PAC-Bayesian Learning of Linear Classifiers

Pascal Germain PASCAL.GERMAIN.1@ULAVAL.CA
Alexandre Lacasse ALEXANDRE.LACASSE@IFT.ULAVAL.CA
François Laviolette FRANCOIS.LAVIOLETTE@IFT.ULAVAL.CA
Mario Marchand MARIO.MARCHAND@IFT.ULAVAL.CA
Département d'informatique et de génie logiciel, Université Laval, Québec, Canada, G1V-0A6

Abstract

We present a general PAC-Bayes theorem from which all known PAC-Bayes risk bounds are obtained as particular cases. We also propose different learning algorithms for finding linear classifiers that minimize these bounds. These learning algorithms are generally competitive with both AdaBoost and the SVM.

1. Intoduction

For the classification problem, we are given a training set of examples—each generated according to the same (but unknown) distribution D, and the goal is to find a classifier that minimizes the true risk (*i.e.*, the generalization error or the expected loss). Since the true risk is defined only with respect to the *unknown* distribution D, we are automatically confronted with the problem of specifying exactly what we should optimize on the training data to find a classifier having the smallest possible true risk. Many different specifications (of what should be optimized on the training data) have been provided by using different inductive principles but the final guarantee on the true risk, however, always comes with a so-called risk bound that holds uniformly over a set of classifiers. Hence, the formal justification of a learning strategy has always come *a posteriori* via a risk bound. Since a risk bound can be computed from what a classifier achieves on the training data, it automatically suggests the following optimization problem for learning algorithms: given a risk (upper) bound, find a classifier that minimizes it.

Despite the enormous impact they had on our understanding of learning, the VC bounds are generally very loose. These bounds are characterized by the fact that their data-dependencies only comes through the training error of the classifiers. The fact that there also exists VC lower bounds, that are asymptotically identical to the corresponding upper bounds, suggests that significantly tighter bounds can only come through extra data-dependent properties such as the distribution of margins achieved by a classifier on the training data.

Among the data-dependent bounds that have been proposed recently, the PAC-Bayes bounds (McAllester, 2003; Seeger, 2002; Langford, 2005; Catoni, 2007) seem to be especially tight. These bounds thus appear to be a good starting point for the design of a bound-minimizing algorithm. In this paper, we present a general PAC-Bayes theorem and show that all known PAC-Bayes bounds are corollaries of this general theorem. When spherical Gaussians, over the space of linear classifiers, are used for priors and posteriors, we show that the Gibbs classifier that minimizes any of the above-mentioned PAC-Bayes risk bound is obtained from the linear classifier that minimizes a non-convex objective function. We also propose two different learning algorithms for finding linear classifiers that minimize PAC-Bayes risk bounds and a third algorithm that uses cross-validation to determine the value of a parameter which is present in the risk bound of Catoni (2007). The first algorithm uses a non-informative prior to construct a classifier from all the training data. The second algorithm uses a fraction of the training set to construct an informative prior that is used to learn the final linear classifier on the remaining fraction of the training data[1]. The third algorithm is, as the first one, based on a non-informative prior but uses the cross-validation methodology to choose one of the bound's parameters.

Appearing in *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009.

[1]The idea of using a fraction of the training data to construct a prior has been proposed in (Ambroladze et al., 2006) for the problem of choosing the hyperparameter values of the SVM. In contrast, the priors are used here to directly minimize a PAC-Bayes bound.

Our extensive experiments indicate that the second and third algorithms are competitive with both AdaBoost and the SVM and are generally much more effective than the first algorithm in their ability at producing classifiers with small true risk.

2. Simplified PAC-Bayesian Theory

We consider binary classification problems where the input space $\mathcal{X}$ consists of an arbitrary subset of $\mathbb{R}^n$ and the output space $\mathcal{Y} = \{-1, +1\}$. An *example* is an input-output $(\mathbf{x}, y)$ pair where $\mathbf{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$. Throughout the paper, we adopt the PAC setting where each example $(\mathbf{x}, y)$ is drawn according to a fixed, but unknown, distribution D on $\mathcal{X} \times \mathcal{Y}$.

The *risk* $R(h)$ of any classifier $h\colon \mathcal{X} \to \mathcal{Y}$ is defined as the probability that h misclassifies an example drawn according to D. Given a training set S of m examples, the *empirical risk* $R_S(h)$ of any classifier h is defined by the frequency of training errors of h on S. Hence

$$R(h) \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{(\mathbf{x},y)\sim D} I(h(\mathbf{x}) \neq y)\,,$$
$$R_S(h) \stackrel{\text{def}}{=} \frac{1}{m}\sum_{i=1}^{m} I(h(\mathbf{x}_i) \neq y_i)\,,$$

where $I(a) = 1$ if predicate a is true and 0 otherwise.

After observing the training set S, the task of the learner is to choose a *posterior* distribution Q over a space $\mathcal{H}$ of classifiers such that the Q-weighted majority vote classifier B_Q will have the smallest possible risk. On any input example $\mathbf{x}$, the output $B_Q(\mathbf{x})$ of the majority vote classifier B_Q (sometimes called the Bayes classifier) is given by

$$B_Q(\mathbf{x}) \stackrel{\text{def}}{=} \text{sgn}\left[\mathop{\mathbf{E}}_{h\sim Q} h(\mathbf{x})\right],$$

where $\text{sgn}(s) = +1$ if $s > 0$ and $\text{sgn}(s) = -1$ otherwise. The output of the deterministic majority vote classifier B_Q is closely related to the output of a stochastic classifier called the *Gibbs* classifier G_Q. To classify an input example $\mathbf{x}$, the Gibbs classifier G_Q chooses randomly a (deterministic) classifier h according to Q to classify $\mathbf{x}$. The true risk $R(G_Q)$ and the empirical risk $R_S(G_Q)$ of the Gibbs classifier are thus given by

$$R(G_Q) = \mathop{\mathbf{E}}_{h\sim Q} R(h) \quad ; \quad R_S(G_Q) = \mathop{\mathbf{E}}_{h\sim Q} R_S(h)\,.$$

Any bound for $R(G_Q)$ can straightforwardly be turned into a bound for the risk of the majority vote $R(B_Q)$. Indeed, whenever B_Q misclassifies $\mathbf{x}$, at least half of the classifiers (under measure Q) misclassifies $\mathbf{x}$. It follows that the error rate of G_Q is at least half of the error rate of B_Q. Hence $R(B_Q) \leq 2R(G_Q)$. As shown in Langford and Shawe-Taylor (2003), this factor of 2 can sometimes be reduced to $(1 + \epsilon)$.

