Bayes Point Machines: Estimating the Bayes Point in Kernel Space

Ralf Herbrich[*,†], **Thore Graepel**[*]
[*]Computer Science Department
Technical University of Berlin
10587 Berlin, Germany

Colin Campbell[†]
[†]Department of Engineering Mathematics
Bristol University
Bristol BS8 1TR, United Kingdom

Abstract

From a Bayesian perspective Support Vector Machines choose the hypothesis corresponding to the largest possible hypersphere that can be inscribed in *version space*, i.e. in the space of all consistent hypotheses given a training set. Those boundaries of version space which are tangent to the hypersphere define the support vectors. An alternative and potentially better approach is to construct the hypothesis using the whole of version space. This is achieved by using a *Bayes Point Machine* which finds the midpoint of the region of intersection of all hyperplanes bisecting version space into two halves of equal volume (the Bayes point). It is known that the center of mass of version space approximates the Bayes point [Watkin, 1993]. We suggest estimating the center of mass by averaging over the trajectory of a billiard ball bouncing in version space. Experimental results are presented indicating that Bayes Point Machines consistently outperform Support Vector Machines.

1 Introduction

Recently, the study of classification learning has shown that the generalization error of classifiers based on *real–valued* functions can be controlled by making use of a quantity known as the margin. Let us consider the set $\mathcal{H}_k$ of *kernel classifiers* [Vapnik, 1998; Whaba, 1990][1]

$$f(\mathbf{x}) = \sum_{i=1}^{\ell} \alpha_i k(\mathbf{x}_i, \mathbf{x}) \qquad \boldsymbol{\alpha} \in \mathbb{R}^{\ell} \,. \tag{1}$$

Here, k is referred to as a kernel and is assumed to be symmetric and positive definite. It is known from the theory of reproducing kernel Hilbert spaces (RKHS) [Whaba, 1990] that there exists a fixed *feature space* $\mathcal{F}$ not necessarily unique and a mapping $\phi : \mathcal{X} \mapsto \mathcal{F}$ such that f can be expressed as an inner product between the mapped point $\mathbf{x}$ and a vector $\mathbf{w} \in \mathcal{F}$, i.e.

$$\begin{aligned} f(\mathbf{x}) &= \langle \mathbf{w}, \phi(\mathbf{x}) \rangle_{\mathcal{F}} \qquad \mathbf{w} \in \mathcal{F} \,, \\ \mathbf{w} &= \sum_{i=1}^{\ell} \alpha_i \phi(\mathbf{x}_i) \,. \end{aligned}$$

Without loss of generality we assume in the following that $\mathcal{F}$ is the surface of a hypersphere $\|\phi(\mathbf{x})\|_{\mathcal{F}} = 1$. Suppose we are given a training set $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{\ell} \subset (\mathcal{X} \times \{-1, +1\})^{\ell}$. In a similar fashion to PAC analysis we assume that there is a function $f^* \in \mathcal{H}_k$ such that $y_i = f^*(\mathbf{x}_i)$. Then the space of consistent hypotheses — in the following referred to as the version space — is defined by[2]

$$\mathcal{V}(S) = \left\{ \mathbf{w} | y_i f(\mathbf{x}_i) > 0; i = 1, \ldots, \ell; \|\mathbf{w}\|_{\mathcal{F}}^2 = 1 \right\} . \tag{2}$$

Learning algorithms making use of the margin $\gamma = \min_S(y_i f(\mathbf{x}_i))$ bound the volume of version space from below by a ball of radius γ. If the radius of this ball is large relative to the total size of parameter space, the *effective* complexity of the functions contained in the ball is small (c.f. [Shawe-Taylor and Williamson, 1997]). Nevertheless, one can imagine circumstances where $\mathcal{V}(S)$ covers a large volume, which is poorly approximated by the volume of the largest inscribable ball (see Figure 2). Hence, large–margin classifiers are condemned to fail. We will present an algorithm which overcomes this difficulty. The algorithm estimates the center of mass by sampling from the *whole* of version space. Note that large–margin classifiers implicitly approximate the center of mass by the center of the inscribed hypersphere. Since it is very difficult to efficiently sample from $\mathcal{V}(S)$ we follow the idea of [Rujàn, 1997] and average over the trajectory of a billiard ball bounced in $\mathcal{V}(S)$.

