The Set Covering Machine with Data-Dependent Half-Spaces

Mario Marchand MARIO.MARCHAND@IFT.ULAVAL.CA
Département d'Informatique, Université Laval, Québec, Canada, G1K-7P4

Mohak Shah MSHAH@SITE.UOTTAWA.CA
School of Information Technology and Engineering, University of Ottawa, Ottawa, Ont., Canada, K1N-6N5

John Shawe-Taylor JST@CS.RHUL.AC.UK
Department of Computer Science, Royal Holloway, University of London, Egham, UK, TW20-0EX

Marina Sokolova SOKOLOVA@SITE.UOTTAWA.CA
School of Information Technology and Engineering, University of Ottawa, Ottawa, Ont., Canada, K1N-6N5

Abstract

We examine the set covering machine when it uses *data-dependent half-spaces* for its set of features and bound its generalization error in terms of the number of training errors and the number of half-spaces it achieves on the training data. We show that it provides a favorable alternative to data-dependent balls on some natural data sets. Compared to the support vector machine, the set covering machine with data-dependent half-spaces produces substantially sparser classifiers with comparable (and sometimes better) generalization. Furthermore, we show that our bound on the generalization error provides an effective guide for model selection.

1. Introduction

The set covering machine (SCM) has recently been proposed by Marchand and Shawe-Taylor (2001; 2002) as an alternative to the support vector machine (SVM) when the objective is to obtain a sparse classifier with good generalization. Given a feature space, the SCM tries to find the smallest conjunction (or disjunction) of features that gives a small training error. In contrast, the SVM tries to find the maximum soft-margin separating hyperplane on all the features. Hence, the two learning machines are fundamentally different in what they are trying to achieve on the training data.

The learning algorithm for SCM generalizes the two-step algorithm of Valiant (1984) and Haussler (1988) for learning conjunctions (and disjunctions) of Boolean attributes to allow features that are constructed from the data and to allow a trade-off between accuracy and complexity. For the set of features known as *data-dependent balls*, Marchand and Shawe-Taylor (2001; 2002) have shown that good generalization is expected when a SCM with a small number of balls and errors can be found on the training data. Furthermore, on some "natural" data sets, they have found that the SCM achieves a much higher level of sparsity than the SVM with roughly the same generalization error.

In this paper, we introduce a new set of features for the SCM that we call *data-dependent half-spaces*. Since our goal is to construct sparse classifiers, we want to avoid using $O(d)$ examples to construct each half-space in a d-dimensional input space (like many computational geometric algorithms). Rather, we want to use $O(1)$ examples for each half-space. In fact, we will see that by using only three examples per half-space, we need very few of these half-spaces to achieve a generalization as good (and sometimes better) as the SVM on many "natural" data sets. Moreover, the level of sparsity achieved by the SCM is always substantially superior (sometimes by a factor of

Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), Washington DC, 2003.

at least 50) than the one achieved by the SVM.

Finally, by extending the sample compression technique of Littlestone and Warmuth (1986), we bound the generalization error of the SCM with data-dependent half-spaces in terms of the number of errors and the number of half-spaces it achieves on the training data. We will then show that, on some "natural" data sets, our bound is as effective as 10-fold cross-validation in its ability to select a good SCM model.

2. The Set Covering Machine

We provide here a short description of the Set Covering Machine (SCM), more details are provided in Marchand and Shawe-Taylor (2002).

Let $\mathbf{x}$ denote an arbitrary n-dimensional vector of the input space X which could be arbitrary subsets of $\Re^n$. We consider binary classification problems for which the training set $S = P \cup N$ consists of a set P of positive training examples and a set N of negative training examples. We define a *feature* as an arbitrary Boolean-valued function that maps X onto $\{0, 1\}$. Given any set $\mathcal{H} = \{h_i(\mathbf{x})\}_{i=1}^{|\mathcal{H}|}$ of features $h_i(\mathbf{x})$ and any training set S, the learning algorithm returns a small subset $\mathcal{R} \subset \mathcal{H}$ of features. Given that subset $\mathcal{R}$, and an arbitrary input vector $\mathbf{x}$, the output $f(\mathbf{x})$ of the SCM is defined to be:

$$f(\mathbf{x}) = \begin{cases} \bigvee_{i \in \mathcal{R}} h_i(\mathbf{x}) & \text{for a disjunction} \\ \bigwedge_{i \in \mathcal{R}} h_i(\mathbf{x}) & \text{for a conjunction} \end{cases}$$

