathbf{C}_h^\intercal\,\mathbf{p}\right\|_1,$$

where $\mathbf{p}$ is the prior distribution over $\mathcal{Y}$.

Proof. Multiplying the matrix $\mathbf{C}_h^\intercal$ by the vector $\mathbf{p}$, we obtain the following vector:

$$\mathbf{C}_h^\intercal\,\mathbf{p} = \left\{\sum_{i=1}^{K} c_{ij}\mathrm{p}_i\right\}_{j=1}^{K},$$

where c_{ij} is the (i,j)-entry of the matrix $\mathbf{C}_h$. Computing the 1-norm of the vector we deduce:

$$\left\|\mathbf{C}_h^\intercal\,\mathbf{p}\right\|_1 = \sum_{j=1}^{K}\sum_{i=1}^{K} c_{ij}\mathrm{p}_i = \sum_{i=1}^{K}\sum_{j=1}^{K} c_{ij}\mathrm{p}_i.$$

Notice that $\sum_{j=1}^{K} c_{ij} = \mathbb{P}_{(\mathbf{x},i)\sim\mathcal{D}}(h(\mathbf{x}) \neq i)$. From this we immediately derive the definition of the error rate:

$$\left\|\mathbf{C}_h^\intercal\,\mathbf{p}\right\|_1 = \sum_{i=1}^{K} \mathrm{p}_i\mathbb{P}_{(\mathbf{x},i)\sim\mathcal{D}}(h(\mathbf{x}) \neq i).$$

□

Remark 1.6.9. *According to [20], for a given classifier h, the error rate is bounded in the following way:*

$$\mathtt{E}(h) \leq \sqrt{K}\|\mathbf{C}_h\|_2.$$

As it can be seen the error rate of a classifier is bounded by the spectral norm of the classifier's confusion matrix multiplied by the number of classes. Thus, minimisation of the spectral norm leads to minimisation of the error rate, which is a good property. However, the converse does not take place. Therefore, the minimisation of the spectral norm does not give a desired result, if the goal is to get the error rate as small as possible in an application. In [12] it is shown that for balanced classification tasks the minimisation of the spectral norm does not bring any benefit.

To simplify notations, further, we write $\|\cdot\|$ for the spectral norm. In turn, the spectral norm of a confusion matrix is called the confusion matrix norm. Now, let's define the conditional risk, which is the probability to predict class j, when the true class is i.

Definition 1.6.10. *Given $(i,j) \in \{1,\dots,K\}^2$, $i \neq j$, a classifier $h \in \mathcal{H}$ and the training set $Z_{\mathcal{L}}$, we define the* conditional risk *as follows:*

$$R(h,i,j) := \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_{|y=i}} \mathbb{1}_{h(\mathbf{x})=j} = \mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}(h(\mathbf{x}) = j | y = i). \tag{1.12}$$

From Eq. 1.12 we can observe that the confusion matrix $\mathbf{C}_h = (c_{ij})_{i,j=\{1,\dots,K\}^2}$ can also be formulated as a matrix, where zeros are on the main diagonal and conditional risks are anywhere else:

$$\forall (i,j),\ c_{ij} := \begin{cases} 0 & \text{if } i = j \\ R(h,i,j) & \text{otherwise.} \end{cases}$$

1.7 Multi-class PAC-Bayesian Theory

This section reports significant results that are obtained for the PAC-Bayesian theory in the multi-class framework at the time of writing this master thesis. As before we define the Q-weighted majority vote and the Gibbs classifiers.

Definition 1.7.1. *The Q-*weighted majority vote classifier *(also called the* Bayes classifier*) in multiclass classification is defined as follows:*

$$B_Q(\mathbf{x}) := \underset{k\in\mathcal{Y}}{\operatorname{argmax}} \left[\mathbb{E}_{h\sim Q} \mathbb{1}_{h(\mathbf{x})=k}\right], \quad \forall \mathbf{x} \in \mathcal{X}. \tag{1.13}$$

Then, we define the true and empirical confusion matrices of the Q-weighted majority vote classifier respectively as $\mathbf{C}_{B_Q}$ and $\hat{\mathbf{C}}_{B_Q}$. The conditional risk of the majority vote classifier is designated as $R(B_Q,i,j)$.

Definition 1.7.2. *The* conditional Gibbs risk *is defined as:*

$$R(G_Q,i,j) := \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_{|y=i}} \mathbb{E}_{h\sim Q} \mathbb{1}_{h(\mathbf{x})=j}. \tag{1.14}$$

Definition 1.7.3. *The true and empirical confusion matrices of the Gibbs classifier are respectively defined as:*

$$\mathbf{C}_{G_Q} := \mathbb{E}_{h\sim Q} \mathbb{E}_{Z_{\mathcal{L}}\sim\mathcal{D}^l} \hat{\mathbf{C}}_h,$$
$$\hat{\mathbf{C}}_{G_Q} := \mathbb{E}_{h\sim Q} \hat{\mathbf{C}}_h.$$

Definition 1.7.4. *In multi-class framework, the* function m_Q *is defined for an observation $\mathbf{x} \in \mathcal{X}$ and a label $c \in \mathcal{Y}$ by:*

$$m_Q(\mathbf{x},c) := \mathbb{E}_{h\sim Q} \mathbb{1}_{h(\mathbf{x})=c} = \sum_{h:h(\mathbf{x})=c} Q(c).$$

From Definition 1.7.4 one can see that $m_Q(\mathbf{x}, c)$ corresponds to the expected vote that hypotheses give for the class c given the observation $\mathbf{x}$. In other words, it is the probability to assign the class c for the example $\mathbf{x}$. Hence, we derive that for any observation $\mathbf{x} \in \mathcal{X}$:

$$\sum_{c \in \mathcal{Y}} m(\mathbf{x}, c) = 1. \tag{1.15}$$

Definition 1.7.5. *In the multi-class framework, the margin function is defined in the following way:*

$$\mathcal{M}_Q(\mathbf{x}, y) := \mathbb{E}_{h \sim Q} \mathbb{1}_{h(\mathbf{x})=y} - \max_{c \in \mathcal{Y}, c \neq y} \{\mathbb{E}_{h \sim Q} \mathbb{1}_{h(\mathbf{x})=c} \cdot\} \tag{1.16}$$

Notice that the margin is signed, i.e. can be computed only when the true label y is known. Also, one can see that the margin is the difference between the value of m_Q for the true label and the maximal m_Q among other classes:

$$\mathcal{M}_Q(\mathbf{x}, y) = m_Q(\mathbf{x}, y) - \max_{c \in \mathcal{Y}, c \neq y} m_Q(\mathbf{x}, c).$$

Suppose that the majority vote classifier predicts the label correctly. Then, $m_Q(\mathbf{x}, y) = \max_{c \in \mathcal{Y}} m_Q(\mathbf{x}, c)$ and the margin characterises the difference between two maximal values of m_Q, which somehow indicates how much the classifier is sure. In the case of negative value of the margin, we deduce that the classifier makes a mistake and the value itself describes how much the classifier was wrong. This reasoning is followed by the next remark.

Remark 1.7.6. *It can be noticed that the error rate of the Q-weighted majority vote classifier can be represented as the probability to have a non-positive margin:*

$$\mathtt{E}(B_Q) = \mathbb{P}_{(\mathbf{x}, y) \sim \mathcal{D}}(\mathcal{M}_Q(x, y) \leq 0).$$

Further, a generalisation bound for the norm of the Gibbs confusion matrix introduced in [20] is detailed. There, a matrix concentration inequality from [22] for random self-adjoint matrices' sum are used. Generally speaking, self-adjointness does not hold for an arbitrary confusion matrix. Because of this, based on [21]'s results, [20] suggests to use for non-self-adjoint confusion matrices the following matrix dilation.

Definition 1.7.7. *The matrix dilation for the confusion matrix $\mathbf{C}$ is defined as:*

$$\mathcal{S}(\mathbf{C}) := \begin{pmatrix} \mathbf{0} & \mathbf{C} \\ \mathbf{C}^\intercal & \mathbf{0} \end{pmatrix}.$$

In contrast to $\mathbf{C}$, the matrix $\mathcal{S}(\mathbf{C})$ is self-adjoint. Moreover, according to [21], after dilation of $\mathbf{C}$, its spectral information is preserved. Thus, we have:

$$\|\mathcal{S}(\mathbf{C})\| = \|\mathbf{C}\|.$$

Now, we remind the [22]'s concentration inequality and its corollary proposed by [20].

Theorem 1.7.8 (Theorem 1.3 in [22])**.** *Consider a finite sequence $\{\mathbf{M}_i\}$ of random, independent, self-adjoint matrices of size $K \times K$, and let $\{\mathbf{A}_i\}$ be a sequence of constant self-adjoint matrices. Suppose each random matrix satisfies almost surely $\mathbb{E}\mathbf{M}_i = 0$ and $\mathbf{M}_i^2 \preceq \mathbf{A}_i^2$, where $\mathbf{A} \preceq \mathbf{B}$ denotes that $\mathbf{A}$ precedes $\mathbf{B}$ in the semi-definite order. Then, $\forall\ \epsilon \geq 0$:*

$$\mathbb{P}\left\{\lambda_{max}\left(\sum_i \mathbf{M}_i\right) \geq \epsilon\right\} \leq K\, e^{-\epsilon^2/8\sigma^2},$$

where $\sigma^2 := \|\sum_i \mathbf{A}_i^2\|$.

Corollary 1.7.9 (Corollary 3 in [20]). *Consider a finite sequence $\{\mathbf{M}_i\}_i$ of random, independent matrices of size $K \times K$, and let $\{a_i\}_i$ be a sequence of fixed scalars. Suppose each random matrix satisfies almost surely $\mathbb{E}\mathbf{M}_i = 0$ and $\|\mathbf{M}_i\| \leq a_i$. Then, $\forall\ \epsilon \geq 0$:*

$$\mathbb{P}\left\{\|\sum_i \mathbf{M}_i\| \geq \epsilon\right\} \leq 2K\, e^{-\epsilon^2/8\sigma^2},$$

where $\sigma^2 := \sum_i a_i^2$.

Finally, this corollary was applied in [20] to get an implicit bound for the Gibbs confusion matrix. The result is summarised in the following theorem.

Theorem 1.7.10 (Theorem 2 in [20]). *For every prior distribution P over $\mathcal{H}$ and any $\delta \in (0, 1]$, we have:*

$$\mathbb{P}_{Z_{\mathcal{L}} \sim \mathcal{D}^l}\left\{\forall Q \text{ on } \mathcal{H},\ \|\hat{\mathbf{C}}_{G_Q} - \mathbf{C}_{G_Q}\| \leq \sqrt{\frac{8K}{l_- - 8K}\left[KL(Q \parallel P) + \ln\frac{l_-}{4\delta}\right]}\right\} \geq 1 - \delta,$$

where $l_- = \min_{i=\{1,\ldots,K\}} l_i$.

It can be seen that this bound is implicit, so it makes harder to use it in practice. To prevent this issue, the following corollary deduces the explicit version of the bound, leading unfortunately to a less tight bound.

Corollary 1.7.11 (Corollary 1 in [20]). *Given Theorem 1.7.10, we have:*

$$\mathbb{P}_{Z_{\mathcal{L}} \sim \mathcal{D}^l}\left\{\forall Q \text{ on } \mathcal{H},\ \|\mathbf{C}_{G_Q}\| \leq \|\hat{\mathbf{C}}_{G_Q}\| + \sqrt{\frac{8K}{l_- - 8K}\left[KL(Q \parallel P) + \ln\frac{l_-}{4\delta}\right]}\right\} \geq 1 - \delta.$$

Based on this result, a bound for the confusion matrix norm of the majority vote classifier can be deduced. The following proposition and corollary set the connection between the Gibbs and Q-weighted majority vote classifiers.