The paper is structured as follows: in the subsequent section we revisit methods of learning linear classifiers. In Section 3 we introduce an example of a Bayes Point

[1] As a slight abuse of notation we refer to both f and $\text{sign}(f)$ as classifiers.

[2] Since multiplying each α_i by an arbitrary constant would not change f, uniqueness is ensured by a length constraint on $\mathbf{w}$.

Machine (BPM) which uses a billiard algorithm to approximate the center of mass of $\mathcal{V}(S)$ in kernel space. Then, in Section 4 we present experimental results that support the usefulness of our approach. Finally we conclude the paper and discuss the proposed technique.

2 Approaches to Learning Classifiers

Given version space the main question is: which linear classifier in $\mathcal{V}(S)$ is optimal and consequently should be returned by a learning algorithm? From the point of view of empirical risk minimization all the linear classifiers in $\mathcal{V}(S)$ are equivalent. Basically, two different approaches have been devised.

PAC Style Analysis Bounding the complexity of a subset of classifiers from above, the VC/PAC–theory of learning recommends to return the classifier f_{PAC} originating from a subset of small complexity. Hence, the term complexity refers to the VC–dimension, fat–shattering dimension, or the margin attained on the training set (for a detailed discussion and definition of these concepts see [Shawe-Taylor *et al.*, 1996; Vapnik, 1998]). The following theorem can serve as a basis for the well known class of large–margin algorithms.

Theorem 1 ([Shawe-Taylor *et al.*, 1996])**.** *Suppose inputs are drawn independently according to a distribution whose support is contained in the ball of radius R. If we succeed in correctly classifying ℓ such inputs by a hyperplane $\mathbf{w} \in \mathcal{V}(S)$ achieving a margin of $\gamma = \min_S(y_i f(\mathbf{x}_i))$, then with confidence $1-\delta$ the generalization error will be bounded from above by*

$$\frac{2}{\ell}\left(\kappa \log_2\left(\frac{8e\ell}{\kappa}\right)\log_2(32\ell) + \log_2\frac{8\ell}{\delta}\right),$$

where $\kappa = \lfloor 577R^2/\gamma^2 \rfloor$.

Maximizing the margin γ minimizes κ and thus controls the generalization ability of the learning algorithm. The corresponding learning problem is therefore given by

$$\begin{array}{ll} \mathbf{max}_{\mathbf{w}} & \min_S\left(y_i\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle_{\mathcal{F}}\right) \equiv \Delta \\ \mathbf{s.t.} & \begin{array}{rcl} y_i\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle_{\mathcal{F}} & \geq & \Delta > 0 \quad i = 1, \ldots, \ell \\ \|\mathbf{w}\|^2_{\mathcal{F}} & = & 1\,. \end{array} \end{array}$$

Let us relax the unit norm constraint on $\mathbf{w}$ but instead fix $\min_S(y_i\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle_{\mathcal{F}}) = 1 = \frac{\Delta}{\|\mathbf{w}\|_{\mathcal{F}}}$. Then, the solution $\mathbf{w}_1$ to the former problem is up to a scaling equivalent to the solution $\mathbf{w}_2$ of the following problem

$$\begin{array}{ll} \mathbf{min}_{\mathbf{w}} & \|\mathbf{w}\|^2_{\mathcal{F}} \\ \mathbf{s.t.} & y_i\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle_{\mathcal{F}} \geq 1 \quad i = 1, \ldots, \ell\,. \end{array} \tag{3}$$

This optimization task is a QP problem and its solution corresponds to the solution found by the Support Vector Machine (SVM). Note, that $y_i\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle_{\mathcal{F}}$ can also be read as the distance of $\mathbf{w}$ from the hyperplane with normal $y_i\phi(\mathbf{x}_i)$ if $\|\phi(\mathbf{x}_i)\| = 1$. Therefore SVMs can be viewed as finding the center of the largest hypersphere inscribable in version space (see Figure 1 and 2).