To discuss both the conjunction and the disjunction cases simultaneously, let us use $\mathcal{P}$ to denote set P in the conjunction case but set N in the disjunction case. Similarly, $\mathcal{N}$ denotes set N in the conjunction case but denotes set P in the disjunction case. It then follows that f makes no error with $\mathcal{P}$ if and only if each $h_i \in \mathcal{R}$ makes no error with $\mathcal{P}$. Moreover, if Q_i denotes the subset of examples of $\mathcal{N}$ on which feature h_i makes no errors, then f makes no error on $\mathcal{N}$ if and only if $\bigcup_{i \in \mathcal{R}} Q_i = \mathcal{N}$. Hence, as was first observed by Haussler (1988), the problem of finding the smallest set $\mathcal{R}$ for which f makes no training errors is just the problem of finding the smallest collection of Q_is that covers all $\mathcal{N}$ (where each corresponding h_i makes no error on $\mathcal{P}$). This is the well-known *Minimum Set Cover Problem* (Garey & Johnson, 1979). The interesting fact is that, although it is NP-complete to find the smallest cover, the *set covering greedy algorithm* will always find a cover of size at most $z \ln(|\mathcal{N}|)$ when the smallest cover that exists is of size z (Chvátal, 1979; Kearns & Vazirani, 1994). Moreover this algorithm is very simple to implement and just consists of the following steps: first choose the set Q_i which covers the largest number of elements in $\mathcal{N}$, remove from $\mathcal{N}$ and each Q_j the elements that are in Q_i, then repeat this process of finding the set Q_k of largest cardinality and updating $\mathcal{N}$ and each Q_j until there are no more elements in $\mathcal{N}$.

The SCM built on the features found by the set covering greedy algorithm will make no training errors only when there exists a subset $\mathcal{E} \subset \mathcal{H}$ of features on which a conjunction (or a disjunction) makes zero training error. However, this constraint is not really required in practice since we do want to permit the user of a learning algorithm to control the tradeoff between the accuracy achieved on the training data and the complexity (here the size) of the classifier. Indeed, a small SCM which makes a few errors on the training set might give better generalization than a larger SCM (with more features) which makes zero training errors. One way to include this flexibility into the SCM is to stop the set covering greedy algorithm when there remains a few more training examples to be covered. In this case, the SCM will contain fewer features and will make errors on those training examples that are not covered. But these examples all belong to $\mathcal{N}$ and, in general, we do need to be able to make errors on training examples of both classes. Hence, early stopping is generally not sufficient and, in addition, we need to consider features that also make some errors with $\mathcal{P}$ provided that many more examples in $\mathcal{N}$ can be covered. Hence, for a feature h, let us denote by Q_h the set of examples in $\mathcal{N}$ covered by feature h and by R_h the set of examples in $\mathcal{P}$ for which h makes an error on. Given that each example in $\mathcal{P}$ misclassified by h should decrease by some fixed *penalty* p its "importance", we define the *usefulness* U_h of feature h by:

$$U_h \stackrel{\text{def}}{=} |Q_h| - p \cdot |R_h|$$

Hence, we modify the set covering greedy algorithm in the following way. Instead of using the feature that covers the largest number of examples in $\mathcal{N}$, we use the feature $h \in \mathcal{H}$ that has the highest usefulness value U_h. We removed from $\mathcal{N}$ and each Q_g (for $g \neq h$) the

elements that are in Q_h and we removed from each R_g (for $g \neq h$) the elements that are in R_h. Note that we update each such set R_g because a feature g that makes an error on an example in $\mathcal{P}$ does not increase the error of the machine if another feature h is already making an error on that example. We repeat this process of finding the feature h of largest usefulness U_h and updating $\mathcal{N}$, and each Q_g and R_g, until only a few elements remain in $\mathcal{N}$ (early stopping the greedy).

Here is a formal description of our learning algorithm. The penalty p and the early stopping point s are the two model-selection parameters that give the user the ability to control the proper tradeoff between the training accuracy and the size of the function. Their values could be determined either by using k-fold cross-validation, or by computing our bound (see section 4) on the generalization error based on what has been achieved on the training data. Note that our learning algorithm reduces to the two-step algorithm of Valiant (1984) and Haussler (1988) when both s and p are infinite and when the set of features consists of the set of input attributes and their negations.

