Journal of Machine Learning Research 6 (2005) 427–451 Submitted 9/04; Published 4/05

Learning with Decision Lists of Data-Dependent Features

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

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

Editor: Manfred K. Warmuth

Abstract

We present a learning algorithm for decision lists which allows features that are constructed from the data and allows a trade-off between accuracy and complexity. We provide bounds on the generalization error of this learning algorithm in terms of the number of errors and the size of the classifier it finds on the training data. We also compare its performance on some natural data sets with the set covering machine and the support vector machine. Furthermore, we show that the proposed bounds on the generalization error provide effective guides for model selection.

Keywords: decision list machines, set covering machines, sparsity, data-dependent features, sample compression, model selection, learning theory

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 attempts to find the smallest conjunction (or disjunction) of features that gives a small training error. In contrast, the SVM attempts to find the maximum soft-margin separating hyperplane on all the features. Hence, the two learning machines are fundamentally different in what they are aiming to achieve on the training data.

To investigate if it is worthwhile to consider larger classes of functions than just the conjunctions and disjunctions that are used in the SCM, we focus here on the class of decision lists (Rivest, 1987) because this class strictly includes both conjunctions and disjunctions while being strictly included in the class of linear threshold functions (Ehrenfeucht and Haussler, 1989; Blum and Singh, 1990; Marchand and Golea, 1993).

From a theoretical point of view, the class of decision lists has been extensively studied (Rivest, 1987; Dhagat and Hellerstein, 1994; Eiter et al., 2002; Anthony, 2004) and a few learning algorithms have been proposed. The first learning algorithm, due to Rivest (1987), PAC learns the class of decision lists (also known as 1-decision lists) over the input attributes but may return a classifier that depends on all the input attributes even when the number r of relevant attributes is much smaller than the total number n of attributes. Dhagat and Hellerstein (1994) and Kivinen et al. (1992) have proposed an attribute efficient algorithm that outputs a decision list of $O(r^k \log^k m)$ attributes for a training set of m examples where k denotes the number of *alternations* of the target decision list (see the definition in Section 2). However, both of these algorithms are unattractive for the practitioner because they do not provide an accuracy-complexity tradeoff. Indeed, real-world data are often noisy and, therefore, simpler functions that make some training errors might be better than more complex functions that make no training errors. Since the amount of noise is problem-specific, a learning algorithm should provide to the user a means to control the tradeoff between accuracy and the complexity of a classifier. Ideally, the user should be able to choose from a wide range of functions that includes very simple functions (like constants), that almost always underfit the data, and very complex functions that often overfit the data. But this latter requirement for decision lists can be generally achieved only if the set of features used for the decision list is data-dependent. It is only with a data-dependent set of features that a restricted class of functions like decision lists can almost always overfit any training data set (that does not contain too many pairs of identical examples with opposite classification labels). Hence, in this paper, we present a learning algorithm for decision lists which can be used with any set of features, including those that are defined with respect to the training data, and that provides some "model selection parameters" (also called learning parameters) for allowing the user to choose the proper tradeoff between accuracy and complexity.

We denote by *decision list machine* (DLM) any classifier which computes a decision list of Boolean-valued features, including features that are possibly constructed from the data. In this paper, we use the set of features known as data-dependent balls (Marchand and Shawe-Taylor, 2001; Sokolova et al., 2003) and the set of features known as data-dependent half-spaces (Marchand et al., 2003). We show, on some natural data sets, that the DLM can provide better generalization than the SCM with the same set of features.

We will see that the proposed learning algorithm for the DLM with data-dependent features is effectively compressing the training data into a small subset of examples which is called the *compression set*. Hence, we will show that the DLM with data-dependent features is an example of a sample-compression algorithm and we will thus propose a general risk bound that depends on the number of examples that are used in the final classifier and the size of the information message needed to identify the final classifier from the compression set. The proposed bound will apply to any compression set-dependent distribution of messages (see the definition in Section 4) and allows for the message set to be of variable size (in contrast with the sample compression bound of Littlestone and Warmuth (1986) that requires fixed size). We will apply this general risk bound to DLMs by making appropriate choices for a compression set-dependent distribution of messages and we will show, on natural data sets, that these specialized risk bounds are generally slightly more effective than K-fold cross validation for selecting a good DLM model.

This paper extends the previous preliminary results of Sokolova et al. (2003).

2. The Decision List Machine

Let $\mathbf{x}$ denote an arbitrary n-dimensional vector of the input space $\mathcal{X}$ which is an arbitrary subset of $\mathbb{R}^n$. 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 $\mathcal{X}$ 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 decision list machine (DLM) is given by the following rule

If $(h_1(\mathbf{x}))$ then b_1

Else If $(h_2(\mathbf{x}))$ then b_2

$\ldots$

Else If $(h_r(\mathbf{x}))$ then b_r

Else b_{r+1},

where each $b_i \in \{0,1\}$ defines the output of $f(\mathbf{x})$ if and only if h_i is the first feature to be satisfied on $\mathbf{x}$ (*i.e.* the smallest i for which $h_i(\mathbf{x}) = 1$). The constant b_{r+1} (where $r = |\mathcal{R}|$) is known as the *default value*. Note that f computes a disjunction of the h_is whenever $b_i = 1$ for $i = 1 \ldots r$ and $b_{r+1} = 0$. To compute a conjunction of h_is, we simply place in f the negation of each h_i with $b_i = 0$ for $i = 1 \ldots r$ and $b_{r+1} = 1$. Note, however, that a DLM f that contains one or many *alternations* (*i.e.* a pair (b_i, b_{i+1}) for which $b_i \neq b_{i+1}$ for $i < r$) cannot be represented as a (pure) conjunction or disjunction of h_is (and their negations). Hence, the class of decision lists strictly includes conjunctions and disjunctions.

We can also easily verify that decision lists are a proper subset of linear threshold functions in the following way. Given a DLM with r features as above, we assign a weight value w_i to each h_i in the DLM in order to satisfy

$$|w_i| > \sum_{j=i+1}^{r} |w_j| \quad \forall i \in \{1, \ldots, r\}.$$

Let us satisfy these constraints with $|w_i| = 2^{r-i}$ for $i \in \{1, \ldots, r\}$. Then, for each i, we set $w_i = +|w_i|$ if $b_i = 1$, otherwise we set $w_i = -|w_i|$ if $b_i = 0$. For the threshold θ we use $\theta = -1/2$ if $b_{r+1} = 1$ and $\theta = +1/2$ if $b_{r+1} = 0$. With this prescription, given any example $\mathbf{x}$, we always have that

$$\operatorname{sgn}\left(\sum_{i=1}^{r} w_i h_i(\mathbf{x}) - \theta\right) \;=\; 2b_k - 1,$$

where k is the smallest integer for which $h_k(\mathbf{x}) = 1$ in the DLM or $k = r+1$ if $h_i(\mathbf{x}) = 0 \; \forall i \in \{1, \ldots, r\}$. Hence, with this prescription, the output of the linear threshold function is the same as the output of the DLM for all input $\mathbf{x}$. Finally, to show that the subset is proper we simply point out that a majority vote of three features is a particular case of a linear threshold function that cannot be represented as a decision list since the output of the majority vote cannot be determined from the value of a single feature.

From our definition of the DLM, it seems natural to use the following greedy algorithm for building a DLM from a training set. For a given set $S' = P' \cup N'$ of examples (where $P' \subseteq P$ and $N' \subseteq N$) and a given set $\mathcal{H}$ of features, consider only the features $h_i \in \mathcal{H}$ which either have $h_i(\mathbf{x}) = 0$ for all $\mathbf{x} \in P'$ or $h_i(\mathbf{x}) = 0$ for all $\mathbf{x} \in N'$. Let Q_i be the subset of examples on which $h_i = 1$ (our constraint on the choice of h_i implies that Q_i contains only examples having the same class label). We say that h_i is *covering* Q_i. The greedy algorithm starts with $S' = S$ and an empty DLM. Then it finds a h_i with the largest $|Q_i|$ and appends (h_i, b) to the DLM (where b is the class label of the examples in Q_i). It then removes Q_i from S' and repeats to find the h_k with the largest $|Q_k|$ until either P' or N' is empty. It finally assigns b_{r+1} to the class label of the remaining non-empty set of examples.

Following Rivest (1987), this greedy algorithm is assured to build a DLM that makes no training errors whenever *there exists* a DLM on a set $\mathcal{E} \subseteq \mathcal{H}$ of features that makes zero training errors. 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 DLM which makes a few errors on the training set might give better generalization than a larger DLM (with more features) which makes zero training errors. One way to include this flexibility is to early-stop the greedy algorithm when there remains a few more training examples to be covered. But a further reduction in the size of the DLM can be accomplished by considering features h_i that cover examples of both classes. Indeed, if Q_i denotes the subset of S' on which $h_i = 1$ (as before), let P_i denote the subset of P' that belongs to Q_i and let N_i be the subset of N' that belongs to Q_i (thus $Q_i = P_i \cup N_i$). In the previous greedy algorithm, we were considering only features h_i for which either P_i or N_i was empty. Now we are willing to consider features for which neither P_i nor N_i is empty whenever $\max(|P_i|, |N_i|)$ is substantially larger than before. In other words, we want now to consider features that may err on a few examples whenever they can cover many more examples. We therefore define the *usefulness* U_i of feature h_i by

$$U_i \stackrel{\text{def}}{=} \max\{|P_i| - p_n|N_i|,\ |N_i| - p_p|P_i|\},$$

where p_n denotes the *penalty* of making an error on a negative example whereas p_p denotes the penalty of making an error on a positive example. Indeed, whenever we add to a DLM a feature h_i for which P_i and N_i are both non empty, the output b_i associated with h_i will be 1 if $|P_i| - p_n|N_i| \geq |N_i| - p_p|P_i|$ or 0 otherwise. Hence, the DLM will necessarily incorrectly classify the examples in N_i if $b_i = 1$ or the examples in P_i if $b_i = 0$.

Hence, to include this flexibility in choosing the proper tradeoff between complexity and accuracy, each greedy step will be modified as follows. For a given training set $S' = P' \cup N'$, we will select a feature h_i with the largest value of U_i and append $(h_i, 1)$ to the DLM if $|P_i| - p_n|N_i| \geq |N_i| - p_p|P_i|$, otherwise, we append $(h_i, 0)$ to the DLM. If $(h_i, 1)$ was appended, we will then remove from S' every example in P_i (since they are correctly classified by the current DLM) *and* we will also remove from S' every example in N_i (since a DLM with this feature is already misclassifying N_i, and, consequently, the training error of the DLM will not increase if later features err on the examples in N_i). Similarly if $(h_i, 0)$ was appended, we will then remove from S' the examples in $Q_i = N_i \cup P_i$. Hence, we recover the simple greedy algorithm when $p_p = p_n = \infty$.

The formal description of our learning algorithm is presented in Figure 1. Note that we always set $b_{r+1} = \neg b_r$ since, otherwise, we could remove the rth feature without changing the classifier's output f for any input $\mathbf{x}$.