Proposition 1.7.12 (Proposition 1 in [20]). *The true conditional risk of the Q-weighted majority vote classifier and the conditional Gibbs risk have the following relation:*

$$\forall (i, j),\ R(B_Q, i, j) \leq KR(G_Q, i, j).$$

Corollary 1.7.13 (Corollary 2 in [20]). *The true confusion matrix of the q-weighted majority vote classifier $\mathbf{C}_{B_Q}$ and the Gibbs confusion matrix are related in the following way:*

$$\left\|\mathbf{C}_{B_Q}\right\| \leq K\left\|\mathbf{C}_{G_Q}\right\|.$$

As it can be seen, Proposition 1.7.12 and Corollary 1.7.13 extend Lemma 4.1 from [14] for the multi-class framework. Thus, the confusion matrix's norm of the voted classifier is bounded by the norm of the Gibbs confusion matrix multiplied by the number of classes. Therefore, an upper bound on the latter norm implies the corresponding upper bound on the former one.

Unfortunately, this bound can not be considered as tight. So far, there is no yet any alternative bound for the confusion matrix norm of the majority vote classifier. However, in [15] a bound for the Bayes error rate has been proposed . It is called the C-bound, and it is based on the first two statistical moments of the margin function defined by Eq. (1.16). The following theorem summarises this result.

Theorem 1.7.14 (Theorem 3 in [15]). *For every posterior distribution Q over $\mathcal{H}$, for any distribution $\mathcal{D}$ on $\mathcal{X} \times \mathcal{Y}$ such that $\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\mathcal{M}_Q(\mathbf{x}, y) > 0$, it is true that:*

$$\mathtt{E}(B_Q) = \mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}(\mathcal{M}_Q(x, y) \leq 0) \leq 1 - \frac{\left[\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\,\mathcal{M}_Q(\mathbf{x}, y)\right]^2}{\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[\mathcal{M}_Q(\mathbf{x}, y)\right]^2}.$$

Chapter 2

Transductive Semi-supervised Bounds: Extension to the Multi-class Case

2.1 Problem Statement

In Chapter 1 a review of past works in PAC-Bayesian theory and its application to the semi-supervised learning has been given. Most of them concern the binary classification problem, whereas just few ones are devoted to the multi-class framework. In this chapter, we propose an extension of [1]'s result, which is a transductive bound for the majority vote classifier, when $|\mathcal{Y}| \geq 2$. To the best of our knowledge, this is the first attempt to bound the majority vote classifier in the multi-class transductive setting. We extend the work of [1] by means the error rate representation through the confusion matrix as in [20]. Through this, we are able to propose a bound for each non-zero entry of the confusion matrix, which is the conditional risk.

Let us remind the framework that we are going to work with. In our study, we consider the multi-class, mono-label classification problem with $\mathcal{Y} = \{1, \ldots, K\}$, $K \geq 2$. Let $\mathcal{X} \subset \mathbb{R}^d$ be the input space. Designate $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$. Consider a set of labelled examples $\mathrm{Z}_{\mathcal{L}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$, $\mathrm{Z}_{\mathcal{L}} \in \mathcal{Z}^l$ as well as a set of unlabelled observations $\mathrm{X}_{\mathcal{U}} = \{\mathbf{x}'_i\}_{i=l+1}^{l+u}$, $\mathrm{X}_{\mathcal{U}} \in \mathcal{X}^u$. We suppose that all pairs $(\mathbf{x}, y)$ from $\mathrm{Z}_{\mathcal{L}}$ are i.i.d. with a joint probability $\mathcal{D}$, and $\mathbf{x}$ are distributed according to $P_X(\mathbf{x})$ over $\mathcal{X}$. Furthermore, we suppose that there is a fixed class of prediction functions $\mathcal{H} = \{h | h : \mathcal{X} \to \mathcal{Y}\}$ that is called the hypothesis space.

We treat the problem by the transductive inference. Then, we suppose that the labelled $\mathrm{Z}_{\mathcal{L}}$ and the unlabelled $\mathrm{X}_{\mathcal{U}}$ sets were obtained according to Vapnik's setting 1 described in Section 1.1. We consider the deterministic case when there exists a function $\phi : \mathcal{X} \to \mathcal{Y}$ that labels examples by $y = \phi(x)$. It implies that for any $\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}$ there is one and only one possible label y'.

As in Chapter 1, we define the Q-weighted majority vote (Bayes) classifier in the following way:

$$B_Q(\mathbf{x}) := \underset{c \in \mathcal{Y}}{\operatorname{argmax}} \left[\mathbb{E}_{h \sim Q} \mathbb{1}_{h(\mathbf{x})=c} \right], \quad \forall \mathbf{x} \in \mathcal{X}. \tag{2.1}$$

The goal of learning is to choose the posterior distribution Q over $\mathcal{H}$ that minimises the transductive error rate of the majority vote classifier.

Definition 2.1.1. *The* transductive error rate *is defined in the following way:*

$$\mathtt{E}_{\mathcal{U}}(h) := \frac{1}{u} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{h(\mathbf{x}') \neq y'},$$

where y' is the true unknown class label of $\mathbf{x}'$.

As before, we consider the majority vote classifier together with the associated Gibbs classifier, which transductive error rate is defined in the following way:

$$\mathtt{E}_{\mathcal{U}}(G_Q) := \frac{1}{u} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{E}_{h \sim Q} \left[\mathbb{1}_{h(\mathbf{x}') \neq y'} \right]. \tag{2.2}$$

Next, we introduce notions of the conditional risk and the confusion matrix for the transductive case, i.e. defined on the unlabelled set $\mathrm{X}_{\mathcal{U}}$. The advantage of the confusion matrix as an error measure is that it provides a richer information compared to the error rate, which does not describe the dispersion of errors regarding each class over all the others.

Definition 2.1.2. *Given a classifier h, for each class pair $(i, j) \in \{1, \ldots, K\}^2$ s.t. $i \neq j$, we define the* transductive conditional risk $R_{\mathcal{U}}$ *in the following way:*

$$R_{\mathcal{U}}(h, i, j) := \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{h(\mathbf{x}')=j} \mathbb{1}_{y'=i},$$

where $u_i = \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{y'=i}$ is the size of class $i \in \mathcal{Y}$.

Definition 2.1.3. *For a classifier h, the* transductive confusion matrix $\mathbf{C}_h^{\mathcal{U}} = (c_{ij})_{i,j=\{1,\ldots,K\}^2}$ *is defined as follows:*

$$c_{ij} := \begin{cases} 0 & i = j \\ R_{\mathcal{U}}(h, i, j) & i \neq j \end{cases}.$$

Remark 2.1.4. *In the case of the Q-weighted majority vote classifier, the matrix $\mathbf{C}_{B_Q}^{\mathcal{U}} := (c_{ij})_{i,j=\{1,\ldots,K\}^2}$ has the following form:*

$$c_{ij} = \begin{cases} 0 & i = j \\ R_{\mathcal{U}}(B_Q, i, j) & i \neq j \end{cases},$$

with $B_Q(\mathbf{x}') = \operatorname{argmax}_{c \in \mathcal{Y}} \left[\mathbb{E}_{h \sim Q} \mathbb{1}_{h(\mathbf{x}')=c} \right]$.

Definition 2.1.5. *For the Gibbs classifier the* transductive confusion matrix *is defined as:*

$$\mathbf{C}_{G_Q}^{\mathcal{U}} := \mathbb{E}_{h \sim Q} \mathbf{C}_h^{\mathcal{U}}.$$

Similarly to the inductive setting (Claim 1.6.8) we can express the error rate through the confusion matrix. This lets us to expand the error rate to the more detailed confusion matrix. Conversely, when it is needed, we can always come back from the notion of the confusion matrix to the error rate that is the target functional of our learning problem.

Remark 2.1.6. *The error rate and the confusion matrix are connected in the following way:*

$$\mathtt{E}_{\mathcal{U}}(h) = \frac{1}{u} \sum_{i=1}^{K} u_i \sum_{\substack{j=1 \\ j \neq i}}^{K} R_{\mathcal{U}}(h, i, j) = \left\| (\mathbf{C}_h^{\mathcal{U}})^{\intercal} \mathbf{p} \right\|_1,$$

where $\mathbf{p} = \{u_i / u\}_{i=1}^K$.

Proof. The same kind of proof as for the inductive case (see Claim 1.6.8). □

As in Section 1.7, we take into consideration the function m_Q:

$$m_Q(\mathbf{x}, y) = \mathbb{E}_{h \sim Q} \mathbb{1}_{h(\mathbf{x})=y} = \sum_{h : h(\mathbf{x})=y} Q(y). \tag{2.3}$$

Further, we use $\mathbf{m_x}$ to denote the vector $(m_Q(\mathbf{x}, c))_{c=1}^K$. It can be noticed that this vector does not contain the information about the true label. Because of that, in this part we call $\mathbf{m_x}$ the *unsigned margin* of the observation $\mathbf{x}$. In turn, in this work, $m_Q(\mathbf{x}, c)$ is sometimes called the *unsigned margin for the class* c of the observation $\mathbf{x}$. This is mostly motivated by a wish to make our reasoning resemble the binary case. However, in the binary case the unsigned margin (Definition 1.2.8) has a different representation, since its value tells us how confident we are in classification. If we look at the value $m_Q(\mathbf{x}, c)$ for a fixed $c \in \{1, \ldots, K\}$, then we can not say for sure how confident we are, since we have to compare this value with margins for the other classes. From this point, $\mathbf{m_x}$ is more appropriate to be called the margin as it contains the whole information and, if it is needed, can be reduced to a scalar confidence score.

Nevertheless, the reader will subsequently find out that in our work there is no necessity to derive one confidence value from an observation. Indeed, we take more interest in comparison between the margins of different observation for one class. In other words, for each class we look at its margin distribution over the unlabelled set. In this sense, we say that if $m_Q(\mathbf{x}', c) > m_Q(\mathbf{x}'', c)$, then hypotheses give to $\mathbf{x}'$ a higher vote to be classified in the class c than to $\mathbf{x}''$. Hence, we are more sure that $\mathbf{x}'$ rather than $\mathbf{x}''$ will be labelled as c.

Following this reasoning, we infer that the notion of a confident margin may differ for each class. Indeed, each class can have a different margin distribution over the unlabelled set. Because of this, we extend the notion of the joint Bayes risk to the multi-class case in a way that it depends on a vector of parameters rather than on a scalar one.

Definition 2.1.7. *Given a vector* $\boldsymbol{\theta} = (\theta_c)_{c=1}^K,\ \boldsymbol{\theta} \in [0,1]^K$, *the* transductive joint Bayes conditional risk $R_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q, i, j)$ *for the class pair* $(i, j) \in \{1, \ldots, K\}^2$ *s.t.* $i \neq j$,, *is defined as follows:*

$$R_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q, i, j) := \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_\mathcal{U}} \mathbb{1}_{B_Q(\mathbf{x}')=j} \mathbb{1}_{y'=i} \mathbb{1}_{m_Q(\mathbf{x}', j) \geq \theta_j},$$

where $u_i = \sum_{\mathbf{x}' \in \mathrm{X}_\mathcal{U}} \mathbb{1}_{y'=i}$.

Thus, the transductive joint Bayes conditional risk counts an example as erroneous, if its true label is i and the majority vote classifier predicts the class j with the margin $m_Q(\mathbf{x}', j) \geq \theta_j$. Similarly to [1], we assume that the majority vote classifier makes an error predicting the label j mostly on examples with a low value of $m_Q(\mathbf{x}', j)$. Then, if θ_j is high enough, then the joint conditional risk computes the probability to make a mistake on "confident" observations.

Definition 2.1.8. *The* transductive joint Bayes confusion matrix $\mathbf{C}_{B_Q}^{\mathcal{U}\wedge\boldsymbol{\theta}} = (c_{ij})_{i,j=\{1,\ldots,K\}^2}$ *given a vector* $\boldsymbol{\theta} = (\theta_n)_{n=1}^K,\ \boldsymbol{\theta} \in [0,1]^K$ *is defined as:*

$$c_{ij} := \begin{cases} 0 & i = j \\ R_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q, i, j) & i \neq j \end{cases}.$$

Definition 2.1.9. *Given a vector* $\boldsymbol{\theta} = (\theta_n)_{n=1}^K,\ \boldsymbol{\theta} \in [0,1]^K$, *the* transductive joint Bayes error rate $\mathtt{E}_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q)$ *is defined in the following way:*

$$\mathtt{E}_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q) := \frac{1}{u} \sum_{\mathbf{x}' \in \mathrm{X}_\mathcal{U}} \mathbb{1}_{k \neq y'} \mathbb{1}_{m_Q(\mathbf{x}', k) \geq \theta_k},$$

where $k := B_Q(\mathbf{x}')$.