The following theorem gives both an upper and a lower bound on $R(G_Q)$ by upper-bounding $\mathcal{D}(R_S(G_Q), R(G_Q))$ for any convex function $\mathcal{D} : [0,1] \times [0,1] \to \mathbb{R}$.

Theorem 2.1. *For any distribution D, for any set $\mathcal{H}$ of classifiers, for any prior distribution P of support $\mathcal{H}$, for any $\delta \in (0,1]$, and for any convex function $\mathcal{D} : [0,1] \times [0,1] \to \mathbb{R}$, we have*

$$\Pr_{S\sim D^m}\left(\forall Q \text{ on } \mathcal{H}\colon \mathcal{D}(R_S(G_Q), R(G_Q)) \leq \frac{1}{m}\left[\text{KL}(Q\|P) + \ln\left(\frac{1}{\delta}\mathop{\mathbf{E}}_{S\sim D^m}\mathop{\mathbf{E}}_{h\sim P} e^{m\mathcal{D}(R_S(h),R(h))}\right)\right]\right) \geq 1 - \delta\,,$$

where $\text{KL}(Q\|P) \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{h\sim Q} \ln\frac{Q(h)}{P(h)}$ *is the Kullback-Leibler divergence between Q and P.*

Proof. Since $\mathop{\mathbf{E}}_{h\sim P} e^{m\mathcal{D}(R_S(h),R(h))}$ is a non-negative random variable, Markov's inequality gives

$$\Pr_{S\sim D^m}\left(\mathop{\mathbf{E}}_{h\sim P} e^{m\mathcal{D}(R_S(h),R(h))} \leq \frac{1}{\delta}\mathop{\mathbf{E}}_{S\sim D^m}\mathop{\mathbf{E}}_{h\sim P} e^{m\mathcal{D}(R_S(h),R(h))}\right) \geq 1 - \delta\,.$$

Hence, by taking the logarithm on each side of the innermost inequality and by transforming the expectation over P into an expectation over Q, we obtain

$$\Pr_{S\sim D^m}\left(\forall Q : \ln\left[\mathop{\mathbf{E}}_{h\sim Q}\frac{P(h)}{Q(h)} e^{m\mathcal{D}(R_S(h),R(h))}\right] \leq \ln\left[\frac{1}{\delta}\mathop{\mathbf{E}}_{S\sim D^m}\mathop{\mathbf{E}}_{h\sim P} e^{m\mathcal{D}(R_S(h),R(h))}\right]\right) \geq 1 - \delta\,.$$

The theorem then follows from two applications of Jensen's inegality: one exploiting the concavity of $\ln(x)$ and the second the convexity of $\mathcal{D}$. □

Theorem 2.1 provides a tool to derive PAC-Bayesian risk bounds. Each such bound is obtained by using a particular convex function $\mathcal{D} : [0,1] \times [0,1] \to \mathbb{R}$ and by upper-bounding $\mathop{\mathbf{E}}_{S\sim D^m}\mathop{\mathbf{E}}_{h\sim P} e^{m\mathcal{D}(R_S(h),R(h))}$.

For example, a slightly tighter PAC-Bayes bound than the one derived by Seeger (2002) and Langford (2005) can be obtained from Theorem 2.1 by using $\mathcal{D}(q,p) = \text{kl}(q,p)$, where

$$\text{kl}(q,p) \stackrel{\text{def}}{=} q\ln\frac{q}{p} + (1-q)\ln\frac{1-q}{1-p}\,.$$

Corollary 2.1. *For any distribution D, for any set $\mathcal{H}$ of classifiers, for any distribution P of support $\mathcal{H}$, for any $\delta \in (0,1]$, we have*

$$\Pr_{S\sim D^m}\left(\forall Q \text{ on } \mathcal{H}\colon \ \mathrm{kl}(R_S(G_Q), R(G_Q)) \leq \frac{1}{m}\left[\mathrm{KL}(Q\|P) + \ln\frac{\xi(m)}{\delta}\right]\right) \geq 1-\delta\,,$$

where $\xi(m) \stackrel{\text{def}}{=} \sum_{k=0}^{m}\binom{m}{k}(k/m)^k(1-k/m)^{m-k}$.

Proof. The corollary immediately follows from Theorem 2.1 by choosing $\mathcal{D}(q,p) = \mathrm{kl}(q,p)$. Indeed, in that case we have

$$\begin{aligned}
&\mathop{\mathbf{E}}_{S\sim D^m}\ \mathop{\mathbf{E}}_{h\sim P}\ e^{m\mathcal{D}(R_S(h),R(h))}\\
&= \mathop{\mathbf{E}}_{h\sim P}\ \mathop{\mathbf{E}}_{S\sim D^m}\ \left(\tfrac{R_S(h)}{R(h)}\right)^{mR_S(h)}\left(\tfrac{1-R_S(h)}{1-R(h)}\right)^{m(1-R_S(h))}\\
&= \mathop{\mathbf{E}}_{h\sim P}\ \sum_{k=0}^{m}\Pr_{S\sim D^m}\left(R_S(h)=\tfrac{k}{m}\right)\left(\tfrac{\frac{k}{m}}{R(h)}\right)^{k}\left(\tfrac{1-\frac{k}{m}}{1-R(h)}\right)^{m-k}\\
&= \sum_{k=0}^{m}\binom{m}{k}(k/m)^k(1-k/m)^{m-k}\,,
\end{aligned}$$

where last equality arises from the fact that $R_S(h)$ is a binomial random variable of mean $R(h)$. □

See Banerjee (2006) for a very similar proof. Note also that we retreive the exact formulation of the PAC-Bayes bound of Langford (2005) if we upper bound $\xi(m)$ by $m+1$. However, $\xi(m) \in \Theta(\sqrt{m})$.

The PAC-Bayes bound of McAllester (2003) can be obtained by using $\mathcal{D}(q,p) = 2(q-p)^2$.

Let us now consider functions that are *linear* in the empirical risk, *i.e.*, functions of the form $\mathcal{D}(q,p) = \mathcal{F}(p) - C\cdot q$ for convex $\mathcal{F}$. As the next corollary shows, this choice for $\mathcal{D}$ gives a PAC-Bayes bound whose minimum is obtained for Gibbs classifiers minimizing a simple linear combination of $R_S(G_Q)$ and $\mathrm{KL}(Q\|P)$. The next corollary has also been found by Catoni (2007)[Th.1.2.1].