The penalty parameters p_p and p_n and the early stopping point s of **BuildDLM** are the model-selection parameters that give the user the ability to control the proper tradeoff between the training accuracy and the size of the DLM. Their values could be determined either by using K-fold cross-validation, or by computing the risk bounds proposed below. It therefore generalizes the learning algorithm of Rivest (1987) by providing this complexity-accuracy tradeoff and by permitting the use of any kind of Boolean-valued features, including those that are constructed from the training data.

```
Algorithm BuildDLM($(S, p_p, p_n, s, \mathcal{H})$)

Input: A set $S$ of examples, the penalty values $p_p$ and $p_n$, a stopping point $s$, and a set
$\mathcal{H} = \{h_i(\mathbf{x})\}_{i=1}^{|\mathcal{H}|}$ of Boolean-valued features.

Output: A decision list $f$ consisting of an ordered set $\mathcal{R} = \{(h_i, b_i)\}_{i=1}^{r}$ of features $h_i$ with their
corresponding output values $b_i$, and a default value $b_{r+1}$.

Initialization: $\mathcal{R} = \emptyset,\ r = 0,\ S' = S,\ b_0 = \neg a$ (where $a$ is the label of the majority class).

  1. For each $h_i \in \mathcal{H}$, let $Q_i = P_i \cup N_i$ be the subset of $S'$ for which $h_i = 1$ (where $P_i$ consists of
     positive examples and $N_i$ consists of negative examples). For each $h_i$ compute $U_i$, where:

         $U_i \overset{\text{def}}{=} \max\{|P_i| - p_n|N_i|,\ |N_i| - p_p|P_i|\}$

  2. Let $h_k$ be a feature with the largest value of $U_k$. If $Q_k = \emptyset$ then go to step 6 (no progress
     possible).

  3. If ($|P_k| - p_n|N_k| \geq |N_k| - p_p|P_k|$) then append $(h_k, 1)$ to $\mathcal{R}$. Else append $(h_k, 0)$ to $\mathcal{R}$.

  4. Let $S' = S' - Q_k$ and let $r = r + 1$.

  5. If ($r < s$ and $S'$ contains examples of both classes) then go to step 1

  6. Set $b_{r+1} = \neg b_r$. Return $f$.
```

Figure 1: The learning algorithm for the decision list machine

The time complexity of **BuildDLM** is trivially bounded as follows. Assuming a time of at most t for evaluating one feature on one example, it takes a time of at most $|\mathcal{H}|mt$ to find the first feature of the DLM for a training set of m examples. For the data-dependent set of features presented in Section 3.1, it is (almost) always possible to find a feature that covers at least one example. In that case, it takes a time of $O(|\mathcal{H}|mst)$ to find s features. Note that the algorithm must stop if, at some greedy step, there does not exists a feature that covers at least one training example.

Generally, we can further reduce the size of the DLM by observing that any feature h_i with $b_i = b_{r+1}$ can be deleted from the DLM if there does not exist a training example $\mathbf{x}$ with label $y = b_{r+1}$ and another feature h_j with $j > i$ and $b_j \neq b_i$ for which $h_i(\mathbf{x}) = h_j(\mathbf{x}) = 1$ (since, in that case, feature h_i can be moved to the end of the DLM without changing the output for any correctly classified training example). The algorithm **PruneDLM** of Figure 2 deletes all such nodes from the DLM.

We typically use both algorithms in the following way. Given a training set, we first run **Build-DLM** without early stopping (*i.e.*, with parameter s set to infinity) to generate what we call a *template* DLM. Then we consider all the possible DLMs that can be obtained by truncating this template. More precisely, if the template DLM contains r features, we build $r + 1$ possible DLMs: the DLM that contains zero features (a constant function), the DLM that contains the first feature only,

```
Algorithm PruneDLM$(S, f)$

Input: A set $S$ of examples, a decision list $f$ consisting of an ordered set $\mathcal{R} = \{(h_i, b_i)\}_{i=1}^{r}$ of
features $h_i$ with their corresponding output values $b_i$, and a default value $b_{r+1}$.

Output: The same decision list $f$ with, possibly, some features removed.

Initialization: $l = r$

  1. Let $(h_k, b_k) \in \mathcal{R}$ be the pair with the largest value of $k$ such that $b_k = b_{r+1}$ and $k < l$.
  2. If $(\nexists (h_j, b_j) \in \mathcal{R} : \; j > k,\; b_j \neq b_k,\; h_j(\mathbf{x}) = h_k(\mathbf{x})$ for some $(\mathbf{x}, y) \in S$ with $y = b_{r+1})$ then
     delete $(h_k, b_k)$ from $\mathcal{R}$.
  3. $l = k$.
  4. If $(l > 1)$ then go to step 1; else stop.
```

Figure 2: The pruning algorithm for the decision list machine

the DLM that contains the first two features, and so on, up to the DLM that contains all r features. Then we run **PruneDLM** on all these DLMs to try to reduce them further. Finally all these DLMs are tested on a provided testing set.

It is quite easy to build artificial data sets for which **PruneDLM** decreases substantially the size of the DLM. However, for the natural data sets used in Section 7, **PruneDLM** almost never deleted any node from the DLM returned by **BuildDLM**.

3. Data-Dependent Features

The set of *data-dependent balls* (Marchand and Shawe-Taylor, 2001) and *data-dependent half-spaces* (Marchand et al., 2003) were introduced for their usage with the SCM. We now need to adapt their definitions for using them with the DLM.

3.1 Balls and Holes

Let $d : \mathcal{X}^2 \to \mathbb{R}$ be a metric for our input space $\mathcal{X}$. Let $h_{\mathbf{c},\rho}$ be a feature identified by a *center* $\mathbf{c}$ and a *radius* ρ. Feature $h_{\mathbf{c},\rho}$ is said to be a *ball* iff $h_{\mathbf{c},\rho}(\mathbf{x}) = 1 \; \forall \mathbf{x}\colon \; d(\mathbf{x}, \mathbf{c}) \leq \rho$ and 0 otherwise. Similarly, feature $h_{\mathbf{c},\rho}$ is said to be a *hole* iff $h_{\mathbf{c},\rho}(\mathbf{x}) = 1 \; \forall \mathbf{x}\colon \; d(\mathbf{x}, \mathbf{c}) > \rho$ and 0 otherwise. Hence, a ball is a feature that covers the examples that are located inside the ball; whereas a hole covers the examples that are located outside. In general, both types of features will be used in the DLM.

Partly to avoid computational difficulties, we are going to restrict the centers of balls and holes to belong to the training set, *i.e.*, each center $\mathbf{c}$ must be chosen among $\{\mathbf{x}_i : (\mathbf{x}_i, y_i) \in S\}$ for a given training set S. Moreover, given a center $\mathbf{c}$, the set of relevant radius values are given by the positions of the other training examples, *i.e.*, the relevant radius values belong to $\{d(\mathbf{c}, \mathbf{x}_i) : (\mathbf{x}_i, y_i) \in S\}$. Hence, each ball and hole is identified by only two training examples: a center $\mathbf{c}$ and a *border* $\mathbf{b}$ that identifies the radius with $d(\mathbf{c}, \mathbf{b})$. Therefore, a DLM made of these two-example features is effectively compressing the training data into the smaller set of examples used for its features. This is the other reason why we have constrained the centers and radii to these values. Hence, given a training set S of m examples, the set $\mathcal{H}$ of features used by the DLM will contain $O(m^2)$ balls and holes. This is a data-dependent set of features since the features are defined with respect to the training data S.

Whenever a ball (or hole) $h_{\mathbf{c},\rho}$ is chosen to be appended to the DLM, we must also provide an output value b which will be the output of the DLM on example $\mathbf{x}$ when $h_{\mathbf{c},\rho}$ is the first feature of the DLM that has $h_{\mathbf{c},\rho}(\mathbf{x}) = 1$. In this paper we always choose b to be the class label of $\mathbf{c}$ if $h_{\mathbf{c},\rho}$ is a ball. If $h_{\mathbf{c},\rho}$ is a hole, then we always choose b to be the negation of the class label of $\mathbf{c}$. We have not explored the possibility of using balls and holes with an output *not* given by the class label of its center because, as we will see later, this would have required an additional information bit in order to reconstruct the ball (or hole) from its center and border and, consequently, would have given a looser generalization error bound without providing additional discriminative power (*i.e.*, power to fit the data) that seemed "natural".

To avoid having examples directly on the decision surface of the DLM, the radius ρ of a ball of center $\mathbf{c}$ will always be given by $\rho = d(\mathbf{c},\mathbf{b}) - \varepsilon$ for some training example $\mathbf{b}$ chosen for the border and some fixed and very small positive value ε. Similarly, the radius of a hole of center $\mathbf{c}$ will always be given by $\rho = d(\mathbf{c},\mathbf{b}) + \varepsilon$. We have not chosen to assign the radius values "in between" two training example since this would have required three examples per ball and hole and would have decreased substantially the tightness of our generalization error bound without providing a significant increase of discriminative power.

With these choices for centers and radii, it is straightforward to see that, for any penalty values p_p and p_n, the set of balls having the largest usefulness U always contains a ball with a center and border of opposite class labels whereas the set of holes having the largest usefulness always contains a hole having a center and border of the same class label. Hence, we will only consider such balls and holes in the set of features for the DLM. For a training set of m_p positive examples and m_n negative examples we have exactly $2m_pm_n$ such balls and $m_p^2 + m_n^2$ such holes. We thus provide to **BuildDLM** a set $\mathcal{H}$ of at most $(m_p + m_n)^2$ features.

Finally, note that this set of features has the property that there always exists a DLM of these features that correctly classifies all the training set S provided that S does not contain a pair of contradictory examples, *i.e.*, $(\mathbf{x},y)$ and $(\mathbf{x}',y')$ such that $\mathbf{x} = \mathbf{x}'$ and $y \neq y'$. Therefore, this feature set gives to the user the ability to choose the proper tradeoff between training accuracy and function size.

3.2 Half-Spaces

With the use of kernels, each input vector $\mathbf{x}$ is implicitly mapped into a high-dimensional vector $\boldsymbol{\phi}(\mathbf{x})$ such that $\boldsymbol{\phi}(\mathbf{x}) \cdot \boldsymbol{\phi}(\mathbf{x}') = k(\mathbf{x},\mathbf{x}')$ (the kernel trick). We consider the case where each feature is a half-space constructed from a set of 3 points $\{\boldsymbol{\phi}_a,\boldsymbol{\phi}_b,\boldsymbol{\phi}_c\}$ where each $\boldsymbol{\phi}_l$ is the image of an input example $\mathbf{x}_l$ taken from the training set S. We consider the case where $\mathbf{x}_a$ and $\mathbf{x}_b$ have opposite class labels and the class label of $\mathbf{x}_c$ is the same as the class label of $\mathbf{x}_b$. The weight vector $\mathbf{w}$ of such a half-space $h^c_{a,b}$ is defined by $\mathbf{w} \overset{\text{def}}{=} \boldsymbol{\phi}_a - \boldsymbol{\phi}_b$ and its threshold t by $t \overset{\text{def}}{=} \mathbf{w} \cdot \boldsymbol{\phi}_c + \varepsilon$ where ε is a small positive real number. We use $\varepsilon > 0$ to avoid having examples directly on the decision surface of the DLM. Hence

$$h^c_{a,b}(\mathbf{x}) \overset{\text{def}}{=} \operatorname{sgn}\{\mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{x}) - t\} = \operatorname{sgn}\{k(\mathbf{x}_a,\mathbf{x}) - k(\mathbf{x}_b,\mathbf{x}) - t\},$$

where

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

Whenever a half-space $h^c_{a,b}$ is chosen to be appended to the DLM, we must also provide an output value b which will be the output of the DLM on example $\mathbf{x}$ when $h^c_{a,b}$ is the first feature of the DLM having $h^c_{a,b}(\mathbf{x}) = 1$. From our definition above, we choose b to be the class label of $\boldsymbol{\phi}_a$. Hence, a DLM made of these three-example features is effectively compressing the training set into the smaller set of examples used for its features.