Proposition 2.1.10. *The transductive joint Bayes error rate can be represented through the transductive joint Bayes confusion matrix in the following way:*

$$\mathtt{E}_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q) = \left\| \left(\mathbf{C}_{B_Q}^{\mathcal{U}\wedge\boldsymbol{\theta}} \right)^\intercal \mathbf{p} \right\|_1,$$

where $\mathbf{p} = \{u_i/u\}_{i=1}^K$.

Proof. First, it can be noticed that

$$\forall \mathbf{x}' \in \mathrm{X}_{\mathcal{U}},\ \mathbb{1}_{B_Q(\mathbf{x}')\neq y'} = \sum_{i=1}^{K}\sum_{\substack{j=1\\ j\neq i}}^{K} \mathbb{1}_{B_Q(\mathbf{x}')=j}\mathbb{1}_{y'=i}.$$

From this equality we can deduce that

$$\forall \mathbf{x}' \in \mathrm{X}_{\mathcal{U}},\ \mathbb{1}_{k\neq y'}\mathbb{1}_{m_Q(\mathbf{x}',k)\geq\theta_k} = \sum_{i=1}^{K}\sum_{\substack{j=1\\ j\neq i}}^{K} \mathbb{1}_{B_Q(\mathbf{x}')=j}\mathbb{1}_{y'=i}\mathbb{1}_{m_Q(\mathbf{x}',j)\geq\theta_j},$$

where $k := B_Q(\mathbf{x}')$. Finally, we conclude:

$$\begin{aligned}
\mathtt{E}_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q) &= \frac{1}{u}\sum_{\mathbf{x}'\in\mathrm{X}_{\mathcal{U}}} \mathbb{1}_{k\neq y'}\mathbb{1}_{m_Q(\mathbf{x}',k)\geq\theta_k} \\
&= \frac{1}{u}\sum_{\mathbf{x}'\in\mathrm{X}_{\mathcal{U}}}\sum_{i=1}^{K}\sum_{\substack{j=1\\ j\neq i}}^{K} \mathbb{1}_{B_Q(\mathbf{x}')=j}\mathbb{1}_{y'=i}\mathbb{1}_{m_Q(\mathbf{x}',j)\geq\theta_j} \\
&= \sum_{i=1}^{K}\frac{u_i}{u}\sum_{\substack{j=1\\ j\neq i}}^{K}\frac{1}{u_i}\sum_{\mathbf{x}'\in\mathrm{X}_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}')=j}\mathbb{1}_{y'=i}\mathbb{1}_{m_Q(\mathbf{x}',j)\geq\theta_j} \\
&= \sum_{i=1}^{K}\frac{u_i}{u}\sum_{\substack{j=1\\ j\neq i}}^{K} R_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q,i,j) \\
&= \left\| \left(\mathbf{C}_{B_Q}^{\mathcal{U}\wedge\boldsymbol{\theta}}\right)^{\mathsf{T}} \mathbf{p} \right\|_1 .
\end{aligned}$$

□

2.2 Transductive Bounds for the Majority Vote Classifier in the Multi-class Framework

In this section we propose a transductive bound for the majority vote classifier in multi-class setting. The bound is based on the margin distribution as well as a bound of the Gibbs risk, which we suppose given. First, we give a theorem that provides a bound for the Bayes conditional risk. Two lemmas precede this theorem. The first one connects the conditional Gibbs risk and the conditional joint Bayes risk in a similar way as Lemma 1.4.1. The second one provides an analytic solution of a linear program. Then, two corollaries of the theorem are given that propose upper bounds for the Bayes confusion matrix and the Bayes error rate. Finally, we propose a setting under which the bound on the conditional Bayes risk becomes tight.

Lemma 2.2.1. *Let* $\Gamma_c = \{\gamma_c |\ \exists\ \mathbf{x}' \in \mathrm{X}_{\mathcal{U}} : \gamma_c = m_Q(\mathbf{x}', c)\}$*, where* $c \in \mathcal{Y}$*. Let enumerate its elements such that they form an ascending order:*

$$\gamma_c^{(1)} \leq \gamma_c^{(2)} \leq \cdots \leq \gamma_c^{(N_c)},$$

where $N_c := |\Gamma_c|$*. Denote* $b_{i,j}^{(n)} := \frac{1}{u_i}\sum_{\mathbf{x}'\in\mathrm{X}_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}')=j}\mathbb{1}_{y'=i}\mathbb{1}_{m_Q(\mathbf{x}',j)=\gamma_j^{(n)}}$*. Then,* $\forall (i,j) \in \mathcal{Y}^2$*:*

$$R_{\mathcal{U}}(G_Q,i,j) = \sum_{n=1}^{N_j} b_{i,j}^{(n)}\gamma_j^{(n)} + \varepsilon_{i,j}, \tag{2.4}$$

where $\varepsilon_{i,j} = \frac{1}{u_i}\sum_{\mathbf{x}'\in X_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}')\neq j}\mathbb{1}_{y'=i} m_Q(\mathbf{x}', j)$. *In addition, we have:*

$$R_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q, i, j) = \sum_{n=k+1}^{N_j} b_{i,j}^{(n)} \tag{2.5}$$

with $k = \begin{cases} 0 & \text{if } \{n|\gamma_j^{(n)} < \theta_j\} = \emptyset \\ \max\{n|\gamma_j^{(n)} < \theta_j\} & \text{otherwise.} \end{cases}$

Proof. First, we obtain the formula (2.4):

$$\begin{aligned}
R_{\mathcal{U}}(G_Q, i, j) &= \frac{1}{u_i}\mathbb{E}_{h\sim Q}\sum_{\mathbf{x}'\in X_{\mathcal{U}}} \mathbb{1}_{h(\mathbf{x}')=j}\mathbb{1}_{y'=i} \\
&= \frac{1}{u_i}\sum_{\mathbf{x}'\in X_{\mathcal{U}}} \mathbb{1}_{y'=i}\mathbb{E}_{h\sim Q}\mathbb{1}_{h(\mathbf{x}')=j} \\
&= \frac{1}{u_i}\sum_{\mathbf{x}'\in X_{\mathcal{U}}} \mathbb{1}_{y'=i} m_Q(\mathbf{x}', j) \\
&= \frac{1}{u_i}\sum_{\mathbf{x}'\in X_{\mathcal{U}}} \mathbb{1}_{y'=i}\mathbb{1}_{B_Q(\mathbf{x}')=j} m_Q(\mathbf{x}', j) + \frac{1}{u_i}\sum_{\mathbf{x}'\in X_{\mathcal{U}}} \mathbb{1}_{y'=i}\mathbb{1}_{B_Q(\mathbf{x}')\neq j} m_Q(\mathbf{x}', j) \\
&= \frac{1}{u_i}\sum_{\mathbf{x}'\in X_{\mathcal{U}}}\sum_{n=1}^{N_j} \mathbb{1}_{y'=i}\mathbb{1}_{B_Q(\mathbf{x}')=j}\mathbb{1}_{m_Q(\mathbf{x}',j)=\gamma_j^{(n)}}\gamma_j^{(n)} + \varepsilon_{i,j} \\
&= \sum_{n=1}^{N_j} b_{i,j}^{(n)}\gamma_j^{(n)} + \varepsilon_{i,j},
\end{aligned}$$

Then, we deduce the formula (2.5):

$$\begin{aligned}
R_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q, i, j) &= \frac{1}{u_i}\sum_{\mathbf{x}'\in X_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}')=j}\mathbb{1}_{y'=i}\mathbb{1}_{m_Q(\mathbf{x}',j)\geq\theta_j} \\
&= \frac{1}{u_i}\sum_{n=1}^{N_j}\sum_{\mathbf{x}'\in X_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}')=j}\mathbb{1}_{y'=i}\mathbb{1}_{m_Q(\mathbf{x}',j)=\gamma_j^{(n)}}\mathbb{1}_{\gamma_j^{(n)}\geq\theta_j} \\
&= \frac{1}{u_i}\sum_{n=k+1}^{N_j}\sum_{\mathbf{x}'\in X_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}')=j}\mathbb{1}_{y'=i}\mathbb{1}_{m_Q(\mathbf{x}',j)=\gamma_j^{(n)}} \\
&= \sum_{n=k+1}^{N_j} b_{i,j}^{(n)}.
\end{aligned}$$

□

Next, we remind Lemma 4 from [1] that gives a solution of a linear program. We deliberately place this lemma in this chapter, not in the previous one, as it plays a crucial role in the theorem we are going to derive.

Lemma 2.2.2 (Lemma 4 in [1])**.** *Let* $(g_i)_{i\in\{1,\ldots,N\}}$ *be such that* $0 < g_1 < g_2 < \cdots < g_{N-1} < g_N \leq 1$. *Consider also* $p_i \geq 0$, $i = 1, \ldots, N$, $B \geq 0$, $k \in \{1, \ldots, N\}$. *Then, the optimal solution of the linear program:*

$$\begin{cases} \max_{q:=(q_1,\ldots,q_N)} F(q) := \max_{q_1,\ldots,q_N} \sum_{i=k+1}^{N} q_i \\ 0 \leq q_i \leq p_i \quad \forall i \in \{1, \ldots, N\} \\ \sum_{i=1}^{N} q_i g_i \leq B \end{cases}$$

will be q^* *defined as* $\forall i \leq k:\ q_i^* = 0$, $\forall i > k:\ q_i^* = \min\left(p_i, \left\lfloor \frac{B - \sum_{j<i} q_j^* g_j}{g_i} \right\rfloor_+\right)$.

Proof. It can be seen that the first k target variables should be zero for the optimal solution. Indeed, they do not influence explicitly the target function F. However, terms $g_i q_i$ for $i \in \{1, \dots, k\}$ are positive, so their increase leads to smaller values of q_i for $i \in \{k+1, \dots, N\}$, which in their turn decrease the value of F. Because of this, we look for a solution in a space $\mathcal{O} = \{0\}^k \times \prod_{i=k+1}^{N} [0, p_i]$. We aim to show that there is a unique optimal solution q^* in $\mathcal{O}$.

Existence. It is known that the linear program under consideration is a convex, feasible and bounded task. Hence, there is a feasible optimal solution $q^{opt} \in \prod_{i=1}^{N} [0, p_i]$. Then, we define $q^{opt,\mathcal{O}} \in \mathcal{O}$:

$$\begin{cases} q_i^{opt,\mathcal{O}} = q_i^{opt} & \text{if } i > k \\ q_i^{opt,\mathcal{O}} = 0 & \text{otherwise.} \end{cases}$$

It can be seen that this solution is feasible: $F(q^{opt,\mathcal{O}}) = F(q^{opt})$. Then, there exists an optimal solution in $\mathcal{O}$. Further, the optimal solution is again designated as q^*.

Unique representation. We would like to find a representation of q^* that is, in fact, unique. Before doing it, one can notice that for q^* the following equation is necessarily true:

$$\sum_{i=1}^{N} q_i^* g_i = B.$$

Indeed, as g_i are fixed, q^* would not be optimal otherwise, and there would exist $\tilde{q}$ such that $\sum_{i=1}^{N} \tilde{q}_i g_i > \sum_{i=1}^{N} q_i^* g_i$, which implies $F(\tilde{q}) > F(q^*)$.

Let's consider the lexicographic order $\succeq$:

$$\forall (q, q') \in \mathbb{R}^N \times \mathbb{R}^N, q \succeq q' \Leftrightarrow \left\{\mathcal{I}(q', q) = \emptyset\right\} \vee \left\{\mathcal{I}(q', q) \neq \emptyset \wedge \min\left(\mathcal{I}(q, q')\right) < \min\left(\mathcal{I}(q', q)\right)\right\},$$

where $\mathcal{I}(q', q) = \{i | q'_i > q_i\}$.