Corollary 2.2. *For any distribution D, any set $\mathcal{H}$ of classifiers, any distribution P of support $\mathcal{H}$, any $\delta \in (0,1]$, and any positive real number C, we have*

$$\Pr_{S\sim D^m}\left(\begin{array}{l}\forall Q \text{ on } \mathcal{H}\colon\\ R(G_Q) \leq \frac{1}{1-e^{-C}}\left\{1-\exp\left[-\left(C\cdot R_S(G_Q) + \frac{1}{m}\left[\mathrm{KL}(Q\|P) + \ln\frac{1}{\delta}\right]\right)\right]\right\}\end{array}\right) \geq 1-\delta.$$

Proof. Put $\mathcal{D}(q,p) = \mathcal{F}(p) - C\cdot q$ for some function $\mathcal{F}$ to be defined. Then

$$\begin{aligned}
&\mathop{\mathbf{E}}_{S\sim D^m}\ \mathop{\mathbf{E}}_{h\sim P}\ e^{m\mathcal{D}(R_S(h),R(h))}\\
&= \mathop{\mathbf{E}}_{h\sim P}\ \mathop{\mathbf{E}}_{S\sim D^m}\ e^{m\mathcal{F}(R(h))-CmR_S(h)}\\
&= \mathop{\mathbf{E}}_{h\sim P}\ e^{m\mathcal{F}(R(h))}\sum_{k=0}^{m}\Pr_{S\sim D^m}\left(R_S(h)=\tfrac{k}{m}\right)e^{-Ck}\\
&= \mathop{\mathbf{E}}_{h\sim P}\ e^{m\mathcal{F}(R(h))}\sum_{k=0}^{m}\binom{m}{k}R(h)^k(1-R(h))^{m-k}e^{-Ck}\\
&= \mathop{\mathbf{E}}_{h\sim P}\ e^{m\mathcal{F}(R(h))}\left(R(h)e^{-C}+(1-R(h))\right)^m\,,
\end{aligned}$$

and the result follows easily from Theorem 2.1 when $\mathcal{F}$ is the convex function $\mathcal{F}(p) = \ln\frac{1}{(1-p\,[1-e^{-C}])}$. □

It is interesting to compare the bounds of Corollaries 2.1 and 2.2. A nice property of the bound of Corollary 2.2 is the fact that its minimization is obtained from the Gibbs classifier G_Q that minimizes $C\cdot mR_S(G_Q) + \mathrm{KL}(Q\|P)$. As we will see, this minimization problem is closely related to the one solved by the SVM when Q is an isotropic Gaussian over the space of linear classifiers. Minimizing the bound given by Corollary 2.1 does not appear to be as simple because the upper bound on $R(G_Q)$ is not an explicit function of $R_S(G_Q)$ and $\mathrm{KL}(Q\|P)$. However, this upper bound does not depend on an arbitrary constant such as C in Corollary 2.2—which gives a computational advantage to Corollary 2.1 since, several bound minimizations (one for each value of C) would be needed in the case of Corollary 2.2. The tightness of these bounds can be compared with the following proposition.

Proposition 2.1. *For any $0 \leq R_S \leq R < 1$, we have*

$$\max_{C\geq 0}\left\{-\ln\left(1-R[1-e^{-C}]\right) - CR_S\right\} \ = \ \mathrm{kl}(R_S, R)\,.$$

Consequently, by omitting $\ln(\xi(m))$, Corollary 2.1 always gives a bound which is tighter or equal to the one given by Corollary 2.2. On another hand, there always exists values of C for which Corollary 2.2 gives a tighter bound than Corollary 2.1.

The next lemma shows that the bound of Corollary 2.2 has the interesting property of having an analytical expression of the optimal posterior Q^* for every prior P.

Lemma 2.1. *For any set $\mathcal{H}$ of classifiers and any prior P of support $\mathcal{H}$, for any positive real number C, the posterior Q^* that minimizes the upper bound on $R(G_Q)$ of Corollary 2.2 has a density which is given by the following Boltzmann distribution :*

$$Q^*(h) \ = \ \frac{1}{Z}P(h)e^{-C\cdot mR_S(h)}\,,$$

where m denotes the number of training examples in S and Z is a normalizing constant.

Proof. We present here a proof for the case where $\mathcal{H}$ is countable. But the theorem also holds for the continuous case. For any fixed C, δ and P, the distribution Q^* minimizing the bound of Corollary 2.2 is the same as the one minimizing $\mathcal{B}(Q)$, where

$$\mathcal{B}(Q) \stackrel{\text{def}}{=} C \cdot \sum_{h \in \mathcal{H}} Q(h) R_S(h) + \frac{\text{KL}(Q\|P)}{m} ,$$

under the constraint $\sum_{h \in \mathcal{H}} Q(h) = 1$. At optimality, Q must satisfy Lagrange constraints, namely that there exists $\lambda \in \mathbb{R}$ such that for any $h \in \mathcal{H}$, we have

$$\lambda = \frac{\partial \mathcal{B}}{\partial Q(h)} = C \cdot R_S(h) + \frac{1}{m}\Big(1 + \log \frac{Q(h)}{P(h)}\Big) .$$

Consequently,

$$Q(h) = P(h) e^{m(\lambda - C \cdot R_S(h)) - 1} = \frac{1}{Z} P(h) e^{-C \cdot m R_S(h)} ,$$

where Z is a normalizing constant. □

It is well known that Bayes classifiers resulting from a Boltzmann distribution can only be expressed via integral formulations. Such intergrals can be approximated by some Markov Chain Monté Carlo sampling, but, since the mixing time is unknown, we have no real control on the precision of the approximation. For this reason, we restrict ourselves here to the case where the posterior Q is chosen from a parameterized set of distributions. Building on the previous work of Langford and Shawe-Taylor (2003) and Langford (2005), we will focus on isotropic Gaussian distributions of linear classifiers since, in this case, we have an exact analytical expression for B_Q, G_Q, $R_S(B_Q)$, $R_S(G_Q)$, and $\text{KL}(Q\|P)$ in terms of the parameters of the posterior Q. These analytic expressions will enable us to perform our computations without performing any Monté-Carlo sampling.