Given a training set S of $m = m_p + m_n$ examples, the set $\mathcal{H}$ of features considered by the DLM will contain at most $m \cdot m_p \cdot m_n$ half-spaces. However, in contrast with the set of balls and holes, we are not guaranteed to always be able to cover all the training set S with these half-spaces.

Finally, 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 formal analysis of their learning method has been given.

4. A Sample Compression Risk Bound

Since our learning algorithm tries to build a DLM with the smallest number of data-dependent features, and since each feature is described in terms of small number of training examples (two for balls and holes and three for half-spaces), 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*.

Hence, in this section, we provide a general risk bound that depends on the number of examples that are used in the final classifier and the size of the information message needed to identify the final classifier from the compression set. Such a risk bound was first obtained by Littlestone and Warmuth (1986). The bound provided here allows the message set to be of variable size (whereas previous bounds require fixed size). In the next section, we will compare this bound with other well known bounds. Later, we apply this general risk bound to DLMs by making appropriate choices for a compression set-dependent distribution of messages. Finally, we will show, on natural data sets, that these specialized risk bounds provide an effective guide for choosing the model-selection parameters of **BuildDLM**.

Recall that we consider binary classification problems where the input space $\mathcal{X}$ consists of an arbitrary subset of $\mathbb{R}^n$ and the output space $\mathcal{Y} = \{0,1\}$. An example $\mathbf{z} \stackrel{\text{def}}{=} (\mathbf{x}, y)$ is an input-output pair where $\mathbf{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$. We are interested in learning algorithms that have the following property. Given a training set $S = \{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$ of m examples, the classifier $A(S)$ returned by algorithm A is described entirely by two *complementary sources of information*: a subset $\mathbf{z_i}$ of S, called the *compression set*, and a *message string* σ which represents the additional information needed to obtain a classifier from the compression set $\mathbf{z_i}$. This implies that there exists a *reconstruction function* $\mathcal{R}$, associated to A, that outputs a classifier $\mathcal{R}(\sigma, \mathbf{z_i})$ when given an arbitrary compression set $\mathbf{z_i}$ and message string σ chosen from the set $\mathcal{M}(\mathbf{z_i})$ of all distinct messages that can be supplied to $\mathcal{R}$ with the compression set $\mathbf{z_i}$. It is only when such an $\mathcal{R}$ exists that the classifier returned by $A(S)$ is *always* identified by a compression set $\mathbf{z_i}$ and a message string σ.

Given a training set S, the compression set $\mathbf{z_i}$ is defined by a vector $\mathbf{i}$ of indices such that

$$\begin{aligned} \mathbf{i} &\stackrel{\text{def}}{=} (i_1, i_2, \ldots, i_{|\mathbf{i}|}) \qquad (1)\\ \text{with} &: i_j \in \{1, \ldots, m\}\ \forall j \\ \text{and} &: i_1 < i_2 < \ldots < i_{|\mathbf{i}|}, \end{aligned}$$

where $|\mathbf{i}|$ denotes the number of indices present in $\mathbf{i}$.

The classical perceptron learning rule and support vector machines are examples of learning algorithms where the final classifier can be reconstructed solely from a compression set (Graepel et al., 2000, 2001). In contrast, we will see in the next section that the reconstruction function for DLMs needs both a compression set and a message string.

We seek a tight risk bound for arbitrary reconstruction functions that holds uniformly for all compression sets and message strings. For this, we adopt the PAC setting where each example $\mathbf{z}$ is drawn according to a fixed, but unknown, probability distribution D on $\mathcal{X} \times \mathcal{Y}$. The risk $R(f)$ of any classifier f is defined as the probability that it misclassifies an example drawn according to D:

$$R(f) \stackrel{\text{def}}{=} \Pr_{(\mathbf{x},y)\sim D}\left(f(\mathbf{x}) \neq y\right) = \mathbf{E}_{(\mathbf{x},y)\sim D} I(f(\mathbf{x}) \neq y),$$

where $I(a) = 1$ if predicate a is true and 0 otherwise. Given a training set $S = \{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$ of m examples, the *empirical risk* $R_S(f)$ on S, of any classifier f, is defined according to

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

Let $\mathbf{Z}^m$ denote the collection of m random variables whose instantiation gives a training sample $S = \mathbf{z}^m = \{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$. Let us denote $\Pr_{\mathbf{Z}^m \sim D^m}(\cdot)$ by $\mathbf{P}_{\mathbf{Z}^m}(\cdot)$. The basic method to find a bound on the true risk of a learning algorithm A, is to bound P' where

$$P' \quad \stackrel{\text{def}}{=} \quad \mathbf{P}_{\mathbf{Z}^m}\left(R(A(\mathbf{Z}^m)) > \varepsilon\right). \tag{2}$$

Our goal is to find the smallest value for ε such that $P' \leq \delta$ since, in that case, we have

$$\mathbf{P}_{\mathbf{Z}^m}\left(R(A(\mathbf{Z}^m)) \leq \varepsilon\right) \geq 1 - \delta.$$

Recall that classifier $A(\mathbf{z}^m)$ is described entirely in terms of a compression set $\mathbf{z_i} \subset \mathbf{z}^m$ and a message string $\sigma \in \mathcal{M}(\mathbf{z_i})$. Let I be the set of all 2^m vectors of indices $\mathbf{i}$ as defined by Equation 1. Let $\mathcal{M}(\mathbf{z_i})$ be the set of all messages σ that can be attached to compression set $\mathbf{z_i}$. We assume that the empty message is always present in $\mathcal{M}(\mathbf{z_i})$ so that we always have $|\mathcal{M}(\mathbf{z_i})| \geq 1$. Since any $\mathbf{i} \in I$ and $\sigma \in \mathcal{M}(\mathbf{z_i})$ could *a priori* be reached by classifier $A(\mathbf{z}^m)$, we bound P' by the following probability

$$P' \quad \leq \quad \mathbf{P}_{\mathbf{Z}^m}\left(\exists \mathbf{i} \in I \colon \exists \sigma \in \mathcal{M}(\mathbf{Z_i}) \colon R(\mathcal{R}(\sigma, \mathbf{Z_i})) > \varepsilon\right) \stackrel{\text{def}}{=} P'',$$

where $\mathbf{Z_i}$ are the random variables whose instantiation gives $\mathbf{z_i}$ and where ε depends on $\mathbf{Z_i}, \sigma$ and the amount of training errors. In the sequel, we denote by $\bar{\mathbf{i}}$ the vector of indices made of all the indices not present in $\mathbf{i}$. Since $\mathbf{P}_{\mathbf{Z}^m}(\cdot) = \mathbf{E}_{\mathbf{Z_i}} \mathbf{P}_{\mathbf{Z_{\bar{i}}}|\mathbf{Z_i}}(\cdot)$, we have (by the union bound)

$$\begin{aligned} P'' \quad &\leq \quad \sum_{\mathbf{i}\in I} \mathbf{E}_{\mathbf{Z_i}} \mathbf{P}_{\mathbf{Z_{\bar{i}}}|\mathbf{Z_i}}\left(\exists \sigma \in \mathcal{M}(\mathbf{Z_i}) \colon R(\mathcal{R}(\sigma, \mathbf{Z_i})) > \varepsilon\right) \\ &\leq \quad \sum_{\mathbf{i}\in I} \mathbf{E}_{\mathbf{Z_i}} \sum_{\sigma \in \mathcal{M}(\mathbf{Z_i})} \mathbf{P}_{\mathbf{Z_{\bar{i}}}|\mathbf{Z_i}}\left(R(\mathcal{R}(\sigma, \mathbf{Z_i})) > \varepsilon\right). \end{aligned} \tag{3}$$

We will now stratify $\mathbf{P}_{\mathbf{Z_{\bar{i}}}|\mathbf{Z_i}}(R(\mathcal{R}(\sigma, \mathbf{Z_i})) > \varepsilon)$ in terms of the errors that $\mathcal{R}(\sigma, \mathbf{Z_i})$ can make on the *training examples that are not used for the compression set*. Let $I_\mathbf{i}$ denote the set of vectors of indices where each index is not present in $\mathbf{i}$. Given a training sample $\mathbf{z}^m$ and a compression set $\mathbf{z_i}$, we denote by $\mathbf{R}_{\mathbf{z}_{\bar{\mathbf{i}}}}(f)$ the vector of indices pointing to the examples in $\mathbf{z}_{\bar{\mathbf{i}}}$ which are misclassified by f. We have

$$\mathbf{P}_{\mathbf{Z}_{\bar{\mathbf{i}}}|\mathbf{Z}_\mathbf{i}}\left(R(\mathcal{R}(\sigma,\mathbf{Z_i}))>\varepsilon\right)=\sum_{\mathbf{j}\in I_\mathbf{i}}\mathbf{P}_{\mathbf{Z}_{\bar{\mathbf{i}}}|\mathbf{Z}_\mathbf{i}}\left(R(\mathcal{R}(\sigma,\mathbf{Z_i}))>\varepsilon,\mathbf{R}_{\mathbf{Z}_{\bar{\mathbf{i}}}}(\mathcal{R}(\sigma,\mathbf{Z_i}))=\mathbf{j}\right). \tag{4}$$

But now, since the classifier $\mathcal{R}(\sigma,\mathbf{Z_i})$ is fixed when $(\sigma,\mathbf{Z_i})$ is fixed, and since each $\mathbf{Z}_i$ is independent and identically distributed according to the same (but unknown) distribution D, we have

$$\mathbf{P}_{\mathbf{Z}_{\bar{\mathbf{i}}}|\mathbf{Z}_\mathbf{i}}\left(R(\mathcal{R}(\sigma,\mathbf{Z_i}))>\varepsilon,\mathbf{R}_{\mathbf{Z}_{\bar{\mathbf{i}}}}(\mathcal{R}(\sigma,\mathbf{Z_i}))=\mathbf{j}\right)\leq(1-\varepsilon)^{m-|\mathbf{i}|-|\mathbf{j}|}. \tag{5}$$

Hence, by using Equations 3, 4, and 5, we have

$$P'' \quad \leq \quad \sum_{\mathbf{i}\in I}\sum_{\mathbf{j}\in I_\mathbf{i}}\mathbf{E}_{\mathbf{Z_i}}\sum_{\sigma\in\mathcal{M}(\mathbf{Z_i})}\left[1-\varepsilon(\sigma,\mathbf{Z_i},\mathbf{j})\right]^{m-|\mathbf{i}|-|\mathbf{j}|}, \tag{6}$$

where we have now shown explicitly the dependence of ε on $\mathbf{Z_i}$, σ, and $\mathbf{j}$.

Given any compression set $\mathbf{z_i}$, let us now use any function $P_{\mathcal{M}(\mathbf{z_i})}(\sigma)$ which has the property that

$$\sum_{\sigma\in\mathcal{M}(\mathbf{z_i})}P_{\mathcal{M}(\mathbf{z_i})}(\sigma)\leq 1 \tag{7}$$

and can, therefore, be interpreted as compression set-dependent distribution of messages when it sums to one. Let us then choose ε such that

$$\binom{m}{|\mathbf{i}|}\binom{m-|\mathbf{i}|}{|\mathbf{j}|}\left[1-\varepsilon(\sigma,\mathbf{Z_i},|\mathbf{j}|)\right]^{m-|\mathbf{i}|-|\mathbf{j}|}=P_{\mathcal{M}(\mathbf{Z_i})}(\sigma)\cdot\zeta(|\mathbf{i}|)\cdot\zeta(|\mathbf{j}|)\cdot\delta, \tag{8}$$

where, for any non-negative integer a, we define

$$\zeta(a)\stackrel{\text{def}}{=}\frac{6}{\pi^2}(a+1)^{-2}. \tag{9}$$

In that case, we have indeed that $P'\leq\delta$ since $\sum_{i=1}^{\infty}i^{-2}=\pi^2/6$. Any choice for $\zeta(a)$ is allowed as long as it is a non negative function who's sum is bounded by 1.