Bayesian analysis Assuming an *a–priori* distribution over the space of classifiers *and* the data, return that function f_{MAP} having the maximal posterior probability (MAP) or — using the posterior — the *average* linear classifier f_{Bayes} under the posterior distribution[3]. In a similar fashion to PAC analysis let us make a Bayesian model and consider f_{MAP} as well as the average classifier f_{Bayes}. Hence, we make the following assumptions:

$$\begin{array}{rcl} P(y|\mathbf{x}, \mathbf{w}) & = & \delta(y - \text{sign}(\langle \mathbf{w}, \phi(\mathbf{x})\rangle_{\mathcal{F}}))\,, \\ P(\mathbf{w}) & = & \dfrac{\delta(1 - \|\mathbf{w}\|^2_{\mathcal{F}})}{\int \delta(1 - \|\mathbf{v}\|^2_{\mathcal{F}})\, d\mathbf{v}} = \text{constant}\,. \end{array}$$

where δ refers to the delta function. The first distributional assumption can be viewed as a noise–free learning scenario. The second distributional assumption assumes that each linear classifier is equally likely. Then given a training set S, we have the following estimate for the posterior distribution

$$\begin{array}{rcl} P(\mathbf{w}|S) & \stackrel{\text{iid}}{=} & \dfrac{1}{Z}\displaystyle\prod_{i=1}^{\ell} P(y_i|\mathbf{x}_i, \mathbf{w})P(\mathbf{w}) \\ & = & \begin{cases} \frac{1}{Z} & \text{if } \mathbf{w} \in \mathcal{V}(S) \\ 0 & \text{otherwise} \end{cases}, \\ Z & = & P(S) = \displaystyle\int_{\mathcal{V}(S)} P(S|\mathbf{v}) dP(\mathbf{v})\,. \end{array}$$

It can be seen from these expressions that the f_{MAP} is not unique and thus classical perceptron learning is well justified from a MAP perspective using PAC–like priors. With a slight abuse of notation let us derive the Bayes decision f_{Bayes} at point $\mathbf{x}$ using the posterior

$$\begin{array}{rcl} f_{\text{Bayes}}(\mathbf{x}) & = & \displaystyle\int \langle \mathbf{w}, \phi(\mathbf{x})\rangle_{\mathcal{F}}\, dP(\mathbf{w}|S) \\ & = & \langle \mathbf{w}_{\text{Bayes}}, \phi(\mathbf{x})\rangle_{\mathcal{F}} \\ \mathbf{w}_{\text{Bayes}} & = & \displaystyle\int_{\mathcal{V}(S)} \mathbf{w}\, dP(\mathbf{w}|S)\,. \end{array}$$

It turns out that the center of mass in $\mathcal{V}(S)$ coincides with the so called Bayes point $\mathbf{w}_{\text{Bayes}}$. Here we made use of the fact that our decision functions f are real valued and linear, which gives for *any* prior $P(\mathbf{w})$ that $f_{\text{Bayes}} \in \mathcal{H}_k$. Another motivation for $\mathbf{w}_{\text{Bayes}}$ is given by the following: if we consider a new test point $\mathbf{x}$, we see that the Bayes–optimal decision functions are given by $\{\mathbf{w}|P(\text{sign}(\langle \mathbf{w}, \phi(\mathbf{x})\rangle_{\mathcal{F}})|\mathbf{x}) \geq 0.5\}$. It was shown elsewhere [Watkin, 1993; Opper and Haussler, 1991] that in high–dimensional spaces $\mathbf{w}_{\text{Bayes}}$ converges to the point $\mathbf{w}^*$ that is with high probability (over the choice of $\mathbf{x}$) within the set of Bayes–optimal decision functions.

3 Billiards in Kernel Space

In this section we present a BPM algorithm for estimating the Bayes point by the center of mass[4]. The

[3] Note that this function need not necessarily be contained in the original set of functions.

[4] For further details see [Herbrich *et al.*, 1999a].

approach utilized is similar to the method presented in [Rujàn, 1997]: in order to obtain the center of mass of $\mathcal{V}(S)$ we randomly generate points (hyperplanes in input space) and average over them. Since it is very difficult to generate hyperplanes consistent with S we average over the trajectory of a ball which is placed inside $\mathcal{V}(S)$ and bounced like a billiard ball. Note that the boundaries of the billiard are given by the hyperplanes having normal vectors $y_i\phi(\mathbf{x}_i)$. This process converges to the center of mass if our billiard is *ergodic* w.r.t. the uniform distribution in $\mathcal{V}(S)$. Although we cannot prove this property, we introduce a randomization step in our algorithm which is expected to produce ergodic billiards.