3. Specialization to Linear Classifiers

Let us apply Corollary 2.1 and 2.2 to linear classifiers that are defined over a space of features. Here we suppose that each $\mathbf{x} \in \mathcal{X}$ is mapped to a feature vector $\boldsymbol{\phi}(\mathbf{x}) = (\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots)$ where each ϕ_i is given explicitly as a real-valued function or given implicitly by using a Mercer kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. In the latter case, we have $k(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x}) \cdot \boldsymbol{\phi}(\mathbf{x}') \, \forall (\mathbf{x}, \mathbf{x}') \in \mathcal{X} \times \mathcal{X}$. Each linear classifier $h_{\mathbf{w}}$ is identified by a real-valued weight vector $\mathbf{w}$. The output $h_{\mathbf{w}}(\mathbf{x})$ of $h_{\mathbf{w}}$ on any $\mathbf{x} \in \mathcal{X}$ is given by

$$h_{\mathbf{w}}(\mathbf{x}) \;=\; \text{sgn}\left(\mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{x})\right) .$$

The task of the learner is to produce a posterior Q over the set of all possible weight vectors. If each possible feature vector $\boldsymbol{\phi}$ has N components, the set of all possible weight vectors is $\mathbb{R}^N$. Let $Q(\mathbf{v})$ denote the posterior density evaluated at weight vector $\mathbf{v}$. We restrict ourselves to the case where the learner is going to produce a posterior $Q_{\mathbf{w}}$, parameterized by a chosen weight vector $\mathbf{w}$, such that for any weight vectors $\mathbf{v}$ and $\mathbf{u}$ we have $Q_{\mathbf{w}}(\mathbf{v}) = Q_{\mathbf{w}}(\mathbf{u})$ whenever $\mathbf{v} - \mathbf{w} = -(\mathbf{u} - \mathbf{w})$. Posteriors $Q_{\mathbf{w}}$ satisfying this property are said to be *symmetric about* $\mathbf{w}$. It can be easily shown that for any $Q_{\mathbf{w}}$ symmetric about $\mathbf{w}$ and for any feature vector $\boldsymbol{\phi}$:

$$\text{sgn}\left(\mathop{\mathbf{E}}_{\mathbf{v} \sim Q_{\mathbf{w}}} \text{sgn}\left(\mathbf{v} \cdot \boldsymbol{\phi}\right)\right) = \; \text{sgn}\left(\mathbf{w} \cdot \boldsymbol{\phi}\right) . \qquad (1)$$

In other words, for any input example, the output of the majority vote classifier $B_{Q_{\mathbf{w}}}$ (given by the left hand side of Equation 1) is the same as the one given by the linear classifier $h_{\mathbf{w}}$ whenever $Q_{\mathbf{w}}$ is symmetric about $\mathbf{w}$. Consequently, $R(h_{\mathbf{w}}) = R(B_{Q_{\mathbf{w}}}) \leq 2R(G_{Q_{\mathbf{w}}})$ and, consequently, Corollary 2.1 and 2.2 provide upper bounds on $R(h_{\mathbf{w}})$ for these posteriors. Building on the previous work of Langford and Shawe-Taylor (2003) and Langford (2005), we choose both the prior $P_{\mathbf{w}_p}$ and the posterior $Q_{\mathbf{w}}$ to be spherical Gaussians with identity covariance matrix respectively centered on $\mathbf{w}_p$ and on $\mathbf{w}$. Hence, for any weight vector $\mathbf{v} \in \mathbb{R}^N$:

$$Q_{\mathbf{w}}(\mathbf{v}) \;\; = \;\; \left(\frac{1}{\sqrt{2\pi}}\right)^N \exp\left(-\frac{1}{2}\|\mathbf{v} - \mathbf{w}\|^2\right)$$

Thus, the posterior is parameterized by a weight vector $\mathbf{w}$ that will be chosen by the learner based on the values of $R_S(G_{Q_{\mathbf{w}}})$ and $\text{KL}(Q_{\mathbf{w}}\|P_{\mathbf{w}_p})$. Here, the weight vector $\mathbf{w}_p$ that parameterizes the prior $P_{\mathbf{w}_p}$ represents prior knowledge that the learner might have about the classification task (*i.e.*, about good direction for linear separators). We therefore have $\mathbf{w}_p = \mathbf{0}$ in the absence of prior knowledge so that $P_{\mathbf{0}}$ is the *non-informative prior*. Alternatively, we might set aside a subset S' of the training data S and choose $\mathbf{w}_p$ such that $R_{S'}(G_{P_{\mathbf{w}_p}})$ is small.

By performing simple Gaussian integrals, as in Langford (2005), we find

$$\begin{aligned}
\text{KL}(Q_{\mathbf{w}}\|P_{\mathbf{w}_p}) &= \frac{1}{2}\|\mathbf{w} - \mathbf{w}_p\|^2 \\
R(G_{Q_{\mathbf{w}}}) &= \mathop{\mathbf{E}}_{(\mathbf{x},y)\sim D} \Phi\Big(\|\mathbf{w}\| \, \Gamma_{\mathbf{w}}(\mathbf{x}, y)\Big) \\
R_S(G_{Q_{\mathbf{w}}}) &= \frac{1}{m}\sum_{i=1}^{m} \Phi\Big(\|\mathbf{w}\| \, \Gamma_{\mathbf{w}}(\mathbf{x}_i, y_i)\Big) ,
\end{aligned}$$

where $\Gamma_{\mathbf{w}}(\mathbf{x}, y)$ denotes the normalized margin of $\mathbf{w}$ on $(\mathbf{x}, y)$, *i.e.*, $\Gamma_{\mathbf{w}}(\mathbf{x}, y) \stackrel{\text{def}}{=} \frac{y\mathbf{w}\cdot\boldsymbol{\phi}(\mathbf{x})}{\|\mathbf{w}\| \, \|\boldsymbol{\phi}(\mathbf{x})\|}$, and where $\Phi(a)$

denotes the probability that $X > a$ when X is a $N(0,1)$ random variable, *i.e.*,

$$\Phi(a) \stackrel{\text{def}}{=} \frac{1}{\sqrt{2\pi}} \int_a^\infty \exp\left(-\frac{1}{2}x^2\right) dx\,. \qquad (2)$$

3.1. Two objective functions to minimize

By using the above expressions for $R_S(G_{Q_\mathbf{w}})$ and $\mathrm{KL}(Q_\mathbf{w}\|P_{\mathbf{w}_p})$, Corollaries 2.1 and 2.2 both provide upper bounds to $R(G_{Q_\mathbf{w}})$ and to $R(h_\mathbf{w})$ (since $R(h_\mathbf{w}) \leq 2R(G_{Q_\mathbf{w}})$). Hence, each bound depend on the same quantities: the empirical risk measure, $R_S(G_{Q_\mathbf{w}})$, and $\mathrm{KL}(Q_\mathbf{w}\|P_{\mathbf{w}_p})$ which acts as a regularizer.