Algorithm BuildSCM$(T, P, N, p, s, \mathcal{H})$

Input: A machine type T (which is either "conjunction" or "disjunction"), a set P of positive training examples, a set N of negative training examples, a penalty value p, a stopping point s, and a set $\mathcal{H} = \{h_i(\mathbf{x})\}_{i=1}^{|\mathcal{H}|}$ of Boolean-valued features.

Output: A conjunction (or disjunction) $f(\mathbf{x})$ of a subset $\mathcal{R} \subseteq \mathcal{H}$ of features.

Initialization: $\mathcal{R} = \emptyset$.

1. **If** (T = "conjunction") let $\mathcal{P} \leftarrow P$ and $\mathcal{N} \leftarrow N$. **Else** let $\mathcal{P} \leftarrow N$ and let $\mathcal{N} \leftarrow P$.

2. For each $h_i \in \mathcal{H}$, let Q_i be the subset of $\mathcal{N}$ covered by h_i and let R_i be the subset of $\mathcal{P}$ covered by h_i (*i.e.* examples in $\mathcal{P}$ incorrectly classified by h_i).

3. Let h_k be a feature with the largest value of $|Q_k| - p \cdot |R_k|$. **If** ($|Q_k| = 0$) then go to step 7 (cannot cover the remaining examples in $\mathcal{N}$).

4. Let $\mathcal{R} \leftarrow \mathcal{R} \cup \{h_k\}$. Let $\mathcal{N} \leftarrow \mathcal{N} - Q_k$ and let $\mathcal{P} \leftarrow \mathcal{P} - R_k$.

5. For all i do: $Q_i \leftarrow Q_i - Q_k$ and $R_i \leftarrow R_i - R_k$.

6. **If** ($\mathcal{N} = \emptyset$ or $|\mathcal{R}| \geq s$) then go to step 7 (no more examples to cover or early stopping). **Else** go to step 3.

7. Return $f(\mathbf{x})$ where:

$$f(\mathbf{x}) = \begin{cases} \bigvee_{i \in \mathcal{R}} h_i(\mathbf{x}) & \text{for a disjunction} \\ \bigwedge_{i \in \mathcal{R}} h_i(\mathbf{x}) & \text{for a conjunction} \end{cases}$$

3. Data-Dependent Half-Spaces

With the use of kernels, each input vector $\mathbf{x}$ is implicitly mapped into a high-dimensional vector $\phi(\mathbf{x})$ such that $\phi(\mathbf{x}) \cdot \phi(\mathbf{x}') = k(\mathbf{x}, \mathbf{x}')$ (the kernel trick). We consider the case where each feature is a half-space constructed from a set of 3 points $\{\phi_a, \phi_b, \phi_c\}$ where ϕ_a is the image of a positive example $\mathbf{x}_a$, ϕ_b is the image of a negative example $\mathbf{x}_b$, and ϕ_c is the image of a $\mathcal{P}$-example $\mathbf{x}_c$. The weight vector $\mathbf{w}$ of such an half-space $h^c_{a,b}$ is defined by $\mathbf{w} \stackrel{\text{def}}{=} \phi_a - \phi_b$ and its threshold t is identified by $t \stackrel{\text{def}}{=} \mathbf{w} \cdot \phi_c - \epsilon$, where ϵ is a small positive real number in the case of a conjunction but a small negative number in the case of a disjunction. Hence

$$\begin{aligned} h^c_{a,b}(\mathbf{x}) &\stackrel{\text{def}}{=} \operatorname{sgn}\{\mathbf{w} \cdot \phi(\mathbf{x}) - t\} \\ &= \operatorname{sgn}\{k(\mathbf{x}_a, \mathbf{x}) - k(\mathbf{x}_b, \mathbf{x}) - t\} \end{aligned}$$

where

$$t = k(\mathbf{x}_a, \mathbf{x}_c) - k(\mathbf{x}_b, \mathbf{x}_c) - \epsilon.$$

When the penalty parameter p is set to ∞, **BuildSCM** tries to cover with half-spaces the examples of $\mathcal{N}$ without making any error on the examples of $\mathcal{P}$. In that case, ϕ_c is the image of the example $\mathbf{x}_c \in \mathcal{P}$ that gives the smallest value of $\mathbf{w} \cdot \phi(\mathbf{x}_c)$ in the case of a conjunction (but the largest value of $\mathbf{w} \cdot \phi(\mathbf{x}_c)$ in the case of a disjunction). Note that, in contrast with data-dependent balls (Marchand & Shawe-Taylor, 2002), we are not guaranteed to always be able to cover all $\mathcal{N}$ with such half-spaces. When training a SCM with finite p, any $x_c \in \mathcal{P}$ might give the best threshold for a given $(\mathbf{x}_a, \mathbf{x}_b)$ pair. Hence, to find the half-space that maximizes U_h, we need to compute U_h for every triple $(\mathbf{x}_a, \mathbf{x}_b, \mathbf{x}_c)$.