3.1 Notation

Based on the fact that we only play billiards in version space we know that for each position $\mathbf{b}$, direction vector $\mathbf{v}$ of the ball, and each estimate $\mathbf{w}_n$ of the center of mass of $\mathcal{V}(S)$ we can write

$$\mathbf{b} = \sum_{i=1}^{\ell} \gamma_i \phi(\mathbf{x}_i) \quad \gamma_i \in \mathbb{R}^{\ell}\,, \tag{4}$$

$$\mathbf{v} = \sum_{i=1}^{\ell} \beta_i \phi(\mathbf{x}_i) \quad \beta_i \in \mathbb{R}^{\ell}\,, \tag{5}$$

$$\mathbf{w}_n = \sum_{i=1}^{\ell} \alpha_i \phi(\mathbf{x}_i) \quad \alpha_i \in \mathbb{R}^{\ell}\,. \tag{6}$$

Due to the uniqueness constraint we have to rescale these vectors several times to unit length. This can be achieved by virtue of

$$\|\mathbf{b}\|_{\mathcal{F}}^2 = \sum_{i=1}^{\ell}\sum_{j=1}^{\ell} \gamma_i\gamma_j k(\mathbf{x}_i,\mathbf{x}_j)\,, \tag{7}$$

and similarly for $\mathbf{v}$ and $\mathbf{w}_n$. At the beginning we assume that $\mathbf{w}_0 = \mathbf{0} \Leftrightarrow \boldsymbol{\alpha} = \mathbf{0}$.

3.2 Playing Billiards

Once we have a starting point $\mathbf{b}_0$ inside version space the algorithm can be subdivided into three steps

1. Determine the closest boundary starting from $\mathbf{b}$ into direction $\mathbf{v}$.
2. Update the ball's position $\mathbf{b}'$ at reflection and calculate the new direction vector $\mathbf{v}'$.
3. Update the center of mass of the trajectory by the new line segment from $\mathbf{b}$ to $\mathbf{b}'$ calculated on the Riemannian manifold $\mathcal{V}(S)$.

Bouncing the ball Since it is very complicated to compute the flight time of the ball *on* the Riemannian manifold we can make use of the fact that the distances in Euclidean and Riemannian spaces are order–preserving (if the Euclidean distance exists). Thus, we have for the flight time τ_j of the ball at position $\mathbf{b}$ in direction $\mathbf{v}$ to the hyperplane with normal vector $y_j\phi(\mathbf{x}_j)$

$$\begin{aligned} d_j &= y_j \sum_{i=1}^{\ell} \gamma_i k(\mathbf{x}_i,\mathbf{x}_j) \\ \nu_j &= y_j \sum_{i=1}^{\ell} \beta_i k(\mathbf{x}_i,\mathbf{x}_j) \\ \tau_j &= -\frac{d_j}{\nu_j} \end{aligned} \tag{8}$$

After computing all ℓ flight times, we look for the smallest positive, i.e.

$$m = \arg\min_{j:\tau_j>0} \tau_j\,.$$

Update the ball position and the direction vector The new point $\mathbf{b}'$ and the new direction $\mathbf{v}'$ are calculated from

$$\mathbf{b}' = \mathbf{b} + \tau_m \mathbf{v} = \sum_{i=1}^{\ell} (\gamma_i + \tau_m\beta_i)\phi(\mathbf{x}_i) \tag{9}$$

$$\begin{aligned} \mathbf{v}' &= \mathbf{v} - 2\nu_m y_m \frac{\phi(\mathbf{x}_m)}{\|\phi(\mathbf{x}_m)\|_{\mathcal{F}}^2} \\ &= \sum_{i=1}^{\ell} \left(\beta_i - \delta_{im}\frac{2\nu_i y_i}{k(\mathbf{x}_i,\mathbf{x}_i)}\right)\phi(\mathbf{x}_i)\,. \end{aligned} \tag{10}$$

Afterwards the position $\mathbf{b}'$ and the direction vector $\mathbf{v}'$ need to be normalized. This can easily be achieved using Equation (7).