The solution to Equation 8 is given by

$$\begin{aligned}\varepsilon(\sigma,\mathbf{Z_i},|\mathbf{j}|,\delta) \quad = \quad & 1-\exp\left(\frac{-1}{m-|\mathbf{i}|-|\mathbf{j}|}\left[\ln\binom{m}{|\mathbf{i}|}+\ln\binom{m-|\mathbf{i}|}{|\mathbf{j}|}+\ln\left(\frac{1}{P_{\mathcal{M}(\mathbf{Z_i})}(\sigma)}\right)+\right.\right.\\ & \left.\left.\ln\left(\frac{1}{\zeta(|\mathbf{i}|)\zeta(|\mathbf{j}|)\delta}\right)\right]\right).\end{aligned} \tag{10}$$

We have therefore shown the following theorem:

Theorem 1 *For any $\delta\in(0,1]$ and for any sample compression learning algorithm with a reconstruction function $\mathcal{R}$ that maps arbitrary subsets of a training set and information messages to classifiers, we have*

$$\mathbf{P}_{\mathbf{Z}^m}\left\{\forall\mathbf{i}\in I,\sigma\in\mathcal{M}(\mathbf{Z_i})\colon R(\mathcal{R}(\sigma,\mathbf{Z_i}))\leq\varepsilon(\sigma,\mathbf{Z_i},|\mathbf{j}|,\delta)\right\}\geq 1-\delta.$$

Although the risk bound given by Theorem 1 (and Equation 10) increases with the amount $|\mathbf{j}|$ of training errors made on the examples that do not belong to the compression set $\mathbf{z_i}$, it is interesting to note that it is *independent* of the amount of errors made on the compression set. However, a reconstruction function will generally need less additional information when it is constrained to produce a classifier making no errors with the compression set. Hence, the above risk bound will generally be smaller for sample-compression learning algorithms that always return a classifier making no errors on the compression set. But this constraint might, in turn, force the learner to produce classifiers with larger compression sets.

Finally note that the risk bound is small for classifiers making a small number $|\mathbf{j}|$ of training errors, having a small compression set size $|\mathbf{i}|$, and having a message string σ with large prior "probability" $P_{\mathcal{M}(\mathbf{z_i})}(\sigma)$. This "probability" is usually larger for short message strings since larger message strings are usually much more numerous at sharing the same "piece" (or fraction) of probability.

5. Comparison with Other Risk Bounds

Although the risk bound of Theorem 1 is basically a sample compression bound it, nevertheless, applies to a much broader class of learning algorithms than just sample compression learning algorithms. Indeed the risk bound depends on two complementary sources of information used to identify the classifier: the sample compression set $\mathbf{z_i}$ and the message string σ. In fact, the bound still holds when the sample compression set vanishes and when the classifier $h = \mathcal{R}(\sigma)$ is described entirely in terms of a message string σ. It is therefore worthwhile to compare the risk bound of Theorem 1 to other well-known bounds.

5.1 Comparison with Data-Independent Bounds

The risk bound of Theorem 1 can be qualified as "data-dependent" when the learning algorithm is searching among a class of functions (classifiers) described in terms of a subset $\mathbf{z_i}$ of the training set. Nevertheless, the bound still holds when the class of functions is "data-independent" and when individual functions of this class are identified only in terms of a (data-independent) message σ. In that limit, $|\mathbf{i}| = 0$ and the risk bound ε depends only on σ and the number $|\mathbf{j}| = k$ of training errors:

$$\varepsilon(\sigma, k, \delta) = 1 - \exp\left(\frac{-1}{m-k}\left[\ln\binom{m}{k} + \ln\left(\frac{1}{P_{\mathcal{M}}(\sigma)}\right) + \ln\left(\frac{1}{\zeta(k)\delta}\right)\right]\right). \tag{11}$$

Since here each classifier h is given by $\mathcal{R}(\sigma)$ for some $\sigma \in \mathcal{M}$, we can consider $\mathcal{M}$ as defining a data-independent set of classifiers. This set may contain infinitely many classifiers but it must be countable. Indeed all that is required is

$$\sum_{\sigma \in \mathcal{M}} P(\sigma) \leq 1$$

for any fixed prior P over $\mathcal{M}$. If we further restrict the learning algorithm to produce a classifier that always make no training errors ($k = 0$) and if we choose $P(\sigma) = 1/|\mathcal{M}| \quad \forall \sigma \in \mathcal{M}$ for some finite set $\mathcal{M}$, we obtain the famous Occam's razor bound (Blumer et al., 1987)

$$\varepsilon(\delta) = 1 - \exp\left(\frac{-1}{m}\left[\ln\left(\frac{|\mathcal{M}|}{\delta}\right)\right]\right) \leq \frac{1}{m}\left[\ln\left(\frac{|\mathcal{M}|}{\delta}\right)\right], \tag{12}$$

where we have used $1-\exp(-x) \le x$. Hence the bound of Equation 11 is a generalization of the Occam's razor bound to the case of an arbitrary (but fixed) prior $P(\sigma)$ over a countably infinite set $\mathcal{M}$ of classifiers which are possibly making some training errors. Consequently, the bound of Theorem 1 is a generalization of the Occam's razor bound to the case where the classifiers are identified by two complementary sources of information: the message string σ and the compression set $\mathbf{z_i}$.

The proposed bound is obtained by using a union bound over the possible compression subsets of the training set and over the possible messages $\sigma \in \mathcal{M}(\mathbf{z_i})$. This bound therefore fails when we consider a continuous set of classifiers. In view of the fact that the set of DLMs of data-dependent features is a subset of the same class of functions but with features that are not constrained to be identified by pairs or triples of training examples, why not use the well-known Vapnik-Chervonenkis (VC) bounds (Vapnik, 1998) or Rademacher bounds (Mendelson, 2002) to characterize the learning algorithms discussed in this paper? The reason is that the proposed algorithms are indeed constrained to use a data-dependent set of features identified by pairs and triples of training examples. The risk bound of Theorem 1 therefore reflects more the set of possible classifiers that can be produced by the proposed algorithms than the VC or Rademacher bounds which are suited for algorithms that can produce any classifier of a continuous set.

5.2 Comparison with Other Sample Compression Risk Bounds

The risk bound of Theorem 1 can be reduced to the sample compression bounds of Littlestone and Warmuth (1986) if we perform the following changes and specializations:

- We restrict the set $\mathcal{M}$ of possible messages to be a finite set which is the same for all possible compression sets $\mathbf{z_i}$.
- For the distribution of messages, we use[1]

$$P_{\mathcal{M}}(\sigma) = \frac{1}{|\mathcal{M}|} \quad \forall \sigma \in \mathcal{M}.$$

- Theorem 1 is valid for any function ζ that satisfies $\sum_{i=0}^{m} \zeta(i) \le 1$. Here we will use $\zeta(a) = 1/(m+1) \quad \forall a \in \{0,\ldots,m\}$. This choice increases the bound since $6\pi^{-2}(a+1)^{-2} > 1/(m+1)$ for $a < \sqrt{6(m+1)/\pi} - 1$.
- We use the approximation $1-\exp(-x) \le x$ to obtain a looser (but somewhat easier to understand) bound.

With these restrictions and changes we obtain the following bounds for $|\mathbf{j}| = 0$ and $|\mathbf{j}| \ge 0$:

$$\epsilon(|\mathbf{i}|,\delta) \le \frac{1}{m-|\mathbf{i}|}\left[\ln\binom{m}{|\mathbf{i}|} + \ln\left(\frac{|\mathcal{M}|}{\delta}\right) + \ln(m+1)\right] \quad \text{for } |\mathbf{j}| = 0, \tag{13}$$

$$\epsilon(|\mathbf{i}|,|\mathbf{j}|,\delta) \le \frac{1}{m-|\mathbf{i}|-|\mathbf{j}|}\left[\ln\binom{m}{|\mathbf{i}|} + \ln\binom{m-|\mathbf{i}|}{|\mathbf{j}|} + \ln\left(\frac{|\mathcal{M}|}{\delta}\right) + 2\ln(m+1)\right] \quad \text{for } |\mathbf{j}| \ge 0. \tag{14}$$

1. The case of $|\mathcal{M}| = 1$ (no message strings used) is also treated by Floyd and Warmuth (1995).

Apart from the $\ln(m+1)$ terms, these bounds are the same as the sample compression bounds of Littlestone and Warmuth (1986). The $\ln(m+1)$ terms are absent from the Littlestone and Warmuth compression bounds because their bounds hold uniformly for all compression sets of a *fixed size* $|\mathbf{i}|$ and for all configurations of training error points of a fixed amount $|\mathbf{j}|$. A $\ln(m+1)$ term occurs in the bound of Equation 13 from the extra requirement to hold uniformly for all compression set sizes. Still an extra $\ln(m+1)$ term occurs in Equation 14 from the extra requirement to hold uniformly for all amounts $|\mathbf{j}|$ of training errors. The bound of Theorem 1 holds uniformly for all compression sets of arbitrary sizes and for all configurations of training error points of an arbitrary amount. But instead of using $\zeta(a) = 1/(m+1) \quad \forall a \in \{0,\ldots,m\}$ we have used the tighter form given by Equation 9.

It is also interesting to compare the bounds of Equations 13 and 14 with the sample compression bounds given by Theorems 5.17 and 5.18 of Herbrich (2002). The bound of Equation 13 is the same as the bound of Theorem 5.17 of Herbrich (2002) when $|\mathcal{M}| = 1$ (no messages used). When the classifier is allowed to make training errors, the bound of Equation 14 is tighter than the lossy compression bound of Theorem 5.18 of Herbrich (2002) when $|\mathbf{j}| \ll m$ since the latter have used the Hoeffding inequality which becomes tight only when $|\mathbf{j}|$ is close to $m/2$.

Consequently, the bound of Theorem 1 is tighter than the above-mentioned sample compression bounds for three reasons. First, the approximation $1-\exp(-x) \leq x$ was not performed. Second, the function $\zeta(a)$ of Equation 9 was used instead of the looser factor of $1/(m+1)$. Third, in contrast with the other sample compression bounds, the bound of Theorem 1 is valid for any *a priori* defined sample compression-dependent distribution of messages $P_{\mathcal{M}(\mathbf{z_i})}(\sigma)$.

This last characteristic may be the most important contribution of Theorem 1. Indeed, we feel that it is important to allow the set of possible messages and the message set size to depend on the sample compression $\mathbf{z_i}$ since the class labels of the compression set examples give information about the set of possible data-dependent features that can be constructed from $\mathbf{z_i}$. Indeed, it is conceivable that for some $\mathbf{z_i}$, very little extra information may be needed to identify the classifier whereas for some other $\mathbf{z_i}$, more information may be needed. Consider, for example, the case where the compression set consists of two examples that are used by the reconstruction function $\mathcal{R}$ to obtain a single-ball classifier. For the reconstruction function of the set covering machine (described in the next section), a ball border must be a positive example whereas both positive and negative examples are allowed for ball centers. In that case, if the two examples in the compression set have a positive label, the reconstruction function needs a message string of at least one bit that indicates which example is the ball center. If the two examples have opposite class labels, then the negative example is necessarily the ball center and no message at all is needed to reconstruct the classifier. More generally, the set of messages that we use for all types of DLMs proposed in this paper depends on some properties of $\mathbf{z_i}$ like its number $n(\mathbf{z_i})$ of negative examples. Without such a dependency on $\mathbf{z_i}$, the set of possible messages $\mathcal{M}$ could be unnecessarily too large and would then loosen the risk bound.