Note that this set of features (in the linear kernel case $k(\mathbf{x}, \mathbf{x}') = \mathbf{x} \cdot \mathbf{x}'$) was already proposed by Hinton and Revow (1996) for decision tree learning but no analysis of their learning method has been given.

4. Bound on the Generalization Error

First note that we cannot use the "standard" VC theory to bound the generalization error of SCMs with data-dependent half-spaces because this set of functions is defined only *after* obtaining the training data. In contrast, the VC dimension is a property of a function class defined on some input domain *without reference* to the data. Hence, we propose another approach.

Since our learning algorithm tries to build a SCM with the smallest number of data-dependent half-spaces, we seek a bound that depends on this number and, consequently, on the number of examples that are used in the final classifier (the hypothesis). We can thus think of our learning algorithm as compressing the training set into a small subset of examples that we call the *compression set*. It was shown by Littlestone and Warmuth (1986) and Floyd and Warmuth (1995) that we can bound the generalization error of the hypothesis f if we can always reconstruct f from the compression set. Hence, the only requirement is the existence of such a *reconstruction function* and its only purpose is to permit the exact identification of the hypothesis from the compression set and, possibly, additional bits of information. Not surprisingly, the bound on the generalization error raises rapidly in terms of these additional bits of information. So we must make minimal usage of them.

We now describe our reconstruction function and the additional information that it needs to assure, in all cases, the proper reconstruction of the hypothesis from a compression set. As we will see, our proposed scheme works in all cases provided that the learning algorithm returns a hypothesis that always correctly classifies the compression set (but not necessarily all of the training set). Hence, we need to add this constraint in **BuildSCM**[1] for our bound to be valid but, in practice, we have not seen any significant performance variation introduced by this constraint.

[1]For this, it is sufficient to test if a newly added feature does not misclassify any previous $\mathcal{P}$-example in the current compression set.

Given a compression set (returned by the learning algorithm), we first partition it into three disjoint subsets $\Lambda_a, \Lambda_b, \Lambda_c$ that consists of the examples of type $\mathbf{x}_a$, $\mathbf{x}_b$, and $\mathbf{x}_c$ that we have described in section 3. Now, from these sets, we must construct the weight vectors. Recall that each weight vector $\mathbf{w}$ is specified by a pair $(\mathbf{x}_a, \mathbf{x}_b)$. Hence, for each $\mathbf{x}_a \in \Lambda_a$, we must specify the different points $\mathbf{x}_b \in \Lambda_b$ that are used to form a weight vector with $\mathbf{x}_a$. Although each point can participate in more than one weight vector, each pair $(\mathbf{x}_a, \mathbf{x}_b) \in \Lambda_a \times \Lambda_b$ can provide at most one weight vector under the constraint that the compression set is always correctly classified by the hypothesis. Hence, the identification of weight vectors requires at most $\lambda_a \lambda_b$ bits of information (where $\lambda_a = |\Lambda_a|$ and $\lambda_b = |\Lambda_b|$). However, it is more economical to provide instead $\log_2(\lambda_a \lambda_b)$ bits to first specify the number r of weight vectors and then $\log_2 \binom{\lambda_a \lambda_b}{r}$ bits to specify which group of r pairs $(\mathbf{x}_a, \mathbf{x}_b)$ is chosen among the set of all possible groups of r pairs taken from $\Lambda_a \times \Lambda_b$. To find the threshold t for each $\mathbf{w}$, we choose the example $\mathbf{x} \in \Lambda_c \cup \Lambda_a$ that gives the smallest value of $\mathbf{w} \cdot \phi(\mathbf{x})$ in the case of a conjunction. In the case of a disjunction, we choose the example $\mathbf{x} \in \Lambda_c \cup \Lambda_b$ that gives the largest value of $\mathbf{w} \cdot \phi(\mathbf{x})$. This is the only choice that assures that the compression set is always correctly classified by the resulting classifier. Note that we adopt the convention that each point in the compression set is specified only once (without repetitions) and, consequently, a point of Λ_a or Λ_b can also be used to identify the threshold.