We aim to show that the optimal solution is actually the greatest feasible solution in $\mathcal{O}$ for $\succeq$. Let $\mathcal{M}$ be the set$\{i > k | q_i^* < p_i\}$. Then, there are two cases:

- $M = \emptyset$. It means that for all $i > k$, $q_i^* = p_i$ and q^* is then the maximal element for $\succeq$ in $\mathcal{O}$.
- $M \neq \emptyset$. Let's consider $K = \min\{i > k | q_i^* < p_i\}$, $M = \mathcal{I}(q, q^*)$. By contradiction, suppose q^* is not the greatest feasible solution for $\succeq$ and there is $q \in \mathbb{R}^N$ such that $q \succ q^*$.
  1. $M \leq k$. Then, $q_M > q_M^* = 0$. It implies that $q \notin \mathcal{O}$.
  2. $k < M < K$. Then, $q_M > q_M^* = p_M$. The same, $q \notin \mathcal{O}$.
  3. $M \geq K$. Then, $F(q) > F(q^*)$. But it means that $\sum_{i=1}^{N} q_i g_i > \sum_{i=1}^{N} q_i^* g_i = B$.

Hence, we conclude that if the solution is optimal then it is necessarily the greatest feasible solution for $\succeq$. Let's prove that if a solution is not the greatest feasible one then it can not be optimal. With this statement, uniqueness would be proven.

Consider $q \in \mathcal{O}$ such that $q^* \succ q$.

- $\mathcal{I}(q, q^*) = \emptyset$. Then, $F(q^*) > F(q)$ and q is not optimal.
- $\mathcal{I}(q, q^*) \neq \emptyset$. Let $K = \min\left(\mathcal{I}(q^*, q)\right)$ and $M = \min\left(\mathcal{I}(q, q^*)\right)$. Then, $q_M > q_M^* \geq 0$ and $K < M$. Denote $\lambda = \min\left(q_M, \frac{g_M}{g_K}(p_K - q_K)\right)$ and define q' by:

$$q'_i = q_i, \; i \notin \{K, M\}, \quad q'_K = q_K + \frac{g_M}{g_K}\lambda \quad q'_M = q_M - \lambda$$

It can be observed that q' satisfies the box constraints. Moreover, $F(q') = F(q) + \lambda(g_M/g_K - 1) > F(q)$ since $g_K < g_M$ and $\lambda > 0$. Thus, q is not optimal. Summing up, it is proven that there is the only optimal solution in $\mathcal{O}$ and it is the greatest feasible one for $\succeq$.

Then, let's obtain an explicit representation of this solution. As it is the greatest one in lexicographical order, we assign q_i for $i > k$ to maximal feasible values, which are p_i. It continues until the moment when $\sum_{j=1}^{i} q_i g_i$ is close to B. Denote by I the index such that $\sum_{i=1}^{I-1} p_i g_i \leq B$, but $\sum_{i=1}^{I} p_i g_i \geq B$.

- $\sum_{i=1}^{I-1} p_i g_i = B$. Then, $q_i = 0$ for $i \geq I$. It can be also written in the following way:

$$q_i = \left\lfloor \frac{B - \sum_{j<i} q_j g_j}{g_i} \right\rfloor_+, \qquad i \geq I$$

.

- $\sum_{i=1}^{I-1} p_i g_i < B$. Then, q_I is equal to residual:

$$q_I = \frac{B - \sum_{j<I} q_j g_j}{g_I} = \left\lfloor \frac{B - \sum_{j<I} q_j g_j}{g_I} \right\rfloor_+.$$

For the other q_i, $i > I$ we assign to 0.

□

Finally, we formulate the theorem that proposes an upper bound for the Bayes and joint Bayes conditional risks based on an upper bound of the Gibbs conditional risk. Then, from this theorem the corresponding upper bounds for the confusion matrix and error rate are deduced.

Theorem 2.2.3. *Consider B_Q as in (2.1). Suppose an upper bound of the transductive conditional Gibbs risk $R_u^\delta(G_Q, i, j)$ that holds with probability $1 - \delta$ is given. Follow the denotations of Lemma 2.2.1. Then for any Q and $\forall \delta \in (0, 1]$, $\forall \boldsymbol{\theta} \in [0, 1]^K$ with probability at least $1 - \delta$ we have:*

$$R_{\mathcal{U}}(B_Q, i, j) \leq \inf_{\gamma \in [0,1]} \left\{ I_{i,j}^{(\leq,<)}(0, \gamma) + \frac{1}{\gamma} \left\lfloor (K_{i,j}^\delta - M_{i,j}^<(\gamma)) \right\rfloor_+ \right\} \tag{2.6}$$

$$R_{\mathcal{U} \wedge \boldsymbol{\theta}}(B_Q, i, j) \leq \inf_{\gamma \in [\theta_j, 1]} \left\{ I_{i,j}^{(\leq,<)}(\theta_j, \gamma) + \frac{1}{\gamma} \left\lfloor (K_{i,j}^\delta - M_{i,j}^<(\gamma) + M_{i,j}^<(\theta_j)) \right\rfloor_+ \right\}, \tag{2.7}$$

where

- $K_{i,j}^\delta = R_u^\delta(G_Q, i, j) - \varepsilon_{i,j}$.
- $I_{i,j}^{(\lhd_1, \lhd_2)}(t, s) = \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{y'=i} \mathbb{1}_{t \lhd_1 m_Q(\mathbf{x}', j) \lhd_2 s}$, $(\lhd_1, \lhd_2) \in \{<, \leq\}^2$.
- $M_{i,j}^<(t) = \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{y'=i} \mathbb{1}_{m_Q(\mathbf{x}', j) < t} \, m_Q(\mathbf{x}', j)$.
- $\lfloor x \rfloor_+ = x \cdot \mathbb{1}_{x>0}$.

Proof. One can notice that with the fact that $M_{i,j}^<(0) = 0$, Eq. (2.6) can be directly obtained from Eq. (2.7):

$$R_{\mathcal{U}}(B_Q, i, j) = R_{\mathcal{U} \wedge \mathbf{0}_K}(B_Q, i, j),$$

where $\mathbf{0}_K$ is the K-size vector of zeros.

We would like to find an upper bound for the joint Bayes conditional risk. Hence, $\forall (i, j)$, $\forall \boldsymbol{\theta}$, we consider the case when the mistake is maximised. Then, using Lemma 2.2.1:

$$R_{\mathcal{U} \wedge \boldsymbol{\theta}}(B_Q, i, j) = \sum_{n=k}^{N_j} b_{i,j}^{(n)} \leq \max_{b_{i,j}^{(1)}, \ldots, b_{i,j}^{(N_j)}} \sum_{n=k}^{N_j} b_{i,j}^{(n)},$$

with $k = \begin{cases} 0 & \text{if } \{n|\gamma_j^{(n)} < \theta_j\} = \emptyset \\ \max\{n|\gamma_j^{(n)} < \theta_j\} & \text{otherwise} \end{cases}$. Then, it can be noticed that

$$0 \leq b_{i,j}^{(n)} \leq \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{y'=i} \mathbb{1}_{m_Q(\mathbf{x}',j)=\gamma_j^{(n)}}.$$

Consider the upper bound $R_u^{\delta}(G_Q, i, j)$ of the Gibbs conditional risk $R_{\mathcal{U}}(G_Q, i, j)$ that holds with probability $1 - \delta$. Then, from Eq. (2.4) we can derive:

$$\sum_{n=1}^{N_j} b_n^{(j)} \gamma_j^{(n)} = R_{\mathcal{U}}(G_Q, i, j) - \varepsilon_{i,j} \leq R_u^{\delta}(G_Q, i, j) - \varepsilon_{i,j}.$$

Denote $K_{i,j}^{\delta} = R_u^{\delta}(G_Q, i, j) - \varepsilon_{i,j}$ and $B_{i,j}^{(n)} = \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{y'=i} \mathbb{1}_{m_Q(\mathbf{x}',j)=\gamma_j^{(n)}}$. Thus, we can formulate the following linear program task:

$$\begin{array}{ll} \max\limits_{b_{i,j}^{(1)}, \ldots, b_{i,j}^{(N_j)}} & \sum\limits_{n=k}^{N_j} b_{i,j}^{(n)} \\ \text{s.t.} & \forall n,\ 0 \leq b_{i,j}^{(n)} \leq B_{i,j}^{(n)}, \\ & \sum\limits_{n=1}^{N_j} b_{i,j}^{(n)} \gamma_j^{(n)} \leq K_{i,j}^{\delta}. \end{array} \tag{2.8}$$

Then we apply Lemma 2.2.2 and obtain the solution:

$$b_{i,j}^{(n)} = \begin{cases} 0 & \text{if } n \leq k \\ \min\left(B_{i,j}^{(n)}, \left\lfloor \frac{1}{\gamma_j^{(n)}} (K_{i,j}^{\delta} - \sum_{k<w<n} \gamma_j^{(w)} B_{i,j}^{(w)}) \right\rfloor_+ \right) & \text{otherwise.} \end{cases} \tag{2.9}$$

According to the theorem's notations, we have $M_{i,j}^{<}(t) = \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{y'=i} \mathbb{1}_{m_Q(\mathbf{x}',j)<t} m_Q(\mathbf{x}', j)$, which can also be written as $\sum_{w=1}^{k} \gamma_j^{(w)} B_{i,j}^{(w)}$ with $k = \max\{w|\gamma_j^{(w)} < t\}$. Then, we can represent $\sum_{k<w<n} \gamma_j^{(w)} B_{i,j}^{(w)}$ as $M_{i,j}^{<}(\gamma_j^{(n)}) - M_{i,j}^{<}(\theta_j)$. Let's define $p = \max\{n|K_{i,j}^{\delta} - M_{i,j}^{<}(\gamma_j^{(n)}) + M_{i,j}^{<}(\theta_j) > 0\}$. From Lemma 2.2.2 we derive that:

$$b_{i,j}^{(n)} = \begin{cases} 0 & n \leq k \\ B_{i,j}^{(n)} & k+1 \leq n < p \\ \frac{1}{\gamma_j^{(p)}} (K_{i,j}^{\delta} - M_{i,j}^{<}(\gamma_j^{(p)}) + M_{i,j}^{<}(\theta_j)) & n = p \\ 0 & n > p. \end{cases} \tag{2.10}$$

Using this we infer:

$$R_{\mathcal{U} \wedge \boldsymbol{\theta}}(B_Q, i, j) \leq \sum_{n=k+1}^{p-1} B_{i,j}^{(n)} + \frac{1}{\gamma_j^{(p)}} (K_{i,j}^{\delta} - M_{i,j}^{<}(\gamma_j^{(p)}) + M_{i,j}^{<}(\theta_j)).$$

Now, let's consider $\gamma_j^{(w)},\ w \in \{1, \ldots, N_j\}$.