5.3 Comparison with the Set Covering Machine Risk Bound

The risk bound for the set covering machine (SCM) (Marchand and Shawe-Taylor, 2001, 2002) is not a general-purposed sample compression risk bound as the one provided by Theorem 1. It does exploit the fact that the final classifier is partly identified by a small subset of the training examples (the compression set) but, instead of using messages to provide the additional information needed to obtain a classifier, it partitions the compression set into three disjoint sets. Hence, we cannot compare directly the bound of Theorem 1 with the SCM risk bound since the latter is much more specialized than the former. Instead we will show how we can apply the general risk bound of Theorem 1 to the case of the SCM just by choosing an appropriate sample compression-dependent distribution of messages $P_{\mathcal{M}(\mathbf{z_i})}(\sigma)$.

Recall that the task of the SCM is to construct the smallest possible conjunction of (Boolean-valued) features. We discuss here only the conjunction case. The disjunction case is treated similarly just by exchanging the role of the positive with the negative examples.

For the case of data-dependent balls and holes, each feature is identified by a training example called a *center* $(\mathbf{x}_c, y_c)$, and another training example called a *border* $(\mathbf{x}_b, y_b)$. Given any metric d, the output $h(\mathbf{x})$ on any input example $\mathbf{x}$ of such a feature is given by

$$h(\mathbf{x}) = \begin{cases} y_c & \text{if } d(\mathbf{x},\mathbf{x}_c) \leq d(\mathbf{x},\mathbf{x}_b) \\ \neg y_c & \text{otherwise.} \end{cases}$$

In this case, given a compression set $\mathbf{z_i}$, we need to specify the examples in $\mathbf{z_i}$ that are used for a border point without being used as a center. As explained by Marchand and Shawe-Taylor (2001), no additional amount of information is required to pair each center with its border point whenever the reconstruction function $\mathcal{R}$ is constrained to produce a classifier that always correctly classifies the compression set. Furthermore, as argued by Marchand and Shawe-Taylor (2001), we can limit ourselves to the case where each border point is a positive example. In that case, each message $\sigma \in \mathcal{M}(\mathbf{z_i})$ just needs to specify the positive examples that are a border point without being a center. Let $n(\mathbf{z_i})$ and $p(\mathbf{z_i})$ be, respectively, the number of negative and the number of positive examples in compression set $\mathbf{z_i}$. Let $b(\sigma)$ be the number of border point examples specified in message σ and let $\zeta(a)$ be defined by Equation 9. We can then use

$$P_{\mathcal{M}(\mathbf{Z_i})}(\sigma) = \zeta(b(\sigma)) \cdot \binom{p(\mathbf{z_i})}{b(\sigma)}^{-1} \tag{15}$$

since, in that case, we have for any compression set $\mathbf{z_i}$:

$$\sum_{\sigma \in \mathcal{M}(\mathbf{z_i})} P_{\mathcal{M}(\mathbf{z_i})}(\sigma) = \sum_{b=0}^{p(\mathbf{z_i})} \zeta(b) \sum_{\sigma : b(\sigma)=b} \binom{p(\mathbf{z_i})}{b(\sigma)}^{-1} \leq 1.$$

With this distribution $P_{\mathcal{M}(\mathbf{z_i})}$, the risk bound of Theorem 1 specializes to

$$\begin{aligned} \varepsilon(\sigma, \mathbf{Z_i}, |\mathbf{j}|, \delta) &= 1 - \exp\left(\frac{-1}{m - |\mathbf{i}| - |\mathbf{j}|}\left[\ln\binom{m}{|\mathbf{i}|} + \ln\binom{m-|\mathbf{i}|}{|\mathbf{j}|} + \ln\binom{p(\mathbf{z_i})}{b(\sigma)} + \right.\right. \\ &\qquad \left.\left. \ln\left(\frac{1}{\zeta(|\mathbf{i}|)\zeta(|\mathbf{j}|)\zeta(b(\sigma))\delta}\right)\right]\right). \end{aligned} \tag{16}$$

In contrast, the SCM risk bound of Marchand and Shawe-Taylor (2001) is equal to

$$\begin{aligned} \varepsilon'(\sigma, \mathbf{Z_i}, |\mathbf{j}|, \delta) &= 1 - \exp\left(\frac{-1}{m - |\mathbf{i}| - |\mathbf{j}|}\left[\ln\binom{m}{|\mathbf{i}|} + \ln\binom{m-|\mathbf{i}|}{|\mathbf{j}|} + \right.\right. \\ &\qquad \left.\left. \ln\binom{c_p(\mathbf{z_i}) + c_n(\mathbf{z_i}) + b(\mathbf{z_i})}{c_p(\mathbf{z_i})} + \ln\left(\frac{m^2|\mathbf{i}|}{\delta}\right)\right]\right), \end{aligned} \tag{17}$$

where $c_p(\mathbf{z_i})$ and $c_n(\mathbf{z_i})$ denote, respectively, the number of positive centers and the number of negative centers in $\mathbf{z_i}$ and where $b(\mathbf{z_i})$ denotes the the number of borders in $\mathbf{z_i}$.

Hence, we observe only two differences between these two bounds. First, the (larger) $\ln(m^2|\mathbf{i}|/\delta)$ term of Marchand and Shawe-Taylor (2001) has been replaced by the (smaller) $\ln(1/\zeta(|\mathbf{i}|)\zeta(|\mathbf{j}|)\zeta(b(\sigma))\delta)$ term. Second, the coefficient

$$\binom{c_p(\mathbf{z_i})+c_n(\mathbf{z_i})+b(\mathbf{z_i})}{c_p(\mathbf{z_i})}$$

has been replaced by the smaller coefficient

$$\binom{p(\mathbf{z_i})}{b(\sigma)}.$$

We can verify that this last coefficient is indeed smaller since

$$\binom{p(\mathbf{z_i})}{b(\sigma)} = \binom{c_p(\mathbf{z_i})+b(\mathbf{z_i})}{b(\mathbf{z_i})} = \binom{c_p(\mathbf{z_i})+b(\mathbf{z_i})}{c_p(\mathbf{z_i})} \leq \binom{c_p(\mathbf{z_i})+c_n(\mathbf{z_i})+b(\mathbf{z_i})}{c_p(\mathbf{z_i})}.$$

Consequently, the risk bound of Theorem 1, applied to the SCM with the distribution given by Equation 15, is smaller than the SCM risk bound of Marchand and Shawe-Taylor (2001).

6. Risks Bounds for Decision List Machines

To apply the risk bound of Theorem 1, we need to define a distribution of message strings[2] $P_{\mathcal{M}(\mathbf{z_i})}(\sigma)$ for each type of DLM that we will consider. Once that distribution is known, we only need to insert it in Equation 10 to obtain the risk bound. Note that the risk bound does not depend on how we actually code σ (for some receiver, in a communication setting). It only depends on the *a priori* probabilities assigned to each possible realization of σ.

6.1 DLMs Containing Only Balls

Even in this simplest case, the compression set $\mathbf{z_i}$ alone does not contain enough information to identify a DLM classifier (the hypothesis). To identify unambiguously the hypothesis we need to provide also a message string σ.

Recall that, in this case, $\mathbf{z_i}$ contains ball centers and border points. By construction, each center is always correctly classified by the hypothesis. Moreover, each center can only be the center of one ball since the center is removed from the data when a ball is added to the DLM. But a center can be the border of a previous ball in the DLM and a border can be the border of more than one ball (since the border of a ball is not removed from the data when that ball is added to the DLM). Hence, σ needs to specify the border points in $\mathbf{z_i}$ that are a border without being the center of another ball. Let σ_1 be the part of the message string σ that will specify that information and let $P_1(\sigma_1)$ be the probabilities that we assign to each possible realization of σ_1. Since we expect that most of the compression sets will contain roughly the same number of centers and borders, we assign, to each example of $\mathbf{z_i}$, an equal *a priori* probability to be a center or a border. Hence we use

$$P_1(\sigma_1) = \frac{1}{2^{|\mathbf{i}|}} \quad \forall \sigma_1.$$

2. We will refer to $P_{\mathcal{M}(\mathbf{z_i})}$ as the “distribution” of messages even though its summation over the possible realizations of σ might be less than one (as specified by Equation 7).

Once σ_1 is specified, the centers and borders of $\mathbf{z_i}$ are identified. But to identify each ball we need to pair each center with a border point (which could possibly be the center of another ball). For this operation, recall that the border and the center of each ball must have opposite class labels. Let σ_2 be the part of the message string σ that specifies that pairing information and let $P_2(\sigma_2|\sigma_1)$ be the probabilities that we assign to each possible realization of σ_2 once σ_1 is given. Let $n(\mathbf{z_i})$ and $p(\mathbf{z_i})$ be, respectively, the number of negative and the number of positive examples in compression set $\mathbf{z_i}$. Consider now a positive center example $\mathbf{x}$ of $\mathbf{z_i}$. Since a border point can be used for more that one ball and a center can also be used as a border, we assign an equal probability of $1/n(\mathbf{z_i})$ to each negative example of $\mathbf{z_i}$ to be paired with $\mathbf{x}$. Similarly, we assign an equal probability of $1/p(\mathbf{z_i})$ to each positive example to be paired with a negative center of $\mathbf{z_i}$. Let $c_p(\mathbf{z_i})$ and $c_n(\mathbf{z_i})$ be, respectively, the number of positive centers and negative centers in $\mathbf{z_i}$ (this is known once σ_1 is specified). For an arbitrary compression set $\mathbf{z_i}$, we thus assign the following *a priori* probability to each possible pairing information string σ_2:

$$P_2(\sigma_2|\sigma_1) = \left(\frac{1}{n(\mathbf{z_i})}\right)^{c_p(\mathbf{z_i})} \left(\frac{1}{p(\mathbf{z_i})}\right)^{c_n(\mathbf{z_i})} \quad \forall \sigma_2 \mid n(\mathbf{z_i}) \neq 0 \text{ and } p(\mathbf{z_i}) \neq 0.$$

This probability is, indeed, correctly defined only under the condition that $n(\mathbf{z_i}) \neq 0$ and $p(\mathbf{z_i}) \neq 0$. However, since the border and center of each ball must have opposite labels, this condition is the same as $|\mathbf{i}| \neq 0$. When $|\mathbf{i}| = 0$, we can just assign 1 to $P_2(\sigma_2|\sigma_1)$. By using the indicator function $I(a)$ defined previously, we can thus write $P_2(\sigma_2|\sigma_1)$ more generally as

$$P_2(\sigma_2|\sigma_1) = \left(\frac{1}{n(\mathbf{z_i})}\right)^{c_p(\mathbf{z_i})} \left(\frac{1}{p(\mathbf{z_i})}\right)^{c_n(\mathbf{z_i})} I(|\mathbf{i}| \neq 0) \, + \, I(|\mathbf{i}| = 0) \quad \forall \sigma_2.$$