Minimizing the upper bound given by Corollary 2.1, in the case of linear classifiers, amounts to finding $\mathbf{w}^*$ that minimizes the following objective function

$$\mathcal{B}(S,\mathbf{w},\delta) \stackrel{\text{def}}{=} \sup\left\{\epsilon \,:\, \mathrm{kl}(R_S(G_{Q_\mathbf{w}})\|\epsilon) \;\leq\; \frac{1}{m}\left[\mathrm{KL}(Q_\mathbf{w}\|P_{\mathbf{w}_p}) + \ln\frac{\xi(m)}{\delta}\right]\right\}, \quad (F_{2.1})$$

for a fixed value of the confidence parameter δ (say $\delta = 0.05$). Consequently, our problem is to find weight vector $\mathbf{w}^*$ that minimizes $\mathcal{B}$ subject to the constraints

$$\mathrm{kl}\Big(R_S(G_{Q_\mathbf{w}})\Big\|\mathcal{B}\Big) = \frac{1}{m}\left[\mathrm{KL}(Q_\mathbf{w}\|P_{\mathbf{w}_p}) + \ln\frac{\xi(m)}{\delta}\right] \quad (3)$$

$$\mathcal{B} \;>\; R_S(G_{Q_\mathbf{w}})\,. \qquad (4)$$

Minimizing the bound of Corollary 2.2, in the case of linear classifiers, amounts at finding $\mathbf{w}^*$ that minimizes the simple objective function

$$CmR_S(G_{Q_\mathbf{w}}) + \mathrm{KL}(Q_\mathbf{w}\|P_{\mathbf{w}_p}) = C\sum_{i=1}^m \Phi\Big(\frac{y_i\mathbf{w}\cdot\boldsymbol{\phi}(\mathbf{x}_i)}{\|\boldsymbol{\phi}(\mathbf{x}_i)\|}\Big) + \frac{1}{2}\|\mathbf{w}-\mathbf{w}_p\|^2\,, \quad (F_{2.2})$$

for some fixed choice of C and $\mathbf{w}_p$. In the absence of prior knowledge, $\mathbf{w}_p = \mathbf{0}$ and the regularizer becomes identical to the one used by the SVM. Indeed, the learning strategy used by the soft-margin SVM consists at finding $\mathbf{w}$ that minimizes

$$C\sum_{i=1}^m \max\Big(0, 1 - y_i\mathbf{w}\cdot\boldsymbol{\phi}(\mathbf{x}_i)\Big) + \frac{1}{2}\|\mathbf{w}\|^2\,,$$

for some fixed choice of C. Thus, for $\mathbf{w}_p = \mathbf{0}$, both learning strategies are identical except for the fact that the convex SVM hinge loss, $\max(0,\cdot)$, is replaced by the non-convex *probit loss*, $\Phi(\cdot)$. Hence, the objective function minimized by the soft-margin SVM is a convex relaxation of objective function $F_{2.2}$. Each learning strategy has its potential drawback. The single local minimum of the soft-margin SVM solution might be suboptimal, whereas the non-convex PAC-Bayes bound might present several local minima.

Observe that $\mathcal{B}$, the objective function $F_{2.1}$, is defined only implicitly in terms of $\mathbf{w}$ via the constraints given by Equations (3) and (4). This optimization problem appears to be more involved than the (unconstrained) optimization of objective function $F_{2.2}$ that arises from Corollary 2.2. However, it appears also to be more relevant, since, according to Proposition 2.1, the upper bound given by Corollary 2.1 is somewhat tighter than the one given by Corollary 2.2 (apart from the presence of a $\ln(\xi(m))$ term). The optimization of the objective function $F_{2.1}$ has also the advantage of not being dependent of any constant C like the one present in the objective objective function $F_{2.2}$.

4. Gradient Descent of the PAC-Bayes Bound

We are now concerned with the problem of minimizing the (non-convex) objective functions $F_{2.1}$ (for fixed $\mathbf{w}_p$) and $F_{2.2}$ (for fixed C and $\mathbf{w}_p$). As a first approach, it makes sense to minimize these objective functions by gradient-descent. More specifically, we have used the Polak-Ribière conjugate gradient descent algorithm implemented in the GNU Scientific Library (GSL). The gradient (with respect to $\mathbf{w}$) of objective function $F_{2.1}$ is obtained by computing the partial derivative of both sides of Equation (3) with respect to w_j (the jth component of $\mathbf{w}$). After solving for $\partial\mathcal{B}/\partial w_j$, we find that the gradient is given by

$$\frac{1}{m}\frac{\mathcal{B}(1-\mathcal{B})}{\mathcal{B}-R_S}\left[\mathbf{w}-\mathbf{w}_p + \ln\left(\frac{\mathcal{B}(1-R_S)}{R_S(1-\mathcal{B})}\right)\cdot \sum_{i=1}^m \Phi'\left(\frac{y_i\mathbf{w}\cdot\boldsymbol{\phi}(\mathbf{x}_i)}{\|\boldsymbol{\phi}(\mathbf{x}_i)\|}\right)\frac{y_i\boldsymbol{\phi}(\mathbf{x}_i)}{\|\boldsymbol{\phi}(\mathbf{x}_i)\|}\right], \quad (5)$$

where $\Phi'(t)$ denotes the first derivative of Φ evaluated at t. We have observed that objective function $F_{2.1}$ tends to have only one local minimum, even if it is not convex. We have therefore used a single gradient descent run to minimize $F_{2.1}$.

The gradient of objective function $F_{2.2}$ is

$$C\sum_{i=1}^m \Phi'\left(\frac{y_i\mathbf{w}\cdot\boldsymbol{\phi}(\mathbf{x}_i)}{\|\boldsymbol{\phi}(\mathbf{x}_i)\|}\right)\frac{y_i\boldsymbol{\phi}(\mathbf{x}_i)}{\|\boldsymbol{\phi}(\mathbf{x}_i)\|} + (\mathbf{w}-\mathbf{w}_p)\,. \quad (6)$$

Since this objective function might have several local minima, especially for large values of C, each *objective function minimization* of $F_{2.2}$ consisted of k different gradient-descent runs, where each run was initiated

from a new, randomly-chosen, starting position. In the results presented here, we have used $k = 10$ for $C \leq 10$ and $k = 100$ for $C > 10$.