- $w > p$:

$$\begin{aligned}\sum_{n=k+1}^{p-1} B_{i,j}^{(n)} + \frac{1}{\gamma_j^{(p)}}(K_{i,j}^{\delta} - M_{i,j}^{<}(\gamma_j^{(p)}) + M_{i,j}^{<}(\theta_j)) \\ \leq \sum_{n=k+1}^{p} B_{i,j}^{(n)} \\ \leq \sum_{n=k+1}^{w-1} B_{i,j}^{(n)} + \left\lfloor \frac{1}{\gamma_j^{(w)}}(K_{i,j}^{\delta} - M_{i,j}^{<}(\gamma_j^{(w)}) + M_{i,j}^{<}(\theta_j)) \right\rfloor_+ .\end{aligned}$$

- $w < p$: Consider the following difference:

$$\begin{aligned}&\sum_{n=k+1}^{p-1} B_{i,j}^{(n)} + \frac{1}{\gamma_j^{(p)}}(K_{i,j}^{\delta} - M_{i,j}^{<}(\gamma_j^{(p)}) + M_{i,j}^{<}(\theta_j)) \\ &\quad - \sum_{n=k+1}^{w-1} B_{i,j}^{(n)} - \frac{1}{\gamma_j^{(w)}}(K_{i,j}^{\delta} - M_{i,j}^{<}(\gamma_j^{(w)}) + M_{i,j}^{<}(\theta_j)) \\ &= \sum_{n=w}^{p-1} B_{i,j}^{(n)} + \frac{1}{\gamma_j^{(p)}}(K_{i,j}^{\delta} - M_{i,j}^{<}(\gamma_j^{(p)}) + M_{i,j}^{<}(\theta_j)) \\ &\quad - \frac{1}{\gamma_j^{(w)}}(K_{i,j}^{\delta} - M_{i,j}^{<}(\gamma_j^{(w)}) + M_{i,j}^{<}(\theta_j)) \\ &= \sum_{n=w}^{p} b_{i,j}^{(n)} - \frac{1}{\gamma_j^{(w)}}(K_{i,j}^{\delta} - M_{i,j}^{<}(\gamma_j^{(w)}) + M_{i,j}^{<}(\theta_j)) \\ &= \sum_{n=w}^{p} b_{i,j}^{(n)} - \frac{1}{\gamma_j^{(w)}}(\sum_{n=k+1}^{p} b_{i,j}^{(n)}\gamma_j^{(n)} - \sum_{n=k+1}^{w-1} \gamma_j^{(n)} B_{i,j}^{(n)}) \\ &= \sum_{n=w}^{p} b_{i,j}^{(n)} - \frac{1}{\gamma_j^{(w)}}(\sum_{n=k+1}^{p} b_{i,j}^{(n)}\gamma_j^{(n)} - \sum_{n=k+1}^{w-1} \gamma_j^{(n)} b_{i,j}^{(n)}) \\ &= \frac{1}{\gamma_j^{(w)}}(\sum_{n=w}^{p} b_{i,j}^{(n)}\gamma_j^{(w)} - \sum_{n=k+1}^{p} b_{i,j}^{(n)}\gamma_j^{(n)} + \sum_{n=k+1}^{w-1} \gamma_j^{(n)} b_{i,j}^{(n)}) \\ &= \frac{1}{\gamma_j^{(w)}}(\sum_{n=w}^{p} b_{i,j}^{(n)}\gamma_j^{(w)} - \sum_{n=w}^{p} b_{i,j}^{(n)}\gamma_j^{(n)}) \leq 0.\end{aligned}$$

Summing up, we derive Inequality (2.7). □

Corollary 2.2.4. *Let $U_{i,j}^{\delta}(\boldsymbol{\theta})$ be the upper bound for the transductive joint Bayes conditional risk from Theorem 2.2.3 that holds with probability at least $1-\delta$:*

$$U_{i,j}^{\delta}(\boldsymbol{\theta}) := \inf_{\gamma \in [\theta_j, 1]} \left\{ I_{i,j}^{(\leq,<)}(\theta_j, \gamma) + \frac{1}{\gamma} \left\lfloor (K_{i,j}^{\delta} - M_{i,j}^{<}(\gamma) + M_{i,j}^{<}(\theta_j)) \right\rfloor_+ \right\}.$$

Introduce the confusion matrix $\mathbf{U}_{\boldsymbol{\theta}}^{\delta}$ which (i,j)-entry is 0, if $i = j$, and $U_{i,j}^{\delta}(\boldsymbol{\theta})$ otherwise. We consider the spectral norm for confusion matrices as defined in Section 1.6. Then, we have:

$$\|\mathbf{C}_{B_Q}^{\mathcal{U}\wedge\boldsymbol{\theta}}\| \leq \|\mathbf{U}_{\boldsymbol{\theta}}^{\delta}\|,$$

$$\|\mathbf{C}_{B_Q}^{\mathcal{U}}\| \leq \|\mathbf{U}_{\mathbf{0}_K}^{\delta}\|,$$

where $\mathbf{0}_K$ is the K-size vector of zeros.

Proof. Remember a property of the spectral norm:

$$\mathbf{0}_{K,K} \leq \mathbf{A} \leq \mathbf{B} \Rightarrow \|\mathbf{A}\| \leq \|\mathbf{B}\|,$$

where comparison of matrices is element-wise, $\mathbf{0}_{K,K}$ is the K by K zero matrix and matrices $\mathbf{A}, \mathbf{B}$ are of the same size as $\mathbf{0}_{K,K}$.

The confusion matrix is always non-negative. In addition, one can see that $\mathbf{C}_{B_Q}^{\mathcal{U}\wedge\boldsymbol{\theta}}$ is element-wise no greater than $\mathbf{U}_{\boldsymbol{\theta}}^{\delta}$. Hence, we deduce the first inequality. The second inequality holds when in the first inequality $\boldsymbol{\theta} = \mathbf{0}_K$. □

Corollary 2.2.5. *Let $\mathbf{U}_{\boldsymbol{\theta}}^{\delta}$ be the upper bound matrix as it is defined in Corollary 2.2.4. Then, we have:*

$$\mathtt{E}_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q) \leq \left\| \left(\mathbf{U}_{\boldsymbol{\theta}}^{\delta}\right)^{\intercal} \mathbf{p} \right\|_1,$$

$$\mathtt{E}_{\mathcal{U}}(B_Q) \leq \left\| \left(\mathbf{U}_{\mathbf{0}_K}^{\delta}\right)^{\intercal} \mathbf{p} \right\|_1,$$

where $\mathbf{p} = \{u_i/u\}_{i=1}^{K}$ and $\mathbf{0}_K$ is the K-size vector of zeros.

Proof. As in Corollary 2.2.4 we notice again that we have:

$$\mathbf{C}_{B_Q}^{\mathcal{U}\wedge\boldsymbol{\theta}} \leq \mathbf{U}_{\boldsymbol{\theta}}^{\delta}.$$

It still holds, if we transpose matrices and multiply both sides by the vector $\mathbf{p}$:

$$\left(\mathbf{C}_{B_Q}^{\mathcal{U}\wedge\boldsymbol{\theta}}\right)^{\intercal} \mathbf{p} \leq \left(\mathbf{U}_{\boldsymbol{\theta}}^{\delta}\right)^{\intercal} \mathbf{p}.$$

Elements of the left vector are non-negative. Hence the inequality holds for the 1-norm, and taking into account Proposition 2.1.10 we infer:

$$\mathtt{E}_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q) = \left\| \left(\mathbf{C}_{B_Q}^{\mathcal{U}\wedge\boldsymbol{\theta}}\right)^{\intercal} \mathbf{p} \right\|_1 \leq \left\| \left(\mathbf{U}_{\boldsymbol{\theta}}^{\delta}\right)^{\intercal} \mathbf{p} \right\|_1.$$

We deduce the second inequality by substituting $\boldsymbol{\theta} = \mathbf{0}_K$ in the first inequality. □

Theorem 2.2.3 states that the bound of the conditional Bayes risk can be found as the solution of the linear program. We formulate a proposition that indicates conditions under which the bound is tight. Before this proposition, we prove the following lemma.

Lemma 2.2.6. *Consider non-negative real numbers $a, b, c \in \mathbb{R}^+$. Let $b \geq a$. Then, it is true that:*

$$\lfloor b - m \rfloor_+ - \lfloor a - m \rfloor_+ \leq b - a.$$

Proof. We can distinguish three cases:

1. $b > m$ and $a > m$. Then, $\lfloor b - m \rfloor_+ - \lfloor a - m \rfloor_+ = b - m - a + m = b - a$.
2. $b > m$ and $a \leq m$. Then, $\lfloor b - m \rfloor_+ - \lfloor a - m \rfloor_+ = b - m \leq b - a$.
3. $b \leq m$ and $a \leq m$. Then, $\lfloor b - m \rfloor_+ - \lfloor a - m \rfloor_+ = 0 \leq b - a$.

□

Proposition 2.2.7. *For all $\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}$ there exists $C \in [0, 1]$ such that for all $(i, j) \in \mathcal{Y}^2$, for all $\gamma > 0$:*

$$\frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}')=j} \mathbb{1}_{y'=i} \mathbb{1}_{m_Q(\mathbf{x}',j)=\gamma} \neq 0 \Rightarrow$$

$$\frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}')=j} \mathbb{1}_{y'=i} \mathbb{1}_{m_Q(\mathbf{x}',j)<\gamma} \geq C \cdot \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{y'=i} \mathbb{1}_{m_Q(\mathbf{x}',j)<\gamma}. \quad (2.11)$$

Then, with probability at least $1-\delta$ *the following inequality holds:*

$$F^{\delta}_{i,j} - R_{\mathcal{U}}(B_Q, i, j) \leq \frac{1-C}{C} R_{\mathcal{U}}(B_Q, i, j) + \frac{R^{\delta}_u(G_Q, i, j) - R_{\mathcal{U}}(G_Q, i, j)}{\gamma^*},$$

where

- $\gamma^* := \sup\{\gamma \in \Gamma_j | \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}')=j} \mathbb{1}_{y'=i} \mathbb{1}_{m_Q(\mathbf{x}',j)=\gamma} \neq 0\}$.
- $F^{\delta}_{i,j} := \inf_{\gamma \in [0,1]} \left\{ I^{(\leq,<)}_{i,j}(0,\gamma) + \frac{1}{\gamma} \left\lfloor (K^{\delta}_{i,j} - M^{<}_{i,j}(\gamma)) \right\rfloor_+ \right\}$.

Proof. First, let's show that

$$R_{\mathcal{U}}(B_Q, i, j) \geq \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}')=j} \mathbb{1}_{y'=i} \mathbb{1}_{m_Q(\mathbf{x}',j)<\gamma^*} + \frac{1}{\gamma^*} \lfloor K_{i,j} - M^{<}_{i,j}(\gamma^*) \rfloor_+, \tag{2.12}$$

where

- $K_{i,j} = R_{\mathcal{U}}(G_Q, i, j) - \varepsilon_{i,j}$,
- $\varepsilon_{i,j} = \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}') \neq j} \mathbb{1}_{y'=i} m_Q(\mathbf{x}', j)$.

Denote $\gamma^* = \gamma^{(p)}_j$. We apply Lemma 2.2.1 and get that $R_{\mathcal{U}}(G_Q, i, j) = \sum_{n=1}^{p} b^{(n)}_{i,j} \gamma^{(n)}_j + \varepsilon_{i,j}$, where $b^{(n)}_{i,j} := \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}')=j} \mathbb{1}_{y'=i} \mathbb{1}_{m_Q(\mathbf{x}',j)=\gamma^{(n)}_j}$. We can express $b^{(p)}_{i,j}$ in the following way:

$$b^{(p)}_{i,j} = \frac{K_{i,j} - \sum_{n=1}^{p-1} b^{(n)}_{i,j} \gamma^{(n)}_j}{\gamma^{(p)}_j}.$$

Since $-\sum_{n=1}^{p-1} b^{(n)}_{i,j} \gamma^{(n)}_j \geq -\sum_{n=1}^{p-1} B^{(n)}_{i,j} \gamma^{(n)}_j = -M^{<}_{i,j}(\gamma^{(p)}_j) = -M^{<}_{i,j}(\gamma^*)$ as well as $b^{(p)}_{i,j} \geq 0$, we deduce a lower bound for $b^{(p)}_{i,j}$:

$$b^{(p)}_{i,j} \geq \frac{1}{\gamma^*} \lfloor K_{i,j} - M^{<}_{i,j}(\gamma^*) \rfloor_+. \tag{2.13}$$

Also, taking into account Lemma 2.2.1, one can notice that:

$$R_{\mathcal{U}}(B_Q, i, j) = R_{\mathcal{U} \wedge \mathbf{0}}(B_Q, i, j) = \sum_{n=1}^{p} b^{(n)}_{i,j} = b^{(p)}_{i,j} + \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}')=j} \mathbb{1}_{y'=i} \mathbb{1}_{m_Q(\mathbf{x}',j)<\gamma^*}. \tag{2.14}$$

Combining Eq. (2.13) and Eq. (2.14) we deduce Eq. (2.12). Now, we come back to the proof of our proposition. Using the initial assumptions we deduce the following from Eq. (2.12):

$$\begin{aligned} R_{\mathcal{U}}(B_Q, i, j) &\geq C \cdot \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{y'=i} \mathbb{1}_{m_Q(\mathbf{x}',j)<\gamma^*} + \frac{1}{\gamma^*} \lfloor K_{i,j} - M^{<}_{i,j}(\gamma^*) \rfloor_+ \\ &= C \cdot I^{(\leq,<)}_{i,j}(0,\gamma^*) + \frac{1}{\gamma^*} \lfloor K_{i,j} - M^{<}_{i,j}(\gamma^*) \rfloor_+. \end{aligned} \tag{2.15}$$

By definition of $F^{\delta}_{i,j}$ we have:

$$F^{\delta}_{i,j} \leq I^{(\leq,<)}_{i,j}(0,\gamma^*) + \frac{1}{\gamma^*} \left\lfloor (K^{\delta}_{i,j} - M^{<}_{i,j}(\gamma^*)) \right\rfloor_+ \tag{2.16}$$

Subtracting Eq. (2.15) from Eq. (2.16) we obtain:

$$\begin{aligned} F^{\delta}_{i,j} - R_{\mathcal{U}}(B_Q, i, j) \leq (1-C) I^{(\leq,<)}_{i,j}(0,\gamma^*) + \\ \frac{1}{\gamma^*} \left(\left\lfloor (K^{\delta}_{i,j} - M^{<}_{i,j}(\gamma^*)) \right\rfloor_+ - \lfloor K_{i,j} - M^{<}_{i,j}(\gamma^*) \rfloor_+ \right), \end{aligned}$$

which holds with the probability $1-\delta$.