In summary, we can always reconstruct the hypothesis from the compression set when we partition it into the subsets $\Lambda_a, \Lambda_b, \Lambda_c$ defined above and provide, in addition, $\log_2(\lambda_a \lambda_b) + \log_2 \binom{\lambda_a \lambda_b}{r}$ bits to extract the weight vectors from $\Lambda_a \times \Lambda_b$. This is all that is required for the next theorem.

Theorem 1 *Let $S = P \cup N$ be a training set of positive and negative examples of size $m = m_p + m_n$. Let A be the learning algorithm* **BuildSCM** *that uses data-dependent half-spaces for its set of features with the constraint that the returned function $A(S)$ always correctly classifies every example in the compression set. Suppose that $A(S)$ contains r half-spaces, and makes k_p training errors on P, k_n training errors on N (with $k = k_p + k_n$), and has a compression*

set $\Lambda = \Lambda_a \cup \Lambda_b \cup \Lambda_c$ (as defined above) of size $\lambda = \lambda_a + \lambda_b + \lambda_c$. With probability $1-\delta$ over all random training sets S of size m, the generalization error $\mathrm{er}(A(S))$ of $A(S)$ is bounded by

$$\mathrm{er}(A(S)) \leq 1 - \exp\left\{\frac{-1}{m-\lambda-k}\left(\ln B_\lambda + \ln\binom{\lambda_a\lambda_b}{r} + \ln(\lambda_a\lambda_b) + \ln\frac{1}{\delta_\lambda}\right)\right\}$$

where

$$\delta_\lambda \stackrel{\text{def}}{=} \delta\cdot\left(\frac{\pi^2}{6}\right)^{-5}\left((\lambda_a+1)(\lambda_b+1)\cdot(\lambda_c+1)(k_p+1)(k_n+1)\right)^{-2}$$

and where

$$B_\lambda \stackrel{\text{def}}{=} \binom{m_p}{\lambda_a}\binom{m_n}{\lambda_b}\binom{m_p-\lambda_a}{\lambda_c}\binom{m_n-\lambda_b}{k_n}\cdot\binom{m_p-\lambda_a-\lambda_c}{k_p} \text{ for conjunctions}$$

$$B_\lambda \stackrel{\text{def}}{=} \binom{m_p}{\lambda_a}\binom{m_n}{\lambda_b}\binom{m_n-\lambda_b}{\lambda_c}\binom{m_p-\lambda_a}{k_p}\cdot\binom{m_n-\lambda_b-\lambda_c}{k_n} \text{ for disjunctions}$$

Proof Let $\mathcal{X}$ be the set of training sets of size m. Let us first bound the probability

$$P_{\mathbf{m}} \stackrel{\text{def}}{=} P\left\{S\in\mathcal{X} : \mathrm{er}(A(S)) \geq \epsilon \mid \mathbf{m}(S) = \mathbf{m}\right\}$$

given that $\mathbf{m}(S)$ is fixed to some value $\mathbf{m}$ where

$$\mathbf{m} \stackrel{\text{def}}{=} (m, m_p, m_n, \lambda_a, \lambda_b, \lambda_c, k_p, k_n).$$

For this, denote by $\mathcal{E}_p$ the subset of P on which $A(S)$ makes an error and similarly for $\mathcal{E}_n$. Let I be the message of information bits needed to specify the weight vectors (as described above) for a given Λ_a and Λ_b. Now define $P'_{\mathbf{m}}$ to be

$$P'_{\mathbf{m}} \stackrel{\text{def}}{=} P\Big\{S\in\mathcal{X} :\ \mathrm{er}(A(S)) \geq \epsilon \mid \Lambda_a = S_1, \Lambda_b = S_2, \Lambda_c = S_3, \mathcal{E}_p = S_4, \mathcal{E}_n = S_5, I = I_0, \mathbf{m}(S) = \mathbf{m}\Big\}$$

for some fixed set of disjoint subsets $\{S_i\}_{i=1}^5$ of S and some fixed information message I_0. Since B_λ is the number of different ways of choosing the different compression subsets and set of error points in a training set of fixed $\mathbf{m}$, we have:

$$P_{\mathbf{m}} \leq \lambda_a\lambda_b\cdot\binom{\lambda_a\lambda_b}{r}\cdot B_\lambda\cdot P'_{\mathbf{m}}$$