Once σ_1 and σ_2 are known, each ball of the DLM is known. However, to place these balls in the DLM, we need to specify their order. Let $r(\mathbf{z_i}) \overset{\text{def}}{=} c_p(\mathbf{z_i}) + c_n(\mathbf{z_i})$ be the number of balls in the DLM (this is known once σ_1 and σ_2 are specified). Let σ_3 be the part of the message string σ that specifies this ordering information and let $P_3(\sigma_3|\sigma_2,\sigma_1)$ be the probabilities that we assign to each possible realization of σ_3 once σ_1 and σ_2 are given. For an arbitrary compression set $\mathbf{z_i}$, we assign an equal *a priori* probability to each possible ball ordering by using

$$P_3(\sigma_3|\sigma_2,\sigma_1) = \frac{1}{r(\mathbf{z_i})!} \quad \forall \sigma_3.$$

The distribution of messages is then given by $P_1(\sigma_1)P_2(\sigma_2|\sigma_1)P_3(\sigma_3|\sigma_2,\sigma_1)$. Hence

$$P_{\mathcal{M}(\mathbf{z_i})}(\sigma) = \frac{1}{2^{|\mathbf{i}|}} \cdot \left[\left(\frac{1}{n(\mathbf{z_i})}\right)^{c_p(\mathbf{z_i})} \left(\frac{1}{p(\mathbf{z_i})}\right)^{c_n(\mathbf{z_i})} I(|\mathbf{i}| \neq 0) \, + \, I(|\mathbf{i}| = 0) \right] \cdot \frac{1}{r(\mathbf{z_i})!} \quad \forall \sigma. \qquad (18)$$

6.2 DLMs Containing Balls and Holes

The use of holes in addition to balls introduces a few more difficulties that are taken into account by sending a few more bits of information to the reconstruction function. The most important change is that the center of a hole can be used more than once since the covered examples are outside the hole. Hence, the number of features can now exceed the number of centers but it is always smaller than $|\mathbf{i}|^2$. Indeed, in the worst case, each pair of (distinct) examples taken from the compression set $\mathbf{z_i}$ could be used for two holes: giving a total of $|\mathbf{i}|(|\mathbf{i}|-1)$ features. The first part σ_1 of the (complete) message string σ will specify the number $r(\mathbf{z_i})$ of features present in compression set $\mathbf{z_i}$. Since we always have $r(\mathbf{z_i}) < |\mathbf{i}|^2$ for $|\mathbf{i}| > 0$, we could give equal *a priori* probability for each value of $r \in \{0, \ldots, |\mathbf{i}|^2\}$. However since we want to give a slight preference to smaller DLMs, we choose to assign a probability equal to $\zeta(r)$ (defined by Equation 9) for all possible values of r. Hence

$$P_1(\sigma_1) = \zeta(r(\mathbf{z_i})) \quad \forall \sigma_1 .$$

The second part σ_2 of σ specifies, for each feature, if the feature is a ball or a hole. For this, we give equal probability to each of the $r(\mathbf{z_i})$ features to be a ball or a hole. Hence

$$P_2(\sigma_2|\sigma_1) = 2^{-r(\mathbf{z_i})} \quad \forall \sigma_2 .$$

Finally, the third part σ_3 of σ specifies, sequentially for each feature, the center and border point. For this, we give an equal probability of $1/|\mathbf{i}|$ to each example in $\mathbf{z_i}$ of being chosen (whenever $|\mathbf{i}| \neq 0$). Consequently

$$P_3(\sigma_3|\sigma_2,\sigma_1) = |\mathbf{i}|^{-2r(\mathbf{z_i})} I(|\mathbf{i}| \neq 0) + I(|\mathbf{i}| = 0) \quad \forall \sigma_3 .$$

The distribution of messages is then given by $P_1(\sigma_1)P_2(\sigma_2|\sigma_1)P_3(\sigma_3|\sigma_2,\sigma_1)$. Hence

$$P_{\mathcal{M}(\mathbf{z_i})}(\sigma) = \zeta(r(\mathbf{z_i})) \cdot 2^{-r(\mathbf{z_i})} \cdot \left[|\mathbf{i}|^{-2r(\mathbf{z_i})} I(|\mathbf{i}| \neq 0) + I(|\mathbf{i}| = 0) \right] \quad \forall \sigma . \tag{19}$$

6.3 Constrained DLMs Containing Only Balls

A *constrained* DLM is a DLM that has the property of correctly classifying each example of its compression set $\mathbf{z_i}$ with the exception of the compression set examples who's output is determined by the default value. This implies that **BuildDLM** must be modified to ensure that this constraint is satisfied. This is achieved by considering, at each greedy step, only the features h_i with an output b_i and covering set Q_i that satisfy the following property. Every training example $(\mathbf{x}, y) \in Q_i$ that is either a border point of a previous feature (ball or hole) in the DLM or a center of a previous hole in the DLM must have $y = b_i$ and thus be correctly classified by h_i.

We will see that this constraint will enable us to provide less information to the reconstruction function (to identify a classifier) and will thus yield tighter risk bounds. However, this constraint might, in turn, force **BuildDLM** to produce classifiers containing more features. Hence, we do not know *a priori* which version will produce classifiers having a smaller risk.

Let us first describe the simpler case where only balls are permitted.

As before, we use a string σ_1, with the same probability $P_1(\sigma_1) = 2^{-|\mathbf{i}|}$ $\forall \sigma_1$ to specify if each example of the compression set $\mathbf{z_i}$ is a center or a border point. This gives us the set of centers which coincides with the set of balls since each center can only be used once for this type of DLM.

Next we use a string σ_2 to specify the ordering of each center (or ball) in the DLM. As before we assign equal *a priori* probability to each possible ordering. Hence $P_2(\sigma_2|\sigma_1) = 1/r(\mathbf{z_i})!$ $\forall \sigma_2$ where $r(\mathbf{z_i})$ denotes the number of balls for $\mathbf{z_i}$ (an information given by σ_1).

But now, since each feature was constrained to correctly classify the examples of $\mathbf{z_i}$ that it covers (and which were not covered by the features above), we do not need any additional information to specify the border for each center. Indeed, for this task we use the following algorithm. Given a compression set $\mathbf{z_i}$, let P and N denote, respectively, the set of positive and the set of negative examples in $\mathbf{z_i}$. We start with $P' = P, N' = N$ and do the following, sequentially, from the first center (or ball) to the last. If center $\mathbf{c}$ is positive, then its border $\mathbf{b}$ is given by $\text{argmin}_{\mathbf{x}\in N'} d(\mathbf{c},\mathbf{x})$ and we remove from P' (to find the border of the other balls) the center $\mathbf{c}$ and all other positive examples covered by that feature and used by the previous features. If center $\mathbf{c}$ is negative, then its border $\mathbf{b}$ is given by $\text{argmin}_{\mathbf{x}\in P'} d(\mathbf{c},\mathbf{x})$ and we remove from N' the center $\mathbf{c}$ and all other negative examples covered by that feature and used by the previous features.

The distribution of messages is then given by

$$P_{\mathcal{M}(\mathbf{z_i})}(\sigma) = \frac{1}{2^{|\mathbf{i}|}} \cdot \frac{1}{r(\mathbf{z_i})!} \quad \forall\sigma. \tag{20}$$

6.4 Constrained DLMs Containing Balls and Holes

As for the case of Section 6.2, we use a string σ_1 to specify the number $r(\mathbf{z_i})$ of features present in compression set $\mathbf{z_i}$. We also use a string σ_2 to specify, for each feature, if the feature is a ball or a hole. The probabilities $P_1(\sigma_1)$ and $P_2(\sigma_2|\sigma_1)$ used are the same as those defined in Section 6.2. Here, however, we only need to specify the center of each feature, since, as we will see below, no additional information is needed to find the border of each feature when the DLM is constrained to classify correctly each example in $\mathbf{z_i}$. Consequently

$$P_{\mathcal{M}(\mathbf{z_i})}(\sigma) = \zeta(r(\mathbf{z_i})) \cdot 2^{-r(\mathbf{z_i})} \cdot \left[|\mathbf{i}|^{-r(\mathbf{z_i})} I(|\mathbf{i}| \neq 0) + I(|\mathbf{i}| = 0)\right] \quad \forall\sigma. \tag{21}$$

To specify the border of each feature, we use the following algorithm. Given a compression set $\mathbf{z_i}$, let P and N denote, respectively, the set of positive and the set of negative examples in $\mathbf{z_i}$. We start with $P' = P, N' = N$ and do the following, sequentially, from the first feature to the last. If the feature is a ball with a positive center $\mathbf{c}$, then its border is given by $\text{argmin}_{\mathbf{x}\in N'} d(\mathbf{c},\mathbf{x})$ and we remove from P' the center $\mathbf{c}$ and all other positive examples covered by that feature and used by the previous features. If the feature is a hole with a positive center $\mathbf{c}$, then its border is given by $\text{argmax}_{\mathbf{x}\in P'-\{\mathbf{c}\}} d(\mathbf{c},\mathbf{x})$ and we remove from N' all the negative examples covered by that feature and used by the previous features. If the feature is a ball with a negative center $\mathbf{c}$, then its border is given by $\text{argmin}_{\mathbf{x}\in P'} d(\mathbf{c},\mathbf{x})$ and we remove from N' the center $\mathbf{c}$ and all other negative examples covered by that feature and used by the previous features. If the feature is a hole with a negative center $\mathbf{c}$, then its border is given by $\text{argmax}_{\mathbf{x}\in N'-\{\mathbf{c}\}} d(\mathbf{c},\mathbf{x})$ and we remove from P' all the positive examples covered by that feature and used by the previous features.

6.5 Constrained DLMs with Half-Spaces

Recall that each half-space is specified by weight vector $\mathbf{w}$ and a threshold value t. The weight vector is identified by a pair $(\mathbf{x}_a, \mathbf{x}_b)$ of examples having opposite class labels and the threshold is specified by a third example $\mathbf{x}_c$ of the same class label as example $\mathbf{x}_a$.

The first part of the message will be a string σ_1 that specifies the number $r(\mathbf{z_i})$ of half-spaces used in the compression set $\mathbf{z_i}$. As before, let $p(\mathbf{z_i})$ and $n(\mathbf{z_i})$ denote, respectively, the number of positive examples and the number of negative examples in the compression set $\mathbf{z_i}$. Let $P(\mathbf{z_i})$ and $N(\mathbf{z_i})$ denote, respectively, the set of positive examples and the set of negative examples in the compression set $\mathbf{z_i}$. From these definitions, each pair $(\mathbf{x}_a, \mathbf{x}_b) \in P(\mathbf{z_i}) \times N(\mathbf{z_i}) \cup N(\mathbf{z_i}) \times P(\mathbf{z_i})$ can provide one weight vector. Moreover, since a half-space may not cover any point used for its construction, each weight vector may be used for several half-spaces in the DLM. But half-spaces having the same weight vector $\mathbf{w}$ must have a different threshold since, otherwise, they would cover the same set of examples. Hence the total number of half-spaces in the DLM is at most $|\mathbf{i}|p(\mathbf{z_i})n(\mathbf{z_i})$. Therefore, for the string σ_1 that specifies the number $r(\mathbf{z_i})$ of half-spaces used in the compression set $\mathbf{z_i}$, we could assign the same probability to each number between zero and $|\mathbf{i}|p(\mathbf{z_i})n(\mathbf{z_i})$. However, as before, we want to give preference to DLMs having a smaller number of half-spaces. Hence we choose to assign a probability equal to $\zeta(r)$ (defined by Equation 9) for all possible values of r. Therefore

$$P_1(\sigma_1) = \zeta(r(\mathbf{z_i})) \quad \forall \sigma_1 .$$

Next, the second part σ_2 of σ specifies, sequentially for each half-space, the pair $(\mathbf{x}_a, \mathbf{x}_b) \in P(\mathbf{z_i}) \times N(\mathbf{z_i}) \cup N(\mathbf{z_i}) \times P(\mathbf{z_i})$ used for its weight vector $\mathbf{w}$. For this we assign an equal probability of $1/2p(\mathbf{z_i})n(\mathbf{z_i})$ for each possible $\mathbf{w}$ of each half-space. Hence

$$P_2(\sigma_2|\sigma_1) = \left(\frac{1}{2p(\mathbf{z_i})n(\mathbf{z_i})}\right)^{r(\mathbf{z_i})} \quad \forall \sigma_2 \mid n(\mathbf{z_i}) \neq 0 \text{ and } p(\mathbf{z_i}) \neq 0.$$