Then, one can notice that by definition, $R^{\delta}_{\mathcal{U}}(G_Q) \geq R_{\mathcal{U}}(G_Q)$ holds with probability $1-\delta$. From this, we obtain:

$$\begin{aligned} K^{\delta}_{i,j} - K_{i,j} &= R^{\delta}_{\mathcal{U}}(G_Q, i, j) - \varepsilon_{i,j} - R_{\mathcal{U}}(G_Q, i, j) + \varepsilon_{i,j} \\ &= R^{\delta}_{\mathcal{U}}(G_Q, i, j) - R_{\mathcal{U}}(G_Q, i, j) \geq 0. \end{aligned}$$

Using this fact as well as applying Lemma 2.2.6, we deduce that

$$\begin{aligned} \left\lfloor (K^{\delta}_{i,j} - M^{<}_{i,j}(\gamma^*)) \right\rfloor_+ - \lfloor K_{i,j} - M^{<}_{i,j}(\gamma^*) \rfloor_+ &\leq K^{\delta}_{i,j} - K_{i,j} \\ &= R^{\delta}_{\mathcal{U}}(G_Q, i, j) - R_{\mathcal{U}}(G_Q, i, j). \end{aligned} \tag{2.17}$$

Also, from Eq. (2.15) one can derive:

$$I^{(\leq,<)}_{i,j}(0, \gamma^*) \leq \frac{1}{C}(R_{\mathcal{U}}(B_Q, i, j) - \frac{1}{\gamma^*}\lfloor K_{i,j} - M^{<}_{i,j}(\gamma^*) \rfloor_+) \leq \frac{1}{C} R_{\mathcal{U}}(B_Q, i, j). \tag{2.18}$$

Taking into account Eq. (2.17) and Eq. (2.18), we infer:

$$F^{\delta}_{i,j} - R_{\mathcal{U}}(B_Q, i, j) \leq \frac{1-C}{C} R_{\mathcal{U}}(B_Q, i, j) + \frac{R^{\delta}_{\mathcal{U}}(G_Q, i, j) - R_{\mathcal{U}}(G_Q, i, j)}{\gamma^*}.$$

□

This proposition states that if Condition (2.11) holds, the difference between the conditional Bayes risk and its upper bound does not exceed an expression that depends on a constant C. If we assume that the Gibbs conditional risk bound is as tight as possible and the majority vote classifier makes most of its mistake for the class j on observations with the low value of $m_Q(\mathbf{x}', j)$, we obtain that Condition (2.11) accepts a high value C (close to 1), and the bound becomes tight. From theoretical point of view it makes sense to assume that the majority vote classifier mistakes mostly on low margin region, since we suppose that if the class got a relatively high vote from the hypotheses, then we expect that it is predicted correctly.

Chapter 3

Multi-class Self-Learning Algorithm and Its Applications

3.1 Multi-class Self-Learning Algorithm

In this chapter we give an application of the results that are given in Chapter 2. In Theorem 2.2.3 a transductive bound of the majority vote classifier is proposed. It bounds the joint Bayes conditional risk, which is the risk to predict a class j and have the margin more than some θ_j given the true label is equal to i. Moreover, Proposition 2.2.7 states that the bound is tight under certain condition, if we assume that the majority vote classifier makes a mistake predicting j mostly on observations with a low margin value for the class j. From Corollary 2.2.5 we derive the corresponding bound for the joint Bayes error rate. This result suggests us an opportunity to extend the margin-based self-learning algorithm described in Section 1.5.

However, the extension is not straightforward as the bound from Theorem 2.2.3 is implicit. It is implicit in a sense that it depends on the true labels of the observations. Moreover, the theorem assumes that a bound for the Gibbs conditional risk is given. In fact, there is no any work yet that is devoted to this problem. Nevertheless, we propose a solution that allows us to avoid both problems.

We remind that the upper bound given by Theorem 2.2.3 for the conditional joint Bayes risk has the following view:

$$R_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q, i, j) \leq U^{\delta}_{i,j}(\boldsymbol{\theta}) = \inf_{\gamma\in[\theta_j,1]} \left\{ I^{(\leq,<)}_{i,j}(\theta_j,\gamma) + \frac{1}{\gamma}\left|(K^{\delta}_{i,j} - M^{<}_{i,j}(\gamma) + M^{<}_{i,j}(\theta_j))\right|_+ \right\},$$

where

- $K^{\delta}_{i,j} = R^{\delta}_u(G_Q, i, j) - \frac{1}{u_i}\sum_{\mathbf{x}'\in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}')\neq j}\mathbb{1}_{y'=i}\, m_Q(\mathbf{x}', j)$,
- $I^{(\leq,<)}_{i,j}(\theta_j,\gamma) = \frac{1}{u_i}\sum_{\mathbf{x}'\in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{y'=i}\mathbb{1}_{\theta_j\leq m_Q(\mathbf{x}',j)<\gamma}$,
- $M^{<}_{i,j}(t) = \frac{1}{u_i}\sum_{\mathbf{x}'\in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{y'=i}\mathbb{1}_{m_Q(\mathbf{x}',j)<t}\, m_Q(\mathbf{x}', j)$.

From this, we can see that all terms of $U^{\delta}_{i,j}(\boldsymbol{\theta})$ depends on the true labels of the observations. Thus, the upper bound can not be computed in practice. To avoid this problem, we take into consideration the non-deterministic case, namely, we suppose the posterior distribution $P_Y(y|\mathbf{x})$ defined over $\mathcal{Y}$. Then, we can replace the deterministic $\mathbb{1}_{y=i}$ by the corresponding probabilistic $P_Y(i|\mathbf{x})$. With this approach, we do not need an upper bound for the conditional Gibbs risk anymore. Instead, we consider $K_{i,j} = \frac{1}{u_i}\sum_{\mathbf{x}'\in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}')=j}\mathbb{1}_{y'=i} m_Q(\mathbf{x}', j)$. Note that Theorem 2.2.3 works for this case too, since the solution of the linear program (Lemma 2.2.2) keeps to be the same if we change the $\leq$ constraint on the $=$ one. Thus, we consider replace $K_{i,j}$, $I^{(\leq,<)}_{i,j}(\theta_j,\gamma)$, $M^{<}_{i,j}(t)$ by the following:

- $\tilde{K}_{i,j} = \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{B_Q(\mathbf{x}')=j} P_Y(i|\mathbf{x}') m_Q(\mathbf{x}', j)$,
- $\tilde{I}_{i,j}^{(\leq,<)}(\theta_j, \gamma) = \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{\theta_j \leq m_Q(\mathbf{x}', j) < \gamma} P_Y(i|\mathbf{x}')$,
- $\tilde{M}_{i,j}^{<}(t) = \frac{1}{u_i} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{m_Q(\mathbf{x}', j) < t} P_Y(i|\mathbf{x}') m_Q(\mathbf{x}', j)$.

Finally, we make an assumption that the majority classifier is able to well describe the problem in our hypothesis space $\mathcal{H}$. In other words, we suppose that $m_Q(\mathbf{x}, y)$ can be considered as approximation of $P_Y(y|\mathbf{x})$. Although, this assumption is strong, one can notice that even without it we have strong dependency from the choice of the hypothesis space $\mathcal{H}$. Indeed, if the space describes the problem poorly, the majority vote classifier is not able to give "good margins", and then the pseudo-labelling approach can not provide a high increase in performance. Thus, in practice, we replace $P_Y(y|\mathbf{x})$ by $m_Q(\mathbf{x}, y)$ in computation of $\tilde{K}_{i,j}$, $\tilde{I}_{i,j}^{(\leq,<)}(\theta_j, \gamma)$, $\tilde{M}_{i,j}^{<}(t)$.

Now, we are going to describe the *multi-class self-learning algorithm*. Similarly to the self-learning algorithm (Section 1.5), the main principle is first to learn a supervised majority vote classifier over the labelled training examples and then iteratively assign pseudo-labels to unlabelled examples for which the margin for the corresponding predicted class is no less than the corresponding threshold. Then, a new classifier is learned using the augmented labelled set, and the process is repeated until all observations will get pseudo-labels, or no more pseudo-labelling would occur. At each step of the algorithm we find a threshold vector that minimises the conditional Bayes error rate.

Definition 3.1.1. *We define the conditional Bayes error rate* $\mathtt{E}_{\mathcal{U}|\boldsymbol{\theta}}(B_Q)$ *in the following way:*

$$\mathtt{E}_{\mathcal{U}|\boldsymbol{\theta}}(B_Q) := \frac{\mathtt{E}_{\mathcal{U} \wedge \boldsymbol{\theta}}(B_Q)}{\pi(m_Q(\mathbf{x}', k) \geq \theta_k)},$$

where $\pi(m_Q(\mathbf{x}', k) \geq \theta_k) := \frac{1}{u} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{m_Q(\mathbf{x}', k) \geq \theta_k}$ *and* $k := B_Q(\mathbf{x}')$.

In this definition, we evaluate the joint Bayes error rate using Corollary 2.2.5. It can be noticed that the conditional Bayes error rate resemble the conditional probability formula. Indeed, the numerator reflects "probability" to have a mistake on the unlabelled set and the threshold is equal to $\boldsymbol{\theta}$, whereas the denominator computes the "probability" to have a unlabelled observation with the margin no less than the threshold for the predicted class. Thus, the conditional Bayes error rate finds a trade-off between the value of the joint Bayes error rate and the number of pseudo-labelled examples.

The multi-class self-learning algorithm is described in Algorithm 2. Similarly to the self-learning algorithm, in practice, to find an optimal θ^* we perform grid search that is the exhaustive search over the grid of values within the interval $(0, 1]$. The same algorithm is used for computing the optimal γ^* that provides the value of an upper bound for the conditional risk (see Theorem 2.2.3). In contrast to the self-learning algorithm, the direct grid search in the multi-class setting is costly as the complexity becomes $O(S^K)$, where S is the sampling rate of the grid. The following remark helps us to avoid this problem.