Updating the Center of Mass Since our solution $\mathbf{w}_\infty$ has to live in $\mathcal{V}(S)$ we cannot simply take the weighted vector addition to update $\mathbf{w}$. Let us introduce the operation $\oplus_\mu$ acting on vectors of unit length. This function has to have the following properties

$$\begin{aligned} \|\mathbf{s} \oplus_\mu \mathbf{t}\|_{\mathcal{F}}^2 &= 1\,, \\ \|\mathbf{s} - \mathbf{s} \oplus_\mu \mathbf{t}\|_{\mathcal{F}} &= \mu\|\mathbf{s}-\mathbf{t}\|_{\mathcal{F}}\,, \\ \mathbf{s} \oplus_\mu \mathbf{t} &= \varrho_1(\mathbf{s},\mathbf{t},\mu)\mathbf{s} + \varrho_2(\mathbf{s},\mathbf{t},\mu)\mathbf{t} \\ \varrho_1(\mathbf{s},\mathbf{t},\mu) \geq 0 &\quad , \quad \varrho_2(\mathbf{s},\mathbf{t},\mu) \geq 0\,. \end{aligned}$$

A few lines of algebra then give the following formulas for ϱ_1 and ϱ_2

$$\begin{aligned} \varrho_1(\mathbf{s},\mathbf{t},\mu) &= \mu\sqrt{-\frac{\mu^2 - \mu^2\langle\mathbf{s},\mathbf{t}\rangle_{\mathcal{F}} - 2}{\langle\mathbf{s},\mathbf{t}\rangle_{\mathcal{F}} + 1}}\,, \\ \varrho_2(\mathbf{s},\mathbf{t},\mu) &= -\varrho_1(\mathbf{s},\mathbf{t},\mu)\langle\mathbf{s},\mathbf{t}\rangle_{\mathcal{F}} \\ &\quad \pm[\mu^2(1-\langle\mathbf{s},\mathbf{r}\rangle_{\mathcal{F}}) - 1]\,. \end{aligned}$$

By assuming a constant line density on the manifold $\mathcal{V}(S)$ the whole line between $\mathbf{b}$ and $\mathbf{b}'$ can be represented by the midpoint $\mathbf{m}$ on the manifold $\mathcal{V}(S)$ given by

$$\mathbf{m} = \frac{\mathbf{b}+\mathbf{b}'}{\|\mathbf{b}+\mathbf{b}'\|_{\mathcal{F}}}\,. \tag{11}$$

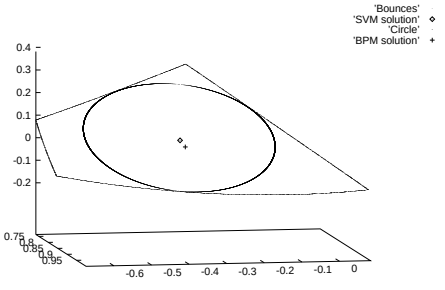

Figure 1: Shown is the version space $\mathcal{V}(S)$ for a 3D–toy problem. One can see that the approximation of the Bayes point by the center of the largest inscribed sphere is reasonable. For further details see the text.

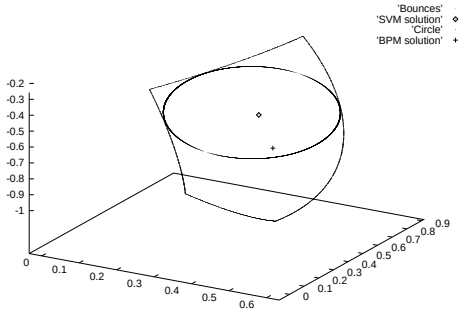

Figure 2: Shown is the version space $\mathcal{V}(S)$ and the largest inscribed sphere found by the SVM. Here, the approximation for the whole version space is bad. This also results in a significant difference in the generalization errors (see text).

Thus, one updates the center of mass of the trajectory by

$$\mathbf{w}_{n+1} = \varrho_1\left(\mathbf{w}_n, \mathbf{m}, \frac{\Lambda_n}{\Lambda_n+\lambda_n}\right)\mathbf{w}_n + \varrho_2\left(\mathbf{w}_n, \mathbf{m}, \frac{\Lambda_n}{\Lambda_n+\lambda_n}\right)\mathbf{m}\,, \qquad (12)$$

where we have used the $\lambda_n = \|\mathbf{b}'_n - \mathbf{b}_n\|_{\mathcal{F}}$ for the length of the trajectory in the n–th step and $\Lambda_n = \sum_{i=1}^{n}\lambda_i$ for the accumulated length up to the n–th step.

