Training text classifiers with SVM on very few positive examples

Janez Brank
janez.brank@ijs.si

Marko Grobelnik
marko.grobelnik@ijs.si

Nataša Milić-Frayling
natasamf@microsoft.com

Dunja Mladenić
dunja.mladenic@ijs.si

April 2003

Technical Report
MSR-TR-2003-34

Text categorization is the problem of automatically assigning text documents into one or more categories. Typically, an amount of labelled data, positive and negative examples for a category, is available for training automatic classifiers. We are particularly concerned with text classification when the training data is highly imbalanced, i.e., the number of positive examples is very small. We show that the linear support vector machine (SVM) learning algorithm is adversely affected by imbalance in the training data. While the resulting hyperplane has a reasonable orientation, the proposed score threshold (parameter b) is too conservative. In our experiments we demonstrate that the SVM-specific cost-learning approach is not effective in dealing with imbalanced classes. We obtained better results with methods that directly modify the score threshold. We propose a method based on the conditional class distributions for SVM scores that works well when very few training examples is available to the learner.

Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
http://www.research.microsoft.com

1 Introduction

Text categorization involves a predefined set of categories and a set of documents that need to be classified using that categorization scheme. Each document can be assigned one or multiple categories (or perhaps none at all). We address the multi-class categorization problem as a set of binary problems where, for each category, the set of positive examples consists of documents belonging to the category while all other documents are considered negative examples. Labelled documents are used as input to various learning algorithms to train classifiers and automatically categorize new unlabelled documents.

Traditionally, machine learning research has assumed that the class distribution in the training data is reasonably balanced. More recently it has been recognized that this is often not the case with realistic data sets where many more negative examples than positive ones are available. The question then arises how best to utilize the available labelled data.

It has been observed that a disproportional abundance of negative examples decreases the performance of learning algorithms such as naive Bayes and decision trees [KM97]. Thus, research has been conducted into balancing the training set by duplicating positive examples (oversampling) or discarding negative ones (downsizing) [Jap00]. When discarding negative examples, the emphasis has sometimes been on those that are close to positive ones. In this way, one reduces the chance that the learning method might produce a classifier that would misclassify positive examples as negatives [KM97]. An alternative approach has been explored in [CS98]. It involves training several classifiers on different balanced data subsets, each constructed to include all positive training examples and a sample of negative examples of a comparable size. The predictions of these classifiers are then combined through stacking.

On the other hand, systematic experiments by [WP01] indicate that neither natural nor balanced distributions are necessarily best for training. It is evident that having a majority of positive examples in the training set is important. However, the proportion of positive and negative examples that leads to best classifiers generally depends on the data set. Implications of rebalancing the training set by modifying the number of negative examples have also been explored analytically in [Elk01]. Most of this work has been done on symbolic domains. However, discarding negative examples has also been used in text categorization [NGL97] in conjunction with the perceptron as the learning algorithm.

In this paper we investigate the problem of text categorization with the linear SVM classifier when very few positive examples are available for training. In order to investigate this issue systematically, we vary the number of positive examples in the training set and apply the resulting classifiers to test data with a natural distribution of positive and negative documents. We are particularly interested in highly imbalanced distributions with the ratio of positive to negative documents ranging from less than 0.1 % up to 10 %.

The linear SVM learner produces an optimal hyperplane $\mathbf{w}^T\mathbf{x} = b$ that separates the positive example from negative ones in the high-dimensional space of features that represent documents. The optimization problem involves determining the trade-off between the error margin and misclassification cost for the examples in the training data. The resulting hyperplane is then used to classify the new, unlabeled data.

Our results, presented in Section 6, show that imbalanced class distributions negatively affect the performance of the SVM classifiers. This appears to be mostly due to the poor estimation of the parameter b, also referred to as the score threshold since the documents with the score $\mathbf{w}^T\mathbf{x}$ above the threshold b are assigned the positive label. The value of b obtained by the original learner is too high, resulting in a too conservative assignment of positive labels to test documents. In addition, our experiments with cost-based adjustment of the dividing hyperplane, through modification of the optimization function, have shown that the learner achieves improved performance mostly through the change in the parameter b. Thus, we investigate several methods for altering the score threshold directly.

In Section 6 we show that this approach is very successful in dealing with highly imbalanced training data. Cross-validation over the training data enables us to set the threshold successfully even when the training data contains less than 0.01 % of positive examples: the macroaveraged F_1 performance over the test data is increased from 0.0166 to 0.3603. This improvement is also a considerable step towards the optimal F_1 performance achievable through the modification of b: $F_1 = 0.4724$.

We structured our paper in the following way. First we provide a brief overview of the SVM learning method, discussing its cost-based learning and score thresholding aspects. Then we describe the experimental set-up and results of the experiments that explore each of these two aspects in detail. We conclude with summary of our findings and discussion of future work.

2 Support vector machines

The support vector machine (SVM) [CV95] is a relatively recent machine learning algorithm proven to be very successful in a variety of domains. It is currently considered the state-of-the-art algorithm in text categorization.

SVM treats learning as an optimization problem. Training and test examples are represented as d-dimensional real vectors in the space of features describing the data. Given a category, each example belongs to one of two classes referred to as positive and negative class, respectively. Thus, the training set consists of pairs $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, l$, where $\mathbf{x}_i \in \mathbb{R}^d$ is the i-th training vector and $y_i \in \{+1, -1\}$ is the class label of this instance.

The learner attempts to separate positive from negative instances using a hyperplane of the form $\mathbf{w}^T\mathbf{x} = b$. Here, $\mathbf{w}$ is the "normal" of the hyperplane, i.e., a vector perpendicular to it, which defines the orientation of the plane. The constant b defines the position of the plane in space. To choose $\mathbf{w}$ and b, SVM minimizes the following criterion function:

$$\begin{array}{rl} & f(\mathbf{w}, b) := \frac{1}{2}||\mathbf{w}||^2 + C\sum_i \xi_i \\ \text{subject to} & \forall i : y_i(\mathbf{w}^T\mathbf{x}_i - b) \geq 1 - \xi_i, \quad \xi_i \geq 0. \end{array}$$

The space between the planes $\mathbf{w}^T\mathbf{x} = b \pm 1$ is called the "margin" and its width can be expressed as a function of $\mathbf{w}$: $2/||\mathbf{w}||$. The above criterion function causes the learner to look for a trade-off between minimizing the term $||\mathbf{w}||^2$, which is equivalent to maximizing the width of the margin, on one hand, and minimizing the classification errors over the training data, expressed by the term containing $\sum_i \xi_i$. The constant C, chosen before learning, defines the relative importance of these two terms. The associated inequality constraints require each training example to lie on the correct side of the hyperplane and sufficiently far from it. Otherwise the slack variable ξ_i becomes positive and decreases the value of the above criterion function for the hyperplane under consideration.

Once $\mathbf{w}$ and b are calculated, the classifier uses the following criterion to predict class labels of new documents:

$$\mathit{prediction}(\mathbf{x}) := \operatorname{sgn}(\mathbf{w}^T\mathbf{x} - b).$$

For that reason, the parameter b is referred to as the score threshold. Documents $\mathbf{x}$ with the score $\mathbf{w}^T\mathbf{x}$ above the value b are assigned the positive label.

This basic formulation has also been extended to accommodate various classification types and address issues such multiple classes and use of nonlinear classifiers.

2.1 Cost-based learning

One important extension of the SVM model is aimed at highly imbalanced training data where the ratio of positive examples to negative examples is very small. It allows us to treat classification errors on positive training examples more seriously than those on negative training examples. The general criterion function takes the form

$$f(\mathbf{w}, b) := \frac{1}{2}||\mathbf{w}||^2 + jC\sum_{i:y_i=+1} \xi_i + C\sum_{i:y_i=-1} \xi_i; \qquad (1)$$

this enables one to set the cost of misclassifying positive examples j times higher than in the original formulation. This is, in fact, equivalent to oversampling, i.e., simulating the situation in which j copies of each positive training example is used in the original optimization problem.

Based on the definition of the optimization problem we expect that the SVM learner will be rather robust with respect to the large number of negative examples. The negative examples that lie below the margin (i.e., $\mathbf{w}^T\mathbf{x}_i < b - 1$) do not have an influence on the learning process. However, the negative examples that are close to the positives, or even interspersed among them, could still be relatively detrimental to the learning process, given that the positives are not so numerous.

2.2 Thresholding strategies

Modifying the score threshold in order to improve the SVM classification performance has been explored by Platt [Pla99], who assumed that the probability that a document with the score $\mathbf{w}^T\mathbf{x}$ is positive, $P(y = +1|\mathbf{w}^T\mathbf{x})$, can be described by a sigmoid function of the score $\mathbf{w}^T\mathbf{x}$. The parameters of the sigmoid are chosen using a validation set and a new threshold is placed where the sigmoid function achieved the value of 1/2. This, in effect, aims at predicting the class with the maximum a posteriori probability. However, the separation of positive and negative examples is not perfect and such a threshold could cause a non-negligible number of positive examples to be misclassified as negative. In case of imbalanced classes, where there were few positive examples to begin with, this can lead to a poor recall and, consequently, poor F_1-measure.

Chakrabarti et al. [CRS02] have also observed that the F_1 performance of an SVM classifier can be improved by tuning the threshold with the aid of a validation set. They investigated this issue only briefly, without systematic consideration of imbalanced training data.

Methods for setting score thresholds are commonly considered in the area of information retrieval [Yang01] which also comprises various text categorization tasks. Among them is the proportional assignment of positive labels, also known as "PCut". First, the test documents are ranked based on the score which estimates the likelihood that a document is positive. A number of top ranked documents is then assigned the positive label so that the percentage of positive predictions on the test set matches the percentage of positive documents in the training set. However, such a technique is less suitable when one needs to model classification of individual documents rather than a collection of documents at once. Other thresholding strategies mentioned in [Yang01] can be used in such situations, however. In particular, "SCut" consists of selecting a separate threshold for each category using a separate validation set of documents. "RCut" assigns each document to a fixed number of categories that appear to be the most likely for this document.

3 Design of the experiments

We designed our experiments to examine cost based and threshold based improvements of the SVM learning algorithm for highly imbalanced classes. We controlled the proportion of positive documents in the training data but evaluated the methods on the test data with the natural class distributions.

We used documents from the Reuters Corpus, Volume 1 [Reu00] that consists of 806,791 Reuters articles dated 20 August 1996 through 19 August 1997. Out of those, we chose 302,323 documents dated later than 14 April 1997 for the test set. For training, we randomly selected document sets from those dated 14 April 1997 or earlier. To ensure that the training data contains a particular number or percentage of positive examples we used a different training set for each category. Thus, we treated classification of each category completely independently of the classification of the others. For each category C we built training sets $Tr(C, P, N)$ with a predefined number P of positive examples, i.e., documents belonging to category C, and a number N of negative examples, i.e., documents not belonging to C. In all our experiments, we fix $N = 10{,}000$ and vary the values of P from 8 to 1024. The training sets for different values of P are nested in the sense that $P < P'$ implies that positive examples in $Tr(C, P, N)$ are a subset of those in $Tr(C, P', N)$. The negative examples are the same in both.

We used the familiar bag-of-words model to represent the documents, eliminating stopwords and words occurring in less than four training documents. Each document was represented by a vector of features weighted by the standard TF-IDF score and was normalized to the Euclidean length of 1.

The original Reuters classification scheme involves 103 categories. In our experiments we used a subset of 16 categories: *c13, c15, c185, c313, e121, e13, e132, e142, e21, ghea, gobit, godd, gpol, gspo, m14, m143*, which were chosen on the basis of preliminary experiments with the full set of categories. We observed the performance of the SVM classifier on a smaller set of training and test data and selected the categories that cover a range of category sizes and classification difficulty, as measured by the breakeven point. The same set of categories has been already been used in our earlier work [BGMM02].

To quantify the classifier performance we calculated the F_1 measure, a function of the precision p and recall r, $F_1 := 2pr/(p+r)$, where p is the proportion of correct predictions among documents that were predicted to be positive, and r is the proportion of correct predictions among the documents that were truly positive.

We describe the performance in terms of the macroaveraged F_1 values over the set of categories. More precisely, given a particular value of P and a particular thresholding strategy, we train a classifier for each category, using $Tr(C, P, N)$ as a training set. The classifier is then applied to the test data and its F_1 performance on the test set is recorded. The macroaveraged F_1 is then simply the average of F_1 values across all 16 categories. Thus the influence of smaller categories on this performance measure is probably smaller than it would be if macroaveraged F_1 were computed over *all* Reuters categories (where the smaller categories would predominate more strongly). At the same time, the influence of the smaller categories is probably larger than it would be if microaveraged performance values were used.

We also observe the precision-recall breakeven point (BEP) [Lew91, p. 105] for individual classifiers and the macroaverage across the classifiers. As the threshold is gradually decreased, more documents are assigned positive labels, resulting in increased recall but, generally, a decreased precision. At some point the precision and recall become equal and this value of precision and recall is defined as the breakeven point (BEP). Thus, by definition, BEP does not depend upon a particular choice of a threshold but only on the ranking of document scores $\mathbf{w}^T\mathbf{x}$. Therefore, it is particularly useful for evaluating the quality of the hyperplane orientation $\mathbf{w}$ determined by the SVM learner.

We used Thorsten Joachims' SVM^{light} program [Joa99] to train the SVM models. We conducted three sets of experiments. The first set explores the effect that the cost parameter j has on the performance of the classifier. The second set investigates strategies for setting the score threshold value b. Finally, in the third set we experiment with the combination of varying j and b simultaneously.

4 Experiments with cost-based learning

In this section we explore the effects of cost adjustment in the SVM optimization problem by varying the value of j (see Section 2.1). We gradually increase j starting with the default value $j = 1$. This is expected to increase the assignment level of positive labels to documents, thus leading to an increase in recall at the expense of decrease in precision. The results of our experiments are presented in Figure 1.

As it can be seen, the j parameter has a relatively small effect on the categorization of test data. This effect is adverse if the training set contains a sufficiently large ratio of positive examples (in this case 10 %, i.e., 1024 documents). Then the gain in recall is outweighed by the loss of precision and, as a result, the F_1 measure decreases.

We also observe that most significant changes occur as j increases from its default value of 1 to $j = 2$. The effect of further increasing j is rather small. It quickly reaches the value for which no misclassifications of positive examples in the training set occur. Given that j only affects errors on positive examples,[1]

[1]Strictly speaking, it influences all the positive training examples that have $\xi_i > 0$. They need not actually be misclassified; they might also lie within the margin (i.e., on the correct side of the separating hyperplane but not far enough from it).