Algorithm 2 Multi-class self-learning algorithm (MSLA)

Input:
Labelled dataset $Z_{\mathcal{L}}$
Unlabelled observations $\mathrm{X}_{\mathcal{U}}$
Initialisation:
A set of pseudo-labelled instances, $Z_{\mathcal{U}} \leftarrow \emptyset$
A classifier H trained on $Z_{\mathcal{L}}$
repeat

1. Compute the margin threshold $\boldsymbol{\theta}^*$ that minimises the conditional Bayes error rate:

$$\boldsymbol{\theta}^* = \operatorname*{argmin}_{\boldsymbol{\theta} \in (0,1]^K} \mathtt{E}_{\mathcal{U}|\boldsymbol{\theta}}(B_Q). \qquad (\star)$$

2. $S \leftarrow \{(\mathbf{x}', y') | \mathbf{x}' \in \mathrm{X}_{\mathcal{U}}; [m_Q(\mathbf{x}', y') \geq \theta_{y'}] \wedge [y' = \operatorname{argmax}_{c \in \mathcal{Y}} m_Q(\mathbf{x}', c)]\}$
3. $Z_{\mathcal{U}} \leftarrow Z_{\mathcal{U}} \cup S$, $\mathrm{X}_{\mathcal{U}} \leftarrow \mathrm{X}_{\mathcal{U}} \setminus S$
4. Learn a classifier H with the following loss function:

$$\mathcal{L}(H, Z_{\mathcal{L}}, Z_{\mathcal{U}}) = \frac{l + |Z_{\mathcal{U}}|}{l} \mathcal{L}(H, Z_{\mathcal{L}}) + \frac{l + |Z_{\mathcal{U}}|}{|Z_{\mathcal{U}}|} \mathcal{L}(H, Z_{\mathcal{U}})$$

until $\mathrm{X}_{\mathcal{U}}$ or S are $\emptyset$
Output: The final classifier H

Remark 3.1.2. *Let* $\mathtt{E}^{(j)}_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q) = \sum_{i=1}^{K} \frac{u_i}{u} R_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q, i, j)$. *Recall that* $R_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q, i, j)$ *depends only on one element of* $\boldsymbol{\theta}$, *which is* θ_j. *Then, we obtain:*

$$\begin{aligned}
\mathtt{E}_{\mathcal{U}|\boldsymbol{\theta}}(B_Q) &= \frac{\mathtt{E}_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q)}{\pi(m_Q(\mathbf{x}', k) \geq \theta_k)} \\
&= \frac{\sum_{j=1}^{K} \mathtt{E}^{(j)}_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q)}{\sum_{c=1}^{K} \frac{1}{u} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{m_Q(\mathbf{x}', c) \geq \theta_c} \mathbb{1}_{B_Q(\mathbf{x}') = c}} \\
&= \sum_{j=1}^{K} \frac{\mathtt{E}^{(j)}_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q)}{\sum_{c=1}^{K} \frac{1}{u} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{m_Q(\mathbf{x}', c) \geq \theta_c} \mathbb{1}_{B_Q(\mathbf{x}') = c}} \\
&\leq \sum_{j=1}^{K} \frac{\mathtt{E}^{(j)}_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q)}{\frac{1}{u} \sum_{\mathbf{x}' \in \mathrm{X}_{\mathcal{U}}} \mathbb{1}_{m_Q(\mathbf{x}', j) \geq \theta_j} \mathbb{1}_{B_Q(\mathbf{x}') = j}} \\
&= \sum_{j=1}^{K} \frac{\mathtt{E}^{(j)}_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q)}{\pi\{(m_Q(\mathbf{x}', j) \geq \theta_j) \wedge (B_Q(\mathbf{x}') = j)\}}. \qquad (*)
\end{aligned}$$

One can notice that $(*)$ is a sum, where j-th term is dependent on θ_j only and independent from other components of the vector $\boldsymbol{\theta}$. Therefore, we propose to minimise this sum instead of the conditional Bayes error rate, as the minimisation of the former leads to the minimisation of the latter. In addition, this approach allows us to reduce computational complexity, since all terms of $(*)$ can be minimised independently from each other. This leads to the fact that each component of $\boldsymbol{\theta}$ is tuned independently and the K-dimensional minimisation task is replaced by K tasks of 1-dimensional minimisation.

3.2 Numerical Experiments

In this section we describe numerical experiments to test on practice the multi-class self-learning algorithm (further denoted by `MSLA`) that is proposed in Section 3.1. We are interested in its

practical use for the tasks of the multi-class semi-supervised learning context. It means that we would like to see if it has good performance in a situation when $l \ll u$. We validate our approach by comparing the performance with other classification algorithms.

In our experiments, `MSLA` is based on the Random Forest approach [2], which is used as a majority vote classifier (H in Algorithm 2). By definition, a forest consists of trees that have equal weights in prediction. Hence, the posterior distribution Q over $\mathcal{H}$ is uniform. Then, the margin $\mathbf{m_x}$ of an observation is evaluated by the mean vector of votes that the trees of the forest give to each class. A tree gives a vote to the class that it predicts by computing the fraction of training examples in a leaf that also belong to this class.

In our experiments we do not tune the hyperparameters of the Random Forest. Indeed, in our work, we consider applications with a small number of labelled examples. Because of this, it does not make a lot of sense to tune hyperparameters on a relatively small data sample. In addition, for the Random Forest tuning does not play a crucial role as it could play for the `SVM` or the logistic regression. Thus, we use a random forest with 200 trees and the maximal depth of trees.

The first algorithm that is compared with `MSLA` is the Random Forest that is trained in the supervised mode. It means that the algorithm learns a model using only labelled examples and then applies it directly for prediction of a whole unlabelled set. We denote this approach `RF`. The second algorithm that we consider for comparison is the multi-class extension of the classical self-learning approach described in [23]. Compared to the `MSLA`, this approach takes one θ as an input parameter and use this for pseudo-labelling keeping threshold unchanged. We mark the algorithm as `FSLA`, and its description can be found in Appendix.

We consider `FSLA` under two settings, when $\theta = 0.7$ and $\theta = 0.9$. The second one can be regarded as a more conservative procedure. However, a high value of θ does not imply that the algorithm is going to work better as it was mentioned in Section 3.1. To reduce the computation time, we stop `FSLA`, if the algorithms makes more than 10 iterations. It may also improve the performance, since, in this case, the algorithm is less affected by noise.

We expect that in most cases `MSLA` outperforms methods described above. It may give a better result compared to the supervised mode, since it makes use of information derived from the unlabelled set. It pseudo-labels confident unlabelled examples, thereby enlarges the training set by additional information. In turn, `MSLA` may outperform `FSLA` as at each step it finds a threshold that minimises the conditional Bayes error rate, whilst its opponent keeps the same threshold regardless the error value that it could give.

We perform our experiments on 5 datasets that are available in public [8, 4]. We chose datasets coming from different applications. First, image classification is considered, for which we use the `MNIST` and the `Pendigits` databases of handwritten digits. Next, we consider an application in signal processing with the `SensIT` dataset for vehicle type classification. We cover speech recognition by the `Vowel` database. Finally, we consider an application in bioinformatics, namely, the `DNA` dataset. We use preprocessed versions [4] of all datasets, except `MNIST`. For the `MNIST` database we extract HOG-features [6] with the following parameters: the cell size is (4, 4), the block size is (5, 5), the number of orientations is 4. Table 3.1 summarises main characteristics of all datasets under consideration.

Dataset	# of labelled examples, l	# of unlabelled examples, u	Dimension, d	# of classes, K
DNA	31	3155	180	3
MNIST	210	41790	901	10
Pendigits	109	10883	16	10
SensIT	49	22831	100	3
Vowel	99	891	10	11

Table 3.1: Experiment setup.

Our experiments are conducted in the following way. For each dataset, we perform a random split on the train and the test sets 20 times. For each split, the performance is evaluated on the test observations. Results are averaged over all 20 trials. Note that the train and the test size are fixed for each dataset. We do not use the train/test splits that are proposed by data sources. Instead, we propose our own splits that makes a situation closer to the semi-supervised context. In Table 3.1 one can observe the considered splits. We evaluate performance over the test set using the classification accuracy and the F1-score.

The results of experiments, namely, means and standard deviations of the accuracy and the F1-score are reported in Table 3.2. In addition, we perform a statistical test to determine whether the best result is significantly better than performance of the other methods. For this, we apply the Mann–Whitney U test proposed in [16] with the level 0.01 of significance. Then, the symbol $\downarrow$ designates that the performance of the method is statistically worse than the best result at the level 0.01.

Dataset	Score	RF	MSLA	FSLA $_{\theta=0.7}$	FSLA $_{\theta=0.9}$
DNA	ACC	$.6986 \pm .0767$	$\mathbf{.7076} \pm .0817$	$.5168^{\downarrow} \pm .082$	$.6921 \pm .0752$
	F1	$.6558 \pm .1144$	$\mathbf{.6665} \pm .1174$	$.3747^{\downarrow} \pm .0852$	$.6467 \pm .1141$
MNIST	ACC	$.9039^{\downarrow} \pm .0120$	$\mathbf{.9448} \pm .0061$	$.8654^{\downarrow} \pm .0658$	$.7039^{\downarrow} \pm .0563$
	F1	$.9031^{\downarrow} \pm .0125$	$\mathbf{.9448} \pm .0063$	$.8450^{\downarrow} \pm .0882$	$.6852^{\downarrow} \pm .0647$
Pendigits	ACC	$.861^{\downarrow} \pm .0201$	$\mathbf{.886} \pm .0162$	$.835^{\downarrow} \pm .0384$	$.7998^{\downarrow} \pm .0287$
	F1	$.8586^{\downarrow} \pm .0229$	$\mathbf{.8845} \pm .0171$	$.8257^{\downarrow} \pm .0488$	$.7906^{\downarrow} \pm .0358$
SensIT	ACC	$.67 \pm .0291$	$\mathbf{.6745} \pm .0288$	$.6192^{\downarrow} \pm .0366$	$.53^{\downarrow} \pm .0391$
	F1	$.654 \pm .0448$	$\mathbf{.6599} \pm .0421$	$.5784^{\downarrow} \pm .0683$	$.4302^{\downarrow} \pm .0887$
Vowel	ACC	$\mathbf{.5851} \pm .0273$	$.5846 \pm .0268$	$.5265^{\downarrow} \pm .0374$	$.5839 \pm .0292$
	F1	$.5733 \pm .0293$	$\mathbf{.5754} \pm .0278$	$.5053^{\downarrow} \pm .0407$	$.5713 \pm .0311$

Table 3.2: The result table of the classification performance on different datasets described in Table 3.1. The performance is computed using two score functions: accuracy and F1. The sign $^{\downarrow}$ shows if the performance is statistically worse than the best result on the level 0.01 of significance.

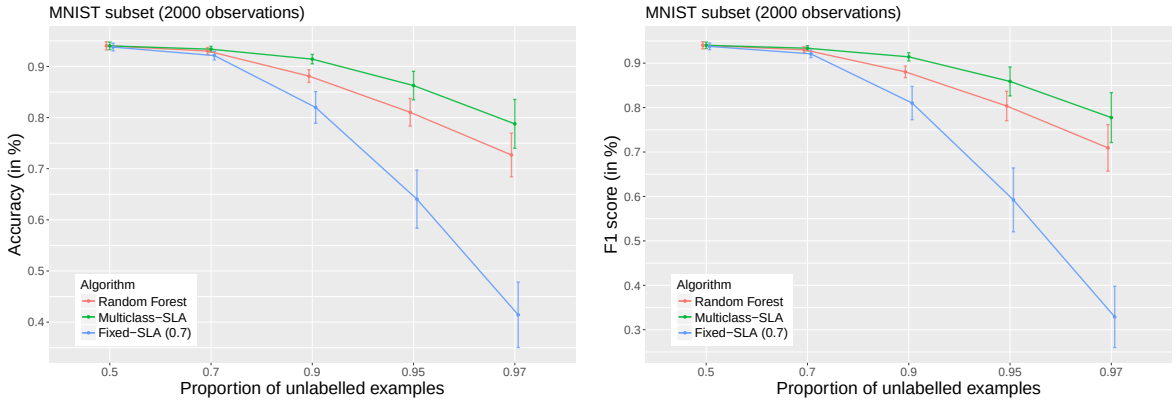
From Table 3.2 we derive the following conclusions:

- As expected, in most cases the MSLA algorithm performs better than other methods under consideration. For the MNIST and the Pendigits datasets the improvement is reported as significant. In the fist case, the MSLA outperforms the RF in average by 4.09% according to the accuracy score, and by 4.17% according to the F1 score. For the Pendigits, the gain is 2.5% and 2.59% according to the accuracy score and the F1 score respectively.

- From results one can deduce that the performance of `MSLA` depends on the performance of the corresponding supervised algorithm. Indeed, the worse predictive ability of the basis classifier is the worse the margin $m_Q(\mathbf{x}, y)$ approximates the conditional probability $P_Y(y|\mathbf{x})$. Thus, regardless the possible benefit `MSLA` could provide, there is always an unrecoverable error that the basis classifier produces on the initial step of the `MSLA`.

- In our experiments we have not found a case when the `FSLA` has any benefit. In most cases it is injected by noise, therefore, as a result, `FSLA` performs even worse than the corresponding supervised algorithm.

From Table 3.1 one can see that in our experiments we took a really small train size for all datasets. In fact, it does not exceed 1% from a whole set. Thereby, the question may arise how the performance of the `MSLA` changes when the number of labelled examples increases? For this, we compare the `MSLA`, the `RF` and the `FSLA` on the `MNIST` dataset splitting the dataset with different proportions of train/test observations. For each partition we perform 20 trials, as before. To reduce the computational time, we subsample the dataset on 2000 observations. The result graphs for the classification accuracy and the F1 score are illustrated in Figure 3.1. For sake of simplicity, we consider only `FSLA` with $\theta = 0.7$.