Exceptions and stopping criterion The only approximate step in the algorithm is made by computing the closest bounding hyperplane in Euclidean space rather than Riemannian space. This can cause problems if the curvature of the manifold is almost orthogonal to the direction vector $\mathbf{v}$ in which case $\tau_m \to \infty$. Here, we suggest randomly generating a direction vector pointing *towards* version space. Assuming that the last bounce took place at the hyperplane having normal $y_m\phi(\mathbf{x}_m)$ this can easily be checked by

$$y_m\langle \mathbf{v}, \phi(\mathbf{x}_m)\rangle_{\mathcal{F}} = y_m\sum_{i=1}^{\ell}\beta_i k(\mathbf{x}_i, \mathbf{x}_m) > 0\,. \qquad (13)$$

This kind of "reset" also has the advantage of introducing a randomization to the billiard which is expected to produce an ergodic billiard.

As a stopping criterion we suggest computing an upper bound on ϱ_2, the weighting factor of the new part of the trajectory. If this value falls below a prespecified threshold (TOL) we stop the algorithm. Note that the increase in Λ_n will always lead to termination.

4 Experiments

For the purpose of visualization we randomly generated two datasets having 10 training and 10000 test points in $\mathbb{R}^3$. The data points were labeled by a randomly generated linear decision rule using the kernel $k(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}'\rangle_{\mathbb{R}^3}$. We ran SVM learning and used this solution as the starting point $\mathbf{b}_0$ of the BPM which was terminated by setting TOL to 0.001 (around 7000 bounces). By tracking all the positions $\mathbf{b}_n$ of the billiard ball we could easily visualize the version space (see Figure 1 and 2). From Figure 1 it can be seen that the spherical approximation of $\mathcal{V}(S)$ by SVMs was reasonable which consequently resulted in a very small generalization error estimated on the test set (SVM: 6.5%, BPM: 6.1%). The situation dramatically changes if the version space is more elongated as can be seen in Figure 2. Here the SVM solution and the Bayes point are far apart, which results in a notable decrease of the generalization error using the Bayes point (SVM: 15.1%, BPM: 8.0%). Note, that in both cases the data points were normalized to unit length.

To study the generalization performance on real world data we used 7 standard datasets. These were `heart`, `thyroid`, `diabetes`, `sonar`, and `ionosphere` from the UCI Repository [UCI, 1990], and `banana` and `waveform`, two toy dataset studied by[5] [Rätsch *et al.*, 1998]. In each case the data has been randomly partitioned into 100 training and test sets generally in the ratio 60%:40%. The means and standard deviations of the average generalization errors on the test sets are presented as percentages in the columns headed SVM and BPM in Table 1. As in the toy example we used the hard margin SVM (see Equation (3)). In every experiment we used RBF kernels $k(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x}-\mathbf{x}'\|^2/2\sigma^2)$ with σ specified in the third column. We fixed the tolerance TOL for termination of the BPM algorithm to 0.01. As can be seen, the BPM outperforms the SVM on almost all datasets at a statistically significant level. In the fourth column of Table 1 the p–values of a paired t–test for *BPM is better than SVM* are given. This justifies the advantage of using the Bayes point rather than the center of the

[5]These datasets have been made publically available at http://horn.first.gmd.de/∼raetsch/data/benchmarks.htm.

	SVM	BPM	σ	p-value
Heart	25.4±0.40	**22.8±0.34**	10.0	1.00
Thyroid	5.3±0.24	**4.4±0.21**	3.00	1.00
Diabetes	33.1±0.24	**32.0±0.25**	5.0	1.00
Waveform	13.0±0.10	**12.1±0.09**	20.0	1.00
Banana	16.2±0.15	**15.1±0.14**	0.5	1.00
Sonar	**15.4±0.37**	15.9±0.38	1.0	0.01
Ionosphere	11.9±0.25	**11.5±0.25**	1.5	0.99

Table 1: Experimental results on seven benchmark datasets. The standard deviation was obtained on 100 different runs. See the text for details.

largest inscribed sphere.

5 Discussion and Conclusion

In this paper we presented an estimation method for the Bayes point considering linear functions in Hilbert space. We showed how the SVM can be viewed as a (spherical) approximation method to the Bayes point hyperplane. By randomly generating consistent hyperplanes playing billiards in the version space we showed how to stochastically approximate this point. In the field of Markov Chain Monte Carlo methods such approaches are known as *reflective slice sampling* [Neal, 1997]. Current investigations in this field include the question of ergodicity of such methods.