4.1. Proposed learning algorithms

We propose three algorithms that can be used either with the *primal* variables (*i.e.*, the components of $\mathbf{w}$) or the *dual* variables $\{\alpha_1, \ldots, \alpha_m\}$ that appear in the linear expansion $\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{\phi}(\mathbf{x}_i)$. In this latter case, the features are implicitly given by a Mercer kernel $k(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x}) \cdot \boldsymbol{\phi}(\mathbf{x}')\ \forall (\mathbf{x}, \mathbf{x}') \in \mathcal{X} \times \mathcal{X}$. The objective functions $F_{2.1}$ and $F_{2.2}$, with their gradients (Eq. (5) and (6)), can then straightforwardly be expressed in terms of $k(\cdot, \cdot)$ and the dual variables.[2]

The first algorithm, called **PBGD1**, uses the prior $P_{\mathbf{0}}$ (*i.e.*, with $\mathbf{w}_p = \mathbf{0}$) to learn a posterior $Q_{\mathbf{w}}$ by minimizing the bound value of Corollary 2.1 (objective function $F_{2.1}$). In this paper, every bound computation has been performed with $\delta = 0.05$.

The second algorithm, called **PBGD2**, was studied to investigate if it is worthwhile to use a fraction x of the training data to construct an informative prior $P_{\mathbf{w}_p}$, for some $\mathbf{w}_p \neq \mathbf{0}$, that will be used to learn a posterior $Q_{\mathbf{w}}$ on the remaining $1-x$ fraction of the training data. In its first stage, PBGD2 minimizes the objective function $F_{2.2}$ by using a fraction x of the training data to construct one posterior for each value of $C \in \{10^k : k = 0, \ldots, 6\}$. Note that a large value for C attempts to generate a $\mathbf{w}$ for which the training error of $G_{\mathbf{w}}$ is small. Each posterior is constructed with the same non-informative prior used for PBGD1 (*i.e.*, with $\mathbf{w}_p = \mathbf{0}$). Then, each of these seven posteriors is used as a prior $P_{\mathbf{w}_p}$, with $\mathbf{w}_p \neq \mathbf{0}$, for learning a posterior $Q_{\mathbf{w}}$ by minimizing the objective function $F_{2.1}$ on the remaining fraction $1 - x$ of the training data. From the union bound argument, the δ term in Corollary 2.1 needs to be replaced by $(\delta/7)$ to get a bound that uniformly holds for these seven priors. Empirically, we have observed that the best fraction x used for constructing the prior was $1/2$. Hence, we report here only the results for $x = 1/2$.

For the third algorithm, called **PBGD3**, we always used the prior $P_{\mathbf{0}}$ to minimize the objective function $F_{2.2}$. But, instead of using the solution obtained for the value of C that gave the smallest bound of Corollary 2.2, we performed 10-fold cross validation on the training set to find the "best" value for C and then used that value of C to find the classifier that minimizes objective function $F_{2.2}$. Hence PBGD3 follows the same cross-validation learning methodology normally employed with the SVM but uses the probit loss instead of the hinge loss.

To compute the risk bound for the linear classifier returned by PBGD3 and the other comparison algorithms (AdaBoost and SVM), we performed a line search, along the direction of the weight vector $\mathbf{w}$ of the returned classifier, to find the norm $\|\mathbf{w}\|$ that minimizes the bound of Corollary 2.1.[3] For each bound computation, we used the non-informative prior $P_{\mathbf{0}}$.

4.2. PBGD with Respect to Primal Variables

For the sake of comparison, all learning algorithms of this subsection are producing a linear classifier $h_{\mathbf{w}}$ on the set of basis functions $\{\phi_1, \phi_2, \ldots\}$ known as *decision stumps*. Each decision stump ϕ_i is a threshold classifier that depends on a single attribute: its output is $+b$ if the tested attribute exceeds a threshold value t, and $-b$ otherwise, where $b \in \{-1, +1\}$. For each attribute, at most ten equally-spaced possible values for t were determined *a priori*.

We have compared the three PBGD algorithms to AdaBoost (Schapire et al., 1998) because the latter is a standard and efficient algorithm when used with decision stumps. Since AdaBoost is an algorithm that minimizes the exponential risk $\frac{1}{m} \sum_{i=1}^{m} \exp(-y_i \mathbf{w} \boldsymbol{\phi}(\mathbf{x}_i))$, it never chooses a $\mathbf{w}$ for which there exists a training example where $-y_i \mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{x}_i)$ is very large. This is to be contrasted with the PBGD's algorithms for which the empirical risk $R_S(G_{Q_{\mathbf{w}}})$ has the "sigmoidal" shape of Equation (2) (and never exceeds one). We thus anticipate that AdaBoost and PBGD's algorithms will select different weight vectors $\mathbf{w}$ on many data sets.

The results obtained for all three algorithms are summarized in Table 1. Except for MNIST, all data sets were taken from the UCI repository. Each data set was randomly split into a training set S of $|S|$ examples and a testing set T of $|T|$ examples. The number n of attributes for each data set is also specified. For AdaBoost, the number of boosting rounds was fixed to 200. For all algorithms, $R_T(\mathbf{w})$ refers to the frequency of errors, measured on the testing set T, of the linear classifier $h_{\mathbf{w}}$ returned by the learner. For the PBGD's algorithms, $G_T(\mathbf{w}) \stackrel{\text{def}}{=} R_T(G_{Q_{\mathbf{w}}})$ refers to the empirical risk on T of the Gibbs classifier. The "*Bnd*" columns refer to the PAC-Bayes bound of Corollary 2.1, computed on the training set. All bounds hold with confidence $1 - \delta = 0.95$. For PBGD1, PBGD3 and AdaBoost, the bound is computed on all the training data

[2] This is true if $\mathbf{w}_p$ can be expanded in terms of examples that do not belong to the training set.

[3] This is justified by the fact that the bound holds uniformly for all weight vectors $\mathbf{w}$.

Table 1. Summary of results for linear classifiers on decision stumps.