The condition that $n(\mathbf{z_i}) \neq 0$ and $p(\mathbf{z_i}) \neq 0$ is equivalent to $|\mathbf{i}| \neq 0$ since, for any half-space, $\mathbf{x}_a$ and $\mathbf{x}_b$ must have opposite labels. Hence, more generally, we have

$$P_2(\sigma_2|\sigma_1) = \left(\frac{1}{2p(\mathbf{z_i})n(\mathbf{z_i})}\right)^{r(\mathbf{z_i})} I(|\mathbf{i}| \neq 0) \; + \; I(|\mathbf{i}| = 0) \quad \forall \sigma_2 .$$

Finally, as for the other constrained DLMs, we do not need any additional message string to identify the threshold point $\mathbf{x}_c \in \mathbf{z_i}$ for each $\mathbf{w}$ of the DLM. Indeed, for this task we can perform the following algorithm. Let P and N denote, respectively, the set of positive and the set of negative examples in $\mathbf{z_i}$. We start with $P' = P, N' = N$ and do the following, sequentially, from the first half-space to the last. Let $\mathbf{w} = \boldsymbol{\phi}(\mathbf{x}_a) - \boldsymbol{\phi}(\mathbf{x}_b)$ be the weight vector of the current half-space. If $\mathbf{x}_a \in P$ then, for the threshold point $\mathbf{x}_c$, we choose $\mathbf{x}_c = \underset{\mathbf{x} \in N'}{\operatorname{argmax}}\, \mathbf{w} \cdot \mathbf{x}$ and we remove from P' the positive examples covered by this half-space and used by the previous half-spaces. Else if $\mathbf{x}_a \in N$ then, for the threshold point $\mathbf{x}_c$, we choose $\mathbf{x}_c = \underset{\mathbf{x} \in P'}{\operatorname{argmax}}\, \mathbf{w} \cdot \mathbf{x}$ and we remove from N' the negative examples covered by this half-space and used by the previous half-spaces.

Consequently, the distribution of message strings is given by

$$P_{\mathcal{M}(\mathbf{z_i})}(\sigma) = \zeta(r(\mathbf{z_i})) \cdot \left[\left(\frac{1}{2p(\mathbf{z_i})n(\mathbf{z_i})}\right)^{r(\mathbf{z_i})} I(|\mathbf{i}| \neq 0) \; + \; I(|\mathbf{i}| = 0) \right] \quad \forall \sigma . \tag{22}$$

6.6 Unconstrained DLMs with Half-Spaces

As for the case of Section 6.5, we use a string σ_1 to specify the number $r(\mathbf{z_i})$ of half-spaces present in compression set $\mathbf{z_i}$. We also use a string σ_2 to specify, sequentially for each half-space, the pair $(\mathbf{x}_a, \mathbf{x}_b) \in P(\mathbf{z_i}) \times N(\mathbf{z_i}) \cup N(\mathbf{z_i}) \times P(\mathbf{z_i})$ used for its weight vector $\mathbf{w}$. Hence, the probabilities $P_1(\sigma_1)$ and $P_2(\sigma_2|\sigma_1)$ used are the same as those defined in Section 6.5. But here, in addition, we need to specify the threshold point $\mathbf{x}_c$ for each $\mathbf{w}$. For this, we give an equal probability of $1/|\mathbf{i}|$ to each example in $\mathbf{z_i}$ of being chosen (when $|\mathbf{i}| \neq 0$). Consequently, the distribution of messages is given by

$$P_{\mathcal{M}(\mathbf{z_i})}(\sigma) = \zeta(r(\mathbf{z_i})) \cdot \left[\left(\frac{1}{2p(\mathbf{z_i})n(\mathbf{z_i})}\right)^{r(\mathbf{z_i})} |\mathbf{i}|^{-r(\mathbf{z_i})} I(|\mathbf{i}| \neq 0) \; + \; I(|\mathbf{i}| = 0) \right] \quad \forall \sigma . \tag{23}$$

7. Empirical Results on Natural Data

We have tested the DLM on several "natural" data sets which were obtained from the machine learning repository at UCI. For each data set, we have removed all examples that contained attributes with unknown values 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 3. 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).

Table 1 and Table 2 show the DLM sizes s and penalty values that gave the smallest 10-fold cross-validation error separately for the following types of DLMs that we have studied in Section 6:

DLM$_b$**:** unconstrained DLMs with balls (only).

DLM$_b^*$**:** constrained DLMs with balls (only).

DLM$_{bh}$**:** unconstrained DLMs with balls and holes.

DLM$_{bh}^*$**:** constrained DLMs with balls and holes.

DLM$_{hsp}$**:** unconstrained DLMs with half-spaces.

DLM$_{hsp}^*$**:** constrained DLMs with half-spaces.

For each of these DLMs, the learning algorithm used was **BuildDLM**. We have observed that **PruneDLM** had no effect on all these data sets, except for Credit where it was sometimes able to remove one feature.

In Table 3, we have compared the performance of the DLM with the set covering machine (SCM) using the same sets of data-dependent features, and the support vector machine (SVM) equipped with a radial basis function kernel of variance $1/\gamma$ and a soft-margin parameter C.

We have used the L_2 metric for the data-dependent features for both DLMs and SCMs to obtain a fair comparison with SVMs. Indeed, the argument of the radial basis function kernel is given by the L_2 metric between two input vectors. For the SVM, the values of s refer to the average number of support vectors obtained from the 10 different training sets of 10-fold cross-validation. For the SCM, the value of T indicates the type of features it used and whether the SCM was a conjunction (c) or a disjunction (d). The values of p and s for the SCM refer to the penalty value and the number of features that gave the smallest 10-fold cross-validation error. We emphasize that, for all learning machines, the values of the learning parameters reported in Tables 1, 2, and 3 are the ones that gave the smallest 10-fold cross-validation error when chosen among a very large list of values. Although this overestimates the performance of every learning algorithm, it was used here to compare equally fairly (or equally unfairly) every learning machine. We will report below the results for DLMs when the testing sets are not used to determine the best values of the learning parameters.

In addition to our estimate of the generalization error, we have also reported in Table 3, 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

Data Set	DLM_b		DLM_b^*		DLM_{bh}		DLM_{bh}^*	
	$\{p_p, p_n, s\}$	err	$\{p_p, p_n, s\}$	err	$\{p_p, p_n, s\}$	err	$\{p_p, p_n, s\}$	err
BreastW	0.7, 2.1, 6	18	0.7, 0, 1	18	2, 1, 2	13	1.6, 1, 2	13
Bupa	1.5, 3.7, 14	104	2.5, 1.5, 21	107	2, 2.1, 4	110	2.5, 1.5, 17	107
Credit	1.7, 2.1, 6	188	0.7, 0.3, 42	195	2.1, 1.4, 11	187	1.3, ∞, 33	195
Glass	2.5, ∞, 8	33	2.5, 3.5, 8	32	4, 4.5, 7	29	∞, 3.7, 7	29
Haberman	∞, 4.5, 23	74	0.5, 4, 3	70	3.7, 3.4, 12	64	1.7, 3.7, 6	65
Heart	1.7, 2.7, 17	94	1.6, 2.3, 8	89	1.5, 2.6, 10	95	2, 2, 9	101
Pima	1.5, 2.2, 5	184	2.5, 2.6, 74	184	1, 1.5, 2	190	1, 1.5, 6	189
Votes	2.5, 1.5, 6	34	2, ∞, 14	36	4.5, ∞, 14	35	1.8, ∞, 23	37

Table 1: Optimal 10-fold cross-validation results for DLMs with balls (and holes).

Data Set	DLM_{hsp}		DLM_{hsp}^*	
	$\{p_p, p_n, s\}$	err	$\{p_p, p_n, s\}$	err
BreastW	1, ∞, 1	18	1.7,∞, 1	20
Bupa	2.7,1.5,15	107	0.9,2,6	102
Credit	3.5,2.5,26	141	1.9,1.5,7	151
Glass	2.1,0.7,4	35	2,1.3,4	37
Haberman	1.8,3,5	66	1.5,1.1,2	70
Heart	0.8,1,1	85	0.8,0.5,3	83
Pima	1.7,1, 4	169	1.9,2,8	183
Votes	4.4, ∞, 13	24	3.3, ∞, 13	33

Table 2: Optimal 10-fold cross-validation results for DLMs with half-spaces.

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.

In terms of generalization error, there is no substantial difference among all the types of DLMs presented in Tables 1 and 2 for most of the data sets—except for Credit where DLMs with half-spaces have a significantly lower error rate.

From the results in Tables 1 and 2, we notice that the effect of constraining the DLM to correctly classify the compression set[3] generally increases the size of the DLM. The increase is substantial for the Credit and Pima data sets (except for half-spaces where the opposite behavior is observed). It is surprising that a DLM_b^* with 74 balls has the same error rate as a DLM_b with 5 balls on the Pima data set. In contrast, constraining an SCM to correctly classify the compression set had virtually no effect on the size of the classifier (for these data sets).

The most striking feature in all the results is the level of sparsity achieved by the SCM and the DLM in comparison with the SVM. This difference is always huge. The other important feature is that DLMs often produce slightly better generalization than SCMs and SVMs. Hence, DLMs can provide a good alternative to SCMs and SVMs.

Recall that the results reported in Tables 1, 2, and 3 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_p, p_n and s. This model-selection strategy is, in that sense, optimal (but not realizable). Hence, we refer to the scores obtained in theses Tables as those obtained by the

3. Recall from Section 6 that this was done to obtain a tighter risk bound.

Data Set		SVM		SCM			DLM		
Name	exs	$\{\gamma, C, s\}$	err	T	$\{p, s\}$	err$\pm\sigma$	T	$\{p_p, p_n, s\}$	err$\pm\sigma$
BreastW	683	0.005, 2, 58	19	b*,c	1.8, 2	15$\pm$3.9	bh*	1.6, 1, 2	**13**$\pm$3.4
Bupa	345	0.002, 0.2, 266	107	hsp*,c	1.4,1	103 $\pm$ 6.2	hsp*	0.9,2,6	**102** $\pm$7.8
Credit	653	0.0006, 32, 423	190	hsp*,d	1.2, 3	148$\pm$10.2	hsp	3.5,2.5,26	**141**$\pm$13.5
Glass	214	0.8, 1.2, 130	34	b*,d	∞, 3	36$\pm$6.3	bh*	∞, 3.7, 7	**29**$\pm$6.7
Haberman	294	0.01, 0.6, 146	71	hsp*,d	0.7,1	68$\pm$ 5.9	bh	3.7, 3.4, 12	**64**$\pm$3.9
Heart	297	0.001, 2, 204	87	hsp*,d	1.3,1	87$\pm$7.1	hsp*	0.8,0.5,3	**83**$\pm$6.9
Pima	768	0.002, 1, 526	203	hsp*,c	1.5, 3	175$\pm$4.9	b	1.7,1,4	**169**$\pm$5
Votes	435	0.2, 1.7, 125	**22**	hsp*, d	∞,13	34$\pm$5.8	hsp	4.4,∞,13	24$\pm$5.2

Table 3: Optimal 10-fold cross-validation results for SVMs, SCMs, and DLMs.