(a) Accuracy

(b) F1 score

Figure 3.1: Classification accuracy and F1 score with respect to the proportion of unlabelled examples for the `MNIST` dataset. On the graphs, dots represent the average performance on the test set over 20 random splits, whilst the vertical bars indicates the standard deviation. Although it is more accepted to draw the standard error, we depict the standard deviation for more clear illustration.

From the results we can see that the performance of all algorithms declines when the number of unlabelled examples increases. The difference between `MSLA` and `RF` becomes more significant when the test size grows, thereby we observe benefit from the pseudo-labelling. In the case of `FSLA`, we do not see any improvement in regard to `RF` and the performance falls drastically. When the number of labelled samples becomes large enough, all algorithms performs in the same way.

In addition, another set of experiments has been performed. One can notice that the `MSLA` can be applied for the binary classification tasks. However, it would not work in the same way as the self-learning algorithm described in Section 1.5. The reason why is the fact that Theorem 2.2.3 provides an upper bound for each conditional risk $R_{\mathcal{U}\wedge\boldsymbol{\theta}}(B_Q, i, j)$ separately, whereas Theorem 1.4.2 evaluates an upper bound for the general risk $R_{\mathcal{U}}(B_Q)$, which is the binary counterpart of the error rate $\mathtt{E}_{\mathcal{U}}(B_Q)$. From this, we can surmise that the `MSLA` works better for imbalanced binary classification tasks. To verify our guess we perform numerical experiments on two datasets for the binary framework, namely, the `Adult` dataset as well as the Breast Cancer Wisconsin (Diagnostic) database (`Wisconsin`) [8]. The description of the datasets can be found in Table 3.3.

Dataset	# of labelled examples, l	# of unlabelled examples, u	Dimension, d	Proportion of $-$ and $+$ examples in %
`Adult`	162	32399	14	75.9% vs. 24.1%
`Wisconsin`	13	686	9	65.5% vs. 34.5%

Table 3.3: Experiment setup.

Our experimental setup is the same. We perform 20 trials on a dataset computing the mean and the standard deviation of all algorithms. This time, we consider classifiers such as the `RF`, `MSLA`, `FSLA` with $\theta = 0.7$ and the self-learning algorithm from Section 1.5, which we denote as `BSLA`. As before, we check the significance of the best result by the Mann–Whitney U test with a significance level of 0.01. The results of classification can be found in Table 3.4.

Dataset	Score	`RF`	`MSLA`	`FSLA`	`BSLA`
`Adult`	`ACC`	$.8256 \pm .0081$	$.8236 \pm .0111$	$.8274 \pm .0099$	$\mathbf{.8285} \pm .0089$
	`F1`	$.8168 \pm .0074$	$\mathbf{.8206} \pm .0093$	$.8156 \pm .0108$	$.8178 \pm .0096$
`Wisconsin`	`ACC`	$.9289 \pm .0508$	$\mathbf{.9534} \pm .018$	$.9149^{\downarrow} \pm .0701$	$.9184^{\downarrow} \pm .0704$
	`F1`	$.925 \pm .0609$	$\mathbf{.953} \pm .0191$	$.9059^{\downarrow} \pm .0983$	$.9096^{\downarrow} \pm .0988$

Table 3.4: The result table of the classification performance on binary classification datasets described in Table 3.3. The performance is computed using two score functions: accuracy and F1. The sign $^{\downarrow}$ shows if the performance is statistically worse than the best result on the level 0.01 of significance.

We report the following conclusions from the results:

- Our guess is mostly confirmed, and the `MSLA` shows decent results compared to the other algorithms. On the `Wisconsin` dataset the gain in performance is 2.45% for the accuracy classification and 2.8% for the F1-score.
- On the `Adult` dataset all algorithms has approximately the same performance. The best accuracy score gives the `BSLA`, while `MSLA` is the best according to the F1-score. This is agreed with what we told before: the `BSLA` tends more to maximise the accuracy score, and the `MSLA` more attentively evaluate the joint error rate with respect to unbalanced distribution of classes, thereby having the best F1-score.
- We note that on the `Wisconsin` dataset the `MSLA` has the lowest standard deviation, while its opponent has the standard deviation 2 or even 3 times greater. From this we may infer that `MSLA` is more stable than the others when a small number of labelled examples is available.

Conclusion

This master thesis makes a twofold contribution. First, we propose novel results from the point of theory. In Chapter 2 transductive learning in the multi-class framework is studied. Theorem 2.2.3 proposes a transductive bound for the conditional risk of the majority vote classifier. This bound is based on the marginal distribution over unlabelled examples for a predicted class as well as the conditional Gibbs risk that is supposed to be given. This theorem follows the idea of [20] to consider the confusion matrix as an error measure instead of the error rate.

Theorem 2.2.3 is proved for a more general case, namely, we look for a transductive bound of the "joint probability" to make a conditional mistake and the corresponding margin is no less than a threshold. The proof is based on Lemma 2.2.1 that finds a connection between the majority vote and the Gibbs classifiers as well as Lemma 2.2.2 that finds an analytic solution of a linear program. From the theorem we derive two corollaries that give the corresponding bounds on the confusion matrix norm and the error rate. To the best our knowledge, transductive bounds for the Bayes error rate and the confusion matrix norm in the multi-class framework is a new result. In addition, Proposition 2.2.7 states that under certain conditions the bound from Theorem 2.2.3 becomes tight.

Second, we contribute by proposing a novel approach for multi-class classification of partially labelled data. This approach has named the multi-class self-learning algorithm and it can be considered as an extension of the self-learning algorithm proposed by [1]. The extension of this algorithm was motivated by the fact that there exists a few number of multi-class semi-supervised methods by now. In addition, the self-learning algorithm of [1] was verified in practice [10]. The idea of the adaptive thresholding allows this method to outperform the classical self-learning algorithm [23].

Our numerical results have demonstrated that the idea of pseudo-labelling with minimisation of the conditional Bayes error rate allow us to enhance the performance compared to the basis supervised approach. Moreover, we test our algorithm for the binary classification tasks with unbalanced class distribution. As a result, our approach shows more stable results than the self-learning algorithm of [1].

Appendix A

Multi-class Self-Learning Algorithm with a Fixed Threshold

Here, we provide a pseudo-code of the multi-class self-learning algorithm with a fixed threshold (which we call `FSLA` in Section 3.2). This approach can be considered as the multi-class extension of the classical self-learning algorithm [23]. Although the algorithm can be defined for a vector of thresholds, for the sake of simplicity, we consider only one threshold $\theta > 0$. The pseudo-code is described in Algorithm 3.

Algorithm 3 Multi-class self-learning algorithm with a fixed threshold (FSLA)

Input:
Labelled dataset $Z_{\mathcal{L}}$
Unlabelled observations $\mathrm{X}_{\mathcal{U}}$
The threshold $\theta > 0$
Initialisation:
A set of pseudo-labelled instances, $Z_{\mathcal{U}} \leftarrow \emptyset$
A classifier H trained on $Z_{\mathcal{L}}$
repeat

1. $S \leftarrow \{(\mathbf{x}', y') | \mathbf{x}' \in \mathrm{X}_{\mathcal{U}}; [m_Q(\mathbf{x}', y') \geq \theta] \wedge [y' = \mathrm{argmax}_{c \in \mathcal{Y}} \, m_Q(\mathbf{x}', c)]\}$

2. $Z_{\mathcal{U}} \leftarrow Z_{\mathcal{U}} \cup S$, $\mathrm{X}_{\mathcal{U}} \leftarrow \mathrm{X}_{\mathcal{U}} \setminus S$

3. Learn a classifier H with the following loss function:

$$\mathcal{L}(H, Z_{\mathcal{L}}, Z_{\mathcal{U}}) = \frac{l + |Z_{\mathcal{U}}|}{l} \mathcal{L}(H, Z_{\mathcal{L}}) + \frac{l + |Z_{\mathcal{U}}|}{|Z_{\mathcal{U}}|} \mathcal{L}(H, Z_{\mathcal{U}})$$

until $\mathrm{X}_{\mathcal{U}}$ or S are $\emptyset$
Output: The final classifier H

Bibliography

[1] Massih-Reza Amini, François Laviolette, and Nicolas Usunier. A transductive bound for the voted classifier with an application to semi-supervised learning. In *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 8-11, 2008*, pages 65–72, 2008.

[2] Leo Breiman. Random Forests. *Mach. Learn.*, 45(1):5–32, October 2001.

[3] Luc Bégin, Pascal Germain, François Laviolette, and Jean-Francis Roy. PAC-Bayesian Theory for Transductive Learning. In Samuel Kaski and Jukka Corander, editors, *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, volume 33 of *Proceedings of Machine Learning Research*, pages 105–113, Reykjavik, Iceland, 22–25 Apr 2014. PMLR.

[4] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A Library for Support Vector Machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27:1–27:27, May 2011.

[5] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. *Semi-Supervised Learning*. The MIT Press, 1st edition, 2010.

[6] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, pages 886–893, Washington, DC, USA, 2005. IEEE Computer Society.

[7] Philip Derbeko, Ran El-Yaniv, and Ron Meir. Explicit learning curves for transduction and application to clustering and compression algorithms. *J. Artif. Int. Res.*, 22(1):117–142, October 2004.

[8] Dua Dheeru and Efi Karra Taniskidou. UCI machine learning repository, 2017.

[9] Thomas G. Dietterich. Ensemble Methods in Machine Learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems*, MCS '00, pages 1–15, London, UK, UK, 2000. Springer-Verlag.

[10] Ali Fakeri-Tabrizi, Massih-Reza Amini, Cyril Goutte, and Nicolas Usunier. Multiview self-learning. *Neurocomput.*, 155(C):117–127, May 2015.

[11] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, August 1997.

[12] Sokol Koço and Cécile Capponi. On multi-class learning through the minimization of the confusion matrix norm. *CoRR*, abs/1303.4015, 2013.

[13] John Langford. Tutorial on practical prediction theory for classification. *J. Mach. Learn. Res.*, 6:273–306, December 2005.

[14] John Langford and John Shawe-Taylor. PAC-Bayes & margins. In *Proceedings of the 15th International Conference on Neural Information Processing Systems*, NIPS'02, pages 439–446, Cambridge, MA, USA, 2002. MIT Press.

[15] François Laviolette, Emilie Morvant, Liva Ralaivola, and Jean-Francis Roy. On Generalizing the C-Bound to the Multiclass and Multi-label Settings. *CoRR*, abs/1501.03001, 2015.

[16] H. B. Mann and D. R. Whitney. On a test of whether one of two random variables is stochastically larger than the other. *Ann. Math. Statist.*, 18(1):50–60, 03 1947.

[17] David McAllester. Simplified PAC-Bayesian margin bounds. In Bernhard Schölkopf and Manfred K. Warmuth, editors, *Learning Theory and Kernel Machines*, pages 203–215, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[18] David A. McAllester. PAC-bayesian model averaging. In *Proceedings of the Twelfth Annual Conference on Computational Learning Theory*, COLT '99, pages 164–170, New York, NY, USA, 1999. ACM.

[19] Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997.

[20] Emilie Morvant, Sokol Koço, and Liva Ralaivola. PAC-Bayesian Generalization Bound on Confusion Matrix for Multi-Class Classification. *CoRR*, abs/1202.6228, 2012.

[21] Vern Paulsen. *Completely Bounded Maps and Operator Algebras*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2003.

[22] Joel A. Tropp. User-Friendly Tail Bounds for Sums of Random Matrices. *Foundations of Computational Mathematics*, 12(4):389–434, Aug 2012.

[23] Gökhan Tür, Dilek Z. Hakkani-Tür, and Robert E. Schapire. Combining active and semi-supervised learning for spoken language understanding. *Speech Communication*, 45:171–186, 2005.

[24] L. G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, November 1984.

[25] Vladimir Vapnik. *Estimation of Dependences Based on Empirical Data: Springer Series in Statistics (Springer Series in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1982.

[26] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.