where the first two factors come from the additional information that is needed to specify the weight vectors. Note that the hypothesis $f \stackrel{\text{def}}{=} A(S)$ is fixed in $P'_{\mathbf{m}}$ (because the compression set is fixed and the required information bits are given). To bound $P'_{\mathbf{m}}$, we make the standard assumption that each example $\mathbf{x}$ is independently and identically generated according to some fixed but unknown distribution. Let p be the probability of obtaining a positive example, let α be the probability that the fixed hypothesis f makes an error on a positive example, and let β be the probability that f makes an error on a negative example. Let $t_p \stackrel{\text{def}}{=} \lambda_a + \lambda_c + k_p$ for the conjunction case (and $t_p \stackrel{\text{def}}{=} \lambda_a + k_p$ for the disjunction case). Similarly, let $t_n \stackrel{\text{def}}{=} \lambda_b + k_n$ for the conjunction case (and $t_n \stackrel{\text{def}}{=} \lambda_b + \lambda_c + k_n$ for the disjunction case). We then have:

$$\begin{aligned}
P'_{\mathbf{m}} &= (1-\alpha)^{m_p-t_p}(1-\beta)^{m-t_n-m_p}\binom{m-t_n-t_p}{m_p-t_p}p^{m_p-t_p}(1-p)^{m-t_n-m_p}\\
&\leq \sum_{m'=t_p}^{m-t_n}(1-\alpha)^{m'-t_p}(1-\beta)^{m-t_n-m'}\binom{m-t_n-t_p}{m'-t_p}p^{m'-t_p}(1-p)^{m-t_n-m'}\\
&= [(1-\alpha)p + (1-\beta)(1-p)]^{m-t_n-t_p}\\
&= (1-\mathrm{er}(f))^{m-t_n-t_p}\\
&\leq (1-\epsilon)^{m-t_n-t_p}
\end{aligned}$$

Consequently:

$$P_{\mathbf{m}} \leq \lambda_a\lambda_b\cdot\binom{\lambda_a\lambda_b}{r}\cdot B_\lambda\cdot(1-\epsilon)^{m-t_n-t_p}.$$

The theorem is obtained by bounding this last expression by the proposed value for $\delta_\lambda(\mathbf{m})$ and solving for ϵ since, in that case, we satisfy the requirement that:

$$
\begin{aligned}
P\Big\{ S \in \mathcal{X} : \mathrm{er}(A(S)) \geq \epsilon \Big\} & \\
&= \sum_{\mathbf{m}} P_{\mathbf{m}} P\Big\{ S \in \mathcal{X} \colon \mathbf{m}(S) = \mathbf{m} \Big\} \\
&\leq \sum_{\mathbf{m}} \delta_\lambda(\mathbf{m}) P\Big\{ S \in \mathcal{X} \colon \mathbf{m}(S) = \mathbf{m} \Big\} \\
&\leq \sum_{\mathbf{m}} \delta_\lambda(\mathbf{m}) \\
&= \delta
\end{aligned}
$$

where the sums are over all possible realizations of $\mathbf{m}$ for a fixed m_p and m_n. With the proposed value for $\delta_\lambda(\mathbf{m})$, the last equality follows from the fact that $\sum_{i=1}^{\infty}(1/i^2) = \pi^2/6$. ■

In order to obtain the tightest possible bound, note that we have generalized the approach of Littlestone and Warmuth by partitioning the compression set into three different subsets and by taking into account the number of positive and negative examples actually observed in the training set.

Basically, our bound states that good generalization is expected when we can find a small SCM that makes few training errors. It may seem complicated but the important feature is that it depends only on what the hypothesis has *achieved* on the training data. Hence, we could use it as a guide for choosing the model selection parameters s and p of algorithm **BuildSCM** since we can compute its value immediately after training.

5. Empirical Results on Natural data

We have compared the practical performance of the SCM with the Support Vector Machine (SVM) equipped with a Gaussian kernel (also called the Radial Basis Function kernel) of variance $1/\gamma$. We have used the SVM program distributed by the Royal Holloway University of London (Saunders et al., 1998). The data sets used and the results obtained are reported in table 1. All these data sets where obtained from the machine learning repository at UCI, except the Glass data set which was obtained from Rob Holte, now at the University of Alberta. For each data set, we have removed all examples that contained attributes with unknown values (this has reduced substantially the "votes" data set) and we have removed examples with contradictory labels (this occurred only for a few examples in the Haberman data set). The remaining number of examples for each data set is reported in table 1. No other preprocessing of the data (such as scaling) was performed. For all these data sets, we have used the 10-fold cross validation error as an estimate of the generalization error. The values reported are expressed as the total number of errors (*i.e.* the sum of errors over all testing sets). We have ensured that each training set and each testing set, used in the 10-fold cross validation process, was the same for each learning machine (*i.e.* each machine was trained on the same training sets and tested on the same testing sets).