"optimal" model-selection strategy. To investigate the extent to which a bound can perform model selection, we use the proposed risk bound to select a DLM among those obtained for a list of at least 1000 pairs of penalty values (which always included the optimal pair of penalty values) and for all possible sizes s. We have compared these results with the K-fold cross-validation model selection method. This latter method, widely used in practice, consists of using K-fold cross-validation to find the best stopping point s and the best penalty values p_p and p_n (among the same list of values used for the previous model selection method) on a given training set and then use these best values on the full training set to find the best DLM. Both model selection methods were tested with K-fold cross-validation. The results are reported in Tables 4, 5 and 6 for all the types of DLMs that we have considered in Section 6. In these tables, "MSfromCV" stands for "model selection from cross-validation" and "MSfromBound" stands for "model selection from bound". For all these results we have used $K = 10$, except for DLM$^*_{hsp}$ where we have used $K = 5$. Except for a few cases, we see that model selection by using the bounds of Section 6 is generally slightly more effective than using K-fold cross validation (and takes substantially less computation time).

Data Set	DLM$_b$				DLM$^*_b$			
	MSfromCV		MSfromBound		MSfromCV		MSfromBound	
	s	err$\pm\sigma$	s	err$\pm\sigma$	s	err$\pm\sigma$	s	err$\pm\sigma$
BreastW	1.8	18$\pm$5.01	2	18$\pm$4.4	1	20 $\pm$4.9	10	20$\pm$6.4
Bupa	15.3	123$\pm$8.5	15.3	112$\pm$5.6	15.3	115$\pm$7.6	27	117$\pm$7.3
Credit	25.7	231$\pm$ 12.3	20.2	194$\pm$5.6	3.5	203 $\pm$13.2	6.8	215$\pm$20.2
Glass	7.2	44$\pm$ 4.1	15.9	36$\pm$6.5	8.1	44$\pm$7.9	14.1	34$\pm$6.1
Haberman	6.3	80$\pm$9.8	37	89$\pm$15.1	3.8	89 $\pm$13.9	13.4	104$\pm$7.8
Heart	8.7	104$\pm$8.5	20.6	97$\pm$6	10.8	103$\pm$8.5	30.6	95$\pm$8.7
Pima	25.5	213$\pm$10.5	13.7	198$\pm$10.2	28.2	230$\pm$10.4	50	192$\pm$9.3
Votes	13.4	48$\pm$6.3	7.2	38$\pm$8.6	13.4	48$\pm$6.3	16.5	40$\pm$7.3

Table 4: Model selection results for DLMs with balls (only).

8. Conclusion and Open Problems

We have introduced a new learning algorithm for decision lists and have shown that it can provide a favorable alternative to the SCM on some "natural" data sets. Compared with SVMs, the proposed

Data Set	DLM_{bh}				DLM^*_{bh}			
	MSfromCV		MSfromBound		MSfromCV		MSfromBound	
	s	err$\pm\sigma$	s	err$\pm\sigma$	s	err$\pm\sigma$	s	err$\pm\sigma$
BreastW	1.8	15$\pm$3.7	2.2	14$\pm$4.1	1.8	18$\pm$4.9	2.3	16$\pm$4.9
Bupa	8.6	124$\pm$7.9	12.9	119$\pm$7.7	15.2	123$\pm$8.4	23.7	118$\pm$8.2
Credit	6.3	209$\pm$14.7	17.6	206$\pm$11.5	25.7	231$\pm$12.1	40.9	208$\pm$9.5
Glass	7.4	37$\pm$8.4	13.4	35$\pm$6.4	7.2	44$\pm$4.6	21.7	30$\pm$6
Haberman	6	74$\pm$3	23.6	68$\pm$4.4	6.3	80$\pm$9.6	20.1	65$\pm$6.1
Heart	9.1	112$\pm$8.2	14.9	104$\pm$6.6	10.9	103$\pm$10.5	24.1	105$\pm$6.4
Pima	2.8	204$\pm$8.9	4	212$\pm$9.5	7.8	212$\pm$8.7	11.1	203$\pm$11.2
Votes	13.5	44$\pm$6.8	19.5	35$\pm$4.8	11.8	48$\pm$5.4	14.5	40$\pm$7.4

Table 5: Model selection results for DLMs with balls and holes.

Data Set	DLM_{hsp}				DLM^*_{hsp}			
	MSfromCV		MSfromBound		MSfromCV		MSfromBound	
	s	err$\pm\sigma$	s	err$\pm\sigma$	s	err$\pm\sigma$	s	err$\pm\sigma$
BreastW	1.6	22$\pm$4.6	7	18$\pm$6	8.2	20$\pm$2.6	9	18 $\pm$ 6.9
Bupa	3.1	117$\pm$2.9	11.8	117$\pm$6.2	3.2	110$\pm$3.7	5.2	117 $\pm$ 9.3
Credit	17.7	152$\pm$13.9	28.3	152 $\pm$17.8	6.4	165$\pm$9.2	7.6	163 $\pm$ 10
Glass	2.2	39$\pm$6	2.9	34$\pm$6.3	2.8	38$\pm$4.7	3.4	34 $\pm$ 5.9
Haberman	5.8	78$\pm$5.8	11.2	68$\pm$8.9	2.4	69$\pm$2.3	4	68 $\pm$ 3.4
Heart	1.6	87$\pm$ 8	3.6	89$\pm$8.2	1.6	99$\pm$3.5	3	120 $\pm$ 2.2
Pima	2.7	178$\pm$11.2	4.3	182 $\pm$12.5	10.5	187$\pm$6.7	15.8	175 $\pm$ 12.7
Votes	9.5	32$\pm$5.4	16.3	26$\pm$5.4	5.6	38$\pm$3.3	13.8	33 $\pm$ 4.3

Table 6: Model selection results for DLMs with half-spaces.

learning algorithm for DLMs produces substantially sparser classifiers with comparable, and often better, generalization.

We have proposed a general risk bound that depends on the number of examples that are used in the final classifier and the size of the information message needed to identify the final classifier from the compression set. The proposed bound is significantly tighter than the one provided by Littlestone and Warmuth (1986) and Floyd and Warmuth (1995) and applies to any compression set-dependent distribution of messages. We have applied this general risk bound to DLMs by making appropriate choices for the compression set-dependent distribution of messages and have shown, on natural data sets, that these specialized risk bounds are generally slightly more effective than K-fold cross validation for selecting a good DLM model.

The next important step is to find risk bounds that hold for asymmetrical loss functions. Indeed, this is the type of loss function which is most appropriate for many natural data sets and we cannot use, in these circumstances, the risk bounds proposed here since they are valid only for the symmetric loss case. Other important issues are the investigation of other metrics and other data-dependent sets of features.

This paper shows that it is sometimes worthwhile to use a decision list of data-dependent features instead of a conjunction or a disjunction of the same set of features. Hence, we may ask if it is worthwhile to consider the larger class of linear threshold functions. With data-dependent features, we want to use (or adapt) algorithms that are efficient when irrelevant features abound. In these cases, the winnow (Littlestone, 1988) and the multi-layered winnow (Nevo and El-Yaniv,

2002) algorithms are obvious candidates. However, these algorithms do not return a sparse solution since many features will be assigned a non-negligible weight value. Moreover, our (preliminary) numerical experiments with the winnow algorithm indicate that this algorithm is simply too slow to be used with $O(m^2)$ features for $m \geq 700$. More generally, we think that this research direction is not attractive in view of the fact that it is (generally) very hard to find a linear threshold function with few non-zero weight values.

Acknowledgments

Work supported by NSERC Discovery grant 122405 - 2002 and NSERC doctoral scholarship 266896 - 2003. We thank John Shawe-Taylor for his comments concerning the proof of Theorem 1. We thank two of the reviewers for many helpful comments.

References

Martin Anthony. Generalization error bounds for threshold decision lists. *Journal of Machine Learning Research*, 5:189–217, 2004.

Avrim Blum and Mona Singh. Learning functions of k terms. In *COLT '90: Proceedings of the third annual workshop on Computational learning theory*, pages 144–153. Morgan Kaufmann Publishers Inc., 1990. ISBN 1-55860-146-5.

Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Occam's razor. *Information Processing Letters*, 24:377–380, 1987.

Aditi Dhagat and Lisa Hellerstein. PAC learning with irrelevant attributes. In *Proc. of the 35rd Annual Symposium on Foundations of Computer Science*, pages 64–74. IEEE Computer Society Press, Los Alamitos, CA, 1994.

Andrzej Ehrenfeucht and David Haussler. Learning decision trees from random examples. *Information and Computation*, 82:231–246, 1989.

Thomas Eiter, Toshihide Ibaraki, and Kazuhiso Makino. Decision lists and related boolean functions. *Theoretical Computer Science*, 270:493–524, 2002.

Sally Floyd and Manfred K. Warmuth. Sample compression, learnability, and the Vapnik-Chervonenkis dimension. *Machine Learning*, 21(3):269–304, 1995.

Thore Graepel, Ralf Herbrich, and John Shawe-Taylor. Generalisation error bounds for sparse linear classifiers. In *Proceedings of the Thirteenth Annual Conference on Computational Learning Theory*, pages 298–303, 2000.

Thore Graepel, Ralf Herbrich, and Robert C. Williamson. From margin to sparsity. In *Advances in neural information processing systems*, volume 13, pages 210–216, 2001.

Ralf Herbrich. *Learning Kernel Classifiers*. MIT Press, Cambridge, Massachusetts, 2002.

Geoffrey Hinton and Michael Revow. Using pairs of data-points to define splits for decision trees. *Advances in Neural Information Processing Systems 8*, pages 507–513, 1996.

Jyrki Kivinen, Heikki Mannila, and Esko Ukkonen. Learning hierarchical rule sets. In *Proceedings of the fifth annual ACM workshop on Computational Learning Theory*, pages 37–44. Morgan Kaufmann, 1992.

Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, 1988.

Nick Littlestone and Manfred K. Warmuth. Relating data compression and learnability. Technical report, University of California Santa Cruz, 1986.

Mario Marchand and Mostefa Golea. On learning simple neural concepts: from halfspace intersections to neural decision lists. *Network: Computation in Neural Systems*, 4:67–85, 1993.

Mario Marchand, Mohak Shah, John Shawe-Taylor, and Marina Sokolova. The set covering machine with data-dependent half-spaces. In *Proceedings of the Twentieth International Conference on Machine Learning(ICML'2003)*, pages 520–527. Morgan Kaufmann, 2003.

Mario Marchand and John Shawe-Taylor. Learning with the set covering machine. *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*, pages 345–352, 2001.

Mario Marchand and John Shawe-Taylor. The set covering machine. *Journal of Machine Learning Reasearch*, 3:723–746, 2002.

Shahar Mendelson. Rademacher averages and phase transitions in Glivenko-Cantelli class. *IEEE Transactions on Information Theory*, 48:251–263, 2002.

Ziv Nevo and Ran El-Yaniv. On online learning of decision lists. *Journal of Machine Learning Reasearch*, 3:271–301, 2002.

Ronald L. Rivest. Learning decision lists. *Machine Learning*, 2:229–246, 1987.

Marina Sokolova, Mario Marchand, Nathalie Japkowicz, and John Shawe-Taylor. The decision list machine. In *Advances in Neural Information Processing Systems(NIPS'2002)*, volume 15, pages 921–928. The MIT Press, 2003.

Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley, New-York, NY, 1998.