Dataset				(a) **AdaBoost**		(1) **PBGD1**			(2) **PBGD2**			(3) **PBGD3**			**SSB**
Name	$\|S\|$	$\|T\|$	n	$R_T(\mathbf{w})$	Bnd	$R_T(\mathbf{w})$	$G_T(\mathbf{w})$	Bnd	$R_T(\mathbf{w})$	$G_T(\mathbf{w})$	Bnd	$R_T(\mathbf{w})$	$G_T(\mathbf{w})$	Bnd	
Usvotes	235	200	16	**0.055**	0.346	0.085	0.103	0.207	0.060	0.058	0.165	0.060	0.057	0.261	
Credit-A	353	300	15	0.170	0.504	0.177	0.243	0.375	0.187	0.191	0.272	**0.143**	0.159	0.420	
Glass	107	107	9	0.178	0.636	0.196	0.346	0.562	0.168	0.176	0.395	**0.150**	0.226	0.581	
Haberman	144	150	3	**0.260**	0.590	0.273	0.283	0.422	0.267	0.287	0.465	0.273	0.386	0.424	
Heart	150	147	13	0.259	0.569	**0.170**	0.250	0.461	0.190	0.205	0.379	0.184	0.214	0.473	
Sonar	104	104	60	0.231	0.644	0.269	0.376	0.579	0.173	0.168	0.547	**0.125**	0.209	0.622	
BreastCancer	343	340	9	0.053	0.295	**0.041**	0.058	0.129	0.047	0.054	0.104	0.044	0.048	0.190	
Tic-tac-toe	479	479	9	0.357	0.483	0.294	0.384	0.462	**0.207**	0.208	0.302	**0.207**	0.217	0.474	(2,3)<(a,1)
Ionosphere	176	175	34	0.120	0.602	0.120	0.223	0.425	0.109	0.129	0.347	**0.103**	0.125	0.557	
Wdbc	285	284	30	0.049	0.447	0.042	0.099	0.272	0.049	0.048	0.147	**0.035**	0.051	0.319	
MNIST:0vs8	500	1916	784	0.008	0.528	0.015	0.052	0.191	0.011	0.016	0.062	**0.006**	0.011	0.262	
MNIST:1vs7	500	1922	784	**0.013**	0.541	0.020	0.055	0.184	0.015	0.016	0.050	0.016	0.017	0.233	
MNIST:1vs8	500	1936	784	0.025	0.552	0.037	0.097	0.247	0.027	0.030	0.087	**0.018**	0.037	0.305	(3)<(1)
MNIST:2vs3	500	1905	784	0.047	0.558	0.046	0.118	0.264	0.040	0.044	0.105	**0.034**	0.048	0.356	
Letter:AvsB	500	1055	16	0.010	0.254	0.009	0.050	0.180	**0.007**	0.011	0.065	**0.007**	0.044	0.180	
Letter:DvsO	500	1058	16	0.036	0.378	0.043	0.124	0.314	0.033	0.039	0.090	**0.024**	0.038	0.360	
Letter:OvsQ	500	1036	16	**0.038**	0.431	0.061	0.170	0.357	0.053	0.053	0.106	0.042	0.049	0.454	
Adult	1809	10000	14	**0.149**	0.394	0.168	0.196	0.270	0.169	0.169	0.209	0.159	0.160	0.364	(a)<(1,2)
Mushroom	4062	4062	22	**0.000**	0.200	0.046	0.065	0.130	0.016	0.017	0.030	0.002	0.004	0.150	(a,3)<(2)<(1)

with the non informative prior $P_{\mathbf{0}}$. For PBGD2, the bound is computed on the second half of the training data with the prior $P_{\mathbf{w}_p}$ constructed from the first half, and, as explain in Section 4.1, with δ replaced by $(\delta/7)$. Note that the bounds values for the classifiers returned by PBGD2 are generally much lower those for the classifiers produced by the other algorithms. This almost always materializes in a smaller testing error for the linear classifier produced by PBGD2. To our knowledge, these training set bounds for PBGD2 are the smallest ones obtained for any learning algorithm producing linear classifiers.

To determine whether or not a difference of empirical risk measured on the testing set T is statistically significant, we have used the test set bound method of Langford (2005) (based on the binomial tail inversion) with a confidence level of 95%. It turns out that no algorithm has succeeded in choosing a linear classifier $h_{\mathbf{w}}$ which was statistically significantly better (SSB) than the one chosen by another algorithm except for the few cases that are list in the column "SSB" of Table 1.

Overall, AdaBoost and PBGD2and3 are very competitive to one another (with no clear winner) and are generally superior to PBGD1. We therefore see an advantage in using half of the training data to learn a good prior over using a non-informative prior and keeping all the data to learn the posterior.

4.3. PBGD with Respect to Dual Variables

In this subsection, we compare the PBGD algorithms to the soft-margin SVM. Here, all four learning algorithms are producing a linear classifier on a feature space defined by the RBF kernel k satisfying $k(\mathbf{x},\mathbf{x}') = \exp(-\frac{1}{2}\|\mathbf{x}-\mathbf{x}'\|^2/\gamma^2)\ \forall(\mathbf{x},\mathbf{x}') \in \mathcal{X}\times\mathcal{X}$. The results obtained for all algorithms are summarized in Table 2. All the data sets are the same as those of the previous subsection. The notation used in this table is identical to the one used for Table 1.

For the SVM and PBGD3, the kernel parameter γ and the soft-margin parameter C was chosen by 10-fold cross validation (on the training set S) among the set of values proposed by Ambroladze et al. (2006). PBGD1and2 also tried the same set of values for γ but used the bound of Corollary 2.1 to select a good value. Since we consider 15 different values of γ, the bound of PBGD1 is therefore computed with δ replaced by $(\delta/15)$. For PBGD2, the bound is, as stated before, computed on the second half of the training data but with δ replaced by $(\delta/(7\cdot15))$. Again, the bounds values for the classifiers returned by PBGD2 are generally much lower those for the classifiers produced by the other algorithms. To our knowledge, the training set bounds for PBGD2 are the smallest ones obtained for any learning algorithm producing linear classifiers.

The same method as in the previous subsection was used to determine whether or not a difference of empirical risk measured on the testing set T is statistically significant. It turns out that no algorithm has chosen a linear classifier $h_{\mathbf{w}}$ which was statistically significantly better than the choices of the others *except* the few cases listed in the "SSB" column. Thus, the SVM and PBGD3 are very competitive to one another (with no clear winner) and are both slightly superior to PBGD2, and a bit more than slightly superior than PBGD1.

5. Conclusion

We have shown that the standard PAC-Bayes risk bounds (McAllester, 2003; Seeger, 2002; Langford, 2005; Catoni, 2007) are specializations of Theorem 2.1 that are obtained by choosing a particular convex func-

Table 2. Summary of results for linear classifiers with a RBF kernel.