The results reported for the SVM are only those obtained for the best values of the kernel parameter γ and the soft margin parameter C found among an *exhaustive* list of *many* values. The "size" column refers to the average number of support vectors contained in SVM machines obtained from the 10 different training sets of 10-fold cross-validation.

We have reported the results for the SCM with data-dependent balls (Marchand & Shawe-Taylor, 2002) (with the L_2 metric) and the SCM with data-dependent half-spaces (with a linear kernel). In both cases the T column refers to type of the best machine found: c for conjunction, and d for disjunction. The p column refers the best value found for the penalty parameter, and the s column refers the the best stopping point in terms of the number of features (*i.e.* balls and half-spaces respectively). Again, only the values that gave the smallest 10-fold cross-validation error are reported. We have also reported, in the "bound" column, the bound on the generalization error obtained by computing the r.h.s. of the inequality of Theorem 1 (with $\delta = .05$), for each of the 10 different training sets involved in 10-fold cross validation, and multiplying that bound with the size of each testing sets. We see that, although the bound is not tight, it is nevertheless non-trivial. This is to be contrasted with the VC dimension bounds which cannot even be applied for our case since the set of functions supported by the SCM depends on the training data. Furthermore, if we exclude the BreastW data set, we can see in the "ratio" column of table 1 that the ratio of the

Table 1. Data sets and results for SCMs and SVMs.

Data Set		SVM				SCM with balls				SCM with half-spaces					
Name	#exs	γ	C	size	errors	T	p	s	errors	T	p	s	errors	bound	ratio
BreastW	683	0.005	2	58	19	c	1.8	2	**15**	c	1.0	1	18	103	5.72
Votes	52	0.05	15	18	**3**	d	0.9	1	6	c	0.8	1	6	20	3.33
Pima	768	0.002	1	526	203	c	1.1	3	189	c	1.5	3	**175**	607	3.47
Haberman	294	0.01	0.6	146	71	c	1.4	1	71	d	0.7	1	**68**	209	3.07
Bupa	345	0.002	0.2	266	107	d	2.8	9	107	c	1.4	1	**103**	297	2.88
Glass	163	0.8	2	92	34	c	0.85	4	**33**	c	1.05	3	39	113	2.90
Credit	653	0.0006	32	423	190	d	1.2	4	194	d	1.2	3	**148**	513	3.47

bound to the generalization error is remarkably stable even across different learning tasks, suggesting that the bound may indeed work well as a model selection criterion.

The most striking feature in table 1 is the level of sparsity achieved by the SCM in comparison with the SVM. This difference is huge. In particular, the SCMs with half-spaces never contained more than 3 half-spaces (*i.e.* a compression set of at most 9 points). Compared with the SVM, the SCM with half-spaces is more than 50 times sparser than the SVM on the Pima, Bupa, and Credit data sets! The other important feature is that SCMs with half-spaces often provide better generalization than SCMs with balls and SVMs. The difference is substantial on the Credit data set. Hence it is quite clear that data-dependent half-spaces provides an alternative to data-dependent balls for the set of features used by the SCM. Although it is within acceptable bounds[2], the price to pay is extra computation time since triples of points needs to be examined to find a half-space but only pairs of points need to be considered for balls.

We now investigate the extent to which our bound can perform model-selection. More specifically, we want to answer the following question. Given a set of SCMs obtained from **BuildSCM** for various values of the model-selection parameters p and s, is our bound on the generalization error, evaluated on the training set, effective at selecting the SCM that will give the best generalization?

Note that the results reported in table 1 are, in fact, the 10-fold cross validation estimate of the generalization error that is achieved by the model selection strategy that correctly guesses the best values for p and s. This model-selection strategy is, in that sense, optimal (but not realizable). Hence, we will refer to the score obtained in table 1 as those obtained by the optimal model-selection strategy.

[2]It takes less than 20 seconds on a 1.6 GHz PC to train once the SCM with half-spaces on the BreastW data set.