We would like to emphasize that an interesting property of the algorithm was the ongoing decrease of the test error even though we always enforced zero training error. This phenomenon was also observed in the application of boosting methods (c.f. [Schapire *et al.*, 1997]). For boosting this could be explained by the maximization of the margin in the class of convex combinations of base classifiers. In fact, we see that playing billiards in kernel space can also be viewed as averaging of base classifiers given by all the midpoints $\mathbf{m}$ (Section 2 and 3). Hence, our current investigation of theoretical results for the Bayes point are made in a similar way (see [Herbrich *et al.*, 1999b]). The difference to the studies made so far (see, e.g. [Cristianini *et al.*, 1998; Schapire *et al.*, 1997]) is the fact that instead of averaging binary classifiers the Bayes point is obtained as an average of real valued classifiers. This recommends the use of margin distribution bounds rather than hard margin bounds (see [Shawe-Taylor and Cristianini, 1998]). Note that in our analysis the convex hull of $\mathcal{H}_k$ coincides with $\mathcal{H}_k$ itself which consequently changes the notion of a margin compared to SVMs. Investigations of the generalization bounds for the Bayes point are also necessary for resolving a major drawback of the presented method, namely its limitation to zero training error.

Acknowledgments

We are greatly indebted to discussions with Søren Fiig Jarner, Klaus Obermayer, Craig Saunders, John Shawe–Taylor, and Jason Weston. We would also like to thank Alex Smola for providing his Support Vector implementation.

References

[Cristianini *et al.*, 1998] Nello Cristianini, John Shawe-Taylor, and Peter Sykacek. Bayesian classifiers are large margin hyperplanes in a hilbert space. Technical report, Royal Holloway, University of London, 1998. NC2–TR–1998–008.

[Herbrich *et al.*, 1999a] Ralf Herbrich, Thore Graepel, and Colin Campbell. Bayesian learning in reproducing kernel hilbert spaces – the usefulness of the bayes point. Technical report, Technical University Berlin, 1999. in preparation.

[Herbrich *et al.*, 1999b] Ralf Herbrich, Thore Graepel, and Klaus Obermayer. Regression models for ordinal data: A machine learning approach. Technical report, TU Berlin, 1999. TR-99/03.

[Neal, 1997] Radford M. Neal. Markov chain monte carlo method based on 'slicing' the density function. Technical report, Department of Statistics, University of Toronto, 1997. TR–9722.

[Opper and Haussler, 1991] Manfred Opper and David Haussler. Generalization performance of bayes optimal classification algorithm for learning a perceptron. *Physical Review Letters*, 66:2677, 1991.

[Rätsch *et al.*, 1998] Gunnar Rätsch, Takashi Onoda, and Klaus-Robert Müller. Soft margins for adaboost. Technical report, Royal Holloway, University of London, 1998. NC–TR–1998–021.

[Rujàn, 1997] Pàl Rujàn. Playing billiard in version space. *Neural Computation*, 9:99–122, 1997.

[Schapire *et al.*, 1997] Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. In *Proccedings of the 14–th International Conference in Machine Learning*, 1997.

[Shawe-Taylor and Cristianini, 1998] John Shawe-Taylor and Nello Cristianini. Robust bounds on generalization from the margin distribution. Technical report, Royal Holloway, University of London, 1998. NC2–TR–1998–029.

[Shawe-Taylor and Williamson, 1997] John Shawe-Taylor and Robert C. Williamson. A PAC analysis of a Bayesian estimator. Technical report, Royal Holloway, University of London, 1997. NC2–TR–1997–013.

[Shawe-Taylor *et al.*, 1996] John Shawe-Taylor, Peter L. Bartlett, Robert C. Williamson, and Martin Anthony. Structural risk minimization over data–dependent hierarchies. Technical report, Royal Holloway, University of London, 1996. NC–TR–1996–053.

[UCI, 1990] UCI. University of California Irvine: Machine Learning Repository, 1990.

[Vapnik, 1998] Vladimir Vapnik. *Statistical Learning Theory.* John Wiley and Sons, New York, 1998.

[Watkin, 1993] T. Watkin. Optimal learning with a neural network. *Europhysics Letters*, 21:871–877, 1993.

[Whaba, 1990] Grace Whaba. *Spline Models for Observational Data.* Society for Industrial and Applied Mathematics, Philadelphia, 1990.