Dataset				(s) SVM		(1) PBGD1			(2) PBGD2			(3) PBGD3			SSB
Name	$\|S\|$	$\|T\|$	n	$R_T(\mathbf{w})$	Bnd	$R_T(\mathbf{w})$	$G_T(\mathbf{w})$	Bnd	$R_T(\mathbf{w})$	$G_T(\mathbf{w})$	Bnd	$R_T(\mathbf{w})$	$G_T(\mathbf{w})$	Bnd	
Usvotes	235	200	16	0.055	0.370	0.080	0.117	0.244	**0.050**	0.050	0.153	0.075	0.085	0.332	
Credit-A	353	300	15	0.183	0.591	**0.150**	0.196	0.341	**0.150**	0.152	0.248	0.160	0.267	0.375	
Glass	107	107	9	0.178	0.571	**0.168**	0.349	0.539	0.215	0.232	0.430	**0.168**	0.316	0.541	
Haberman	144	150	3	0.280	0.423	0.280	0.285	0.417	0.327	0.323	0.444	**0.253**	0.250	0.555	
Heart	150	147	13	0.197	0.513	0.190	0.236	0.441	**0.184**	0.190	0.400	0.197	0.246	0.520	
Sonar	104	104	60	0.163	0.599	0.250	0.379	0.560	0.173	0.231	0.477	**0.144**	0.243	0.585	
BreastCancer	343	340	9	**0.038**	0.146	0.044	0.056	0.132	0.041	0.046	0.101	0.047	0.051	0.162	
Tic-tac-toe	479	479	9	0.081	0.555	0.365	0.369	0.426	0.173	0.193	0.287	**0.077**	0.107	0.548	(s,3)<(2)<(1)
Ionosphere	176	175	34	0.097	0.531	0.114	0.242	0.395	0.103	0.151	0.376	**0.091**	0.165	0.465	
Wdbc	285	284	30	0.074	0.400	0.074	0.204	0.366	**0.067**	0.119	0.298	0.074	0.210	0.367	
MNIST:0vs8	500	1916	784	**0.003**	0.257	0.009	0.053	0.202	0.007	0.015	0.058	0.004	0.011	0.320	
MNIST:1vs7	500	1922	784	0.011	0.216	0.014	0.045	0.161	**0.009**	0.015	0.052	0.010	0.012	0.250	
MNIST:1vs8	500	1936	784	0.011	0.306	0.014	0.066	0.204	0.011	0.019	0.060	**0.010**	0.024	0.291	
MNIST:2vs3	500	1905	784	**0.020**	0.348	0.038	0.112	0.265	0.028	0.043	0.096	0.023	0.036	0.326	(s)<(1)
Letter:AvsB	500	1055	16	**0.001**	0.491	0.005	0.043	0.170	0.003	0.009	0.064	**0.001**	0.408	0.485	
Letter:DvsO	500	1058	16	0.014	0.395	0.017	0.095	0.267	0.024	0.030	0.086	**0.013**	0.031	0.350	
Letter:OvsQ	500	1036	16	0.015	0.332	0.029	0.130	0.299	0.019	0.032	0.078	**0.014**	0.045	0.329	
Adult	1809	10000	14	**0.159**	0.535	0.173	0.198	0.274	0.180	0.181	0.224	0.164	0.174	0.372	(s,3)<(2)
Mushroom	4062	4062	22	**0.000**	0.213	0.007	0.032	0.119	0.001	0.003	0.011	**0.000**	0.001	0.167	(s,2,3)<(1)

tion $\mathcal{D}$ that binds Gibbs' true risk to its empirical estimate. Moreover, when spherical Gaussians over spaces of linear classifiers are used for priors and posteriors, we have shown that the Gibbs classifier $G_{Q_{\mathbf{w}}}$ that minimizes the PAC-Bayes bound of Corollary 2.1 (resp. 2.2) is obtained from the weight vector $\mathbf{w}$ that minimizes the (non-convex) objective function $F_{2.1}$ (resp. $F_{2.2}$). When the prior is non-informative, a simple convex relaxation of $F_{2.2}$ gives the objective function which is minimized by the soft-margin SVM.

We have proposed two learning algorithms (PBGD1 and PBGD2) for finding linear classifiers that minimize the bound of Corollary 2.1, and another algorithm (PBGD3) that uses the cross-validation methodology to determine the value of parameter C in the objective function $F_{2.2}$. PBGD1 uses a non-informative prior to construct the final classifier from all the training data. In contrast, PBGD2 uses a fraction of the training set to construct an informative prior that is used to learn the final linear classifier on the remaining fraction of the training data. Our extensive experiments indicate that PBGD2 is generally much more effective than PBGD1 at producing classifiers with small true risk. Moreover, the training set risk bounds obtained for PBGD2 are, to our knowledge, the smallest obtained so far for any learning algorithm producing linear classifiers. In fact, PBGD2 is a learning algorithm producing classifiers having a good guarantee without the need of using any test set for that purpose. This opens the way to a feasible learning strategy that uses all the available data for training. Our results also indicate that PBGD2 and PBGD3 are competitive with both AdaBoost and the soft-margin SVM at producing classifiers with small true risk. However, as a consequence of the non-convexity of the objective function $F_{2.2}$, PBGD2 and PBGD3 are slower than AdaBoost and the SVM.

Acknowledgements Work supported by NSERC Discovery grants 262067 and 0122405.

References

Ambroladze, A., Parrado-Hernández, E., & Shawe-Taylor, J. (2006). Tighter PAC-Bayes bounds. *Proceedings of the 2006 conference on Neural Information Processing Systems (NIPS-06)* (pp. 9–16).

Banerjee, A. (2006). On bayesian bounds. *ICML '06: Proceedings of the 23rd international conference on Machine learning* (pp. 81–88).

Catoni, O. (2007). *PAC-Bayesian surpevised classification: the thermodynamics of statistical learning.* Monograph series of the Institute of Mathematical Statistics, http://arxiv.org/abs/0712.0248.

Langford, J. (2005). Tutorial on practical prediction theory for classification. *Journal of Machine Learning Research*, *6*, 273–306.

Langford, J., & Shawe-Taylor, J. (2003). PAC-Bayes & margins. In S. T. S. Becker and K. Obermayer (Eds.), *Advances in neural information processing systems 15*, 423–430. Cambridge, MA: MIT Press.

McAllester, D. (2003). PAC-Bayesian stochastic model selection. *Machine Learning*, *51*, 5–21.

Schapire, R. E., Freund, Y., Bartlett, P., & Lee, W. S. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, *26*, 1651–1686.

Seeger, M. (2002). PAC-Bayesian generalization bounds for gaussian processes. *Journal of Machine Learning Research*, *3*, 233–269.