Table 2. Model-selection results.

Data Set	T	MS from 10-fold CV			MS from bound		
		s	errors	std	s	errors	std
BreastW	c	1.2	23	4.7	1.8	25	4.4
Votes	c	1.1	9	3.1	1.0	6	2.8
Pima	c	4.3	191	9.2	3.8	181	11
Haberman	d	1.7	73	5.0	3.8	74	3.8
Bupa	c	2.3	118	6.0	2.5	115	8.2
Glass	c	2.6	50	7.9	2.5	49	7.3
Credit	d	3	157	17	3.5	162	17

The results for the model-selection strategy based on our bound are reported in table 2. Here we have used our bound to select the best SCM among those obtained for various penalty values among a list of fifteen penalty values (that always contained the optimal value) and for all possible sizes s. Also shown in these tables, are the results obtained for the 10-fold cross validation model selection method. This latter method is perhaps the most widely used—here, it consists of using 10-fold cross validation to find the best stopping point s and the best penalty value p on a given training set and then use these best values on the full training set to find the best SCM. Both model selection methods were tested by 10-fold cross validation. Finally, in addition to the error and size (as in the pre-

vious tables), we have also reported a rough estimate of the standard deviation of the error. This estimate was obtained in the following way. We first compute the standard deviation of the generalization error (per example) over the 10 different testing sets and then divide by $\sqrt{10}$ (since the variance of the average of n iid random variables, each with variance σ^2, is σ^2/n). Finally we multiply this estimate by the number of examples in the data set. From the results of table 2, we see that model selection by using our bound is generally as effective as using 10-fold cross validation (but takes substantially less computation time).

6. Conclusion and Outlook

We have introduced a new set of features for the SCM, called data-dependent half-spaces, and have shown that it can provide a favorable alternative to data-dependent balls on some "natural" data sets. Compared with SVMs, our learning algorithm for SCMs with half-spaces produces substantially sparser classifiers (often by a factor of 50) with comparable, and sometimes better, generalization.

By extending the sample compression technique of Littlestone and Warmuth (1986), we have bound the generalization error of the SCM with data-dependent half-spaces in terms of the number of errors and the number of half-spaces it achieves on the training data. Our bound indicates that good generalization error is expected whenever a SCM, with a small number of half-spaces, makes few training errors. Furthermore, on some "natural" data sets, we have seen that our bound is generally as effective as 10-fold cross validation at selecting a good SCM model. Note, however, that our bound applies only to the case of symmetrical loss. Hence, the next important step is to generalize our bound to the case of asymmetrical loss (which frequently occurs in practice) and investigate its effectiveness at performing model selection.

Acknowledgments

Work supported by NSERC grant OGP0122405 and, in part, under the KerMIT project, No. IST-2001-25431.

References

Chvátal, V. (1979). A greedy heuristic for the set covering problem. *Mathematics of Operations Research*, *4*, 233–235.

Floyd, S., & Warmuth, M. (1995). Sample compression, learnability, and the Vapnik-Chervonenkis dimension. *Machine Learning*, *21*, 269–304.

Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability, a guide to the theory of np-completeness*. New York, NY: Freeman.

Haussler, D. (1988). Quantifying inductive bias: AI learning algorithms and Valiant's learning framework. *Artificial Intelligence, 36*, 177–221.

Hinton, G., & Revow, M. (1996). Using pairs of data-points to define splits for decision trees. *Advances in Neural Information Processing Systems 8*, 507–513.

Kearns, M. J., & Vazirani, U. V. (1994). *An introduction to computational learning theory*. Cambridge, Massachusetts: MIT Press.

Littlestone, N., & Warmuth, M. (1986). *Relating data compression and learnability* (Technical Report). University of California Santa Cruz, Santa Cruz, CA.

Marchand, M., & Shawe-Taylor, J. (2001). Learning with the set covering machine. *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*, 345–352.

Marchand, M., & Shawe-Taylor, J. (2002). The set covering machine. *Journal of Machine Learning Reasearch*, *3*, 723–746.

Saunders, C., Stitson, O., Weston, J., Bottou, L., Schoelkopf, B., & Smola, A. (1998). *Support vector machine reference manual* (Technical Report CSD-TR-98-03). Department of Computer Science, Royal Holloway, University of London, London, UK.

Valiant, L. G. (1984). A theory of the learnable. *Communications of the Association of Computing Machinery*, *27*, 1134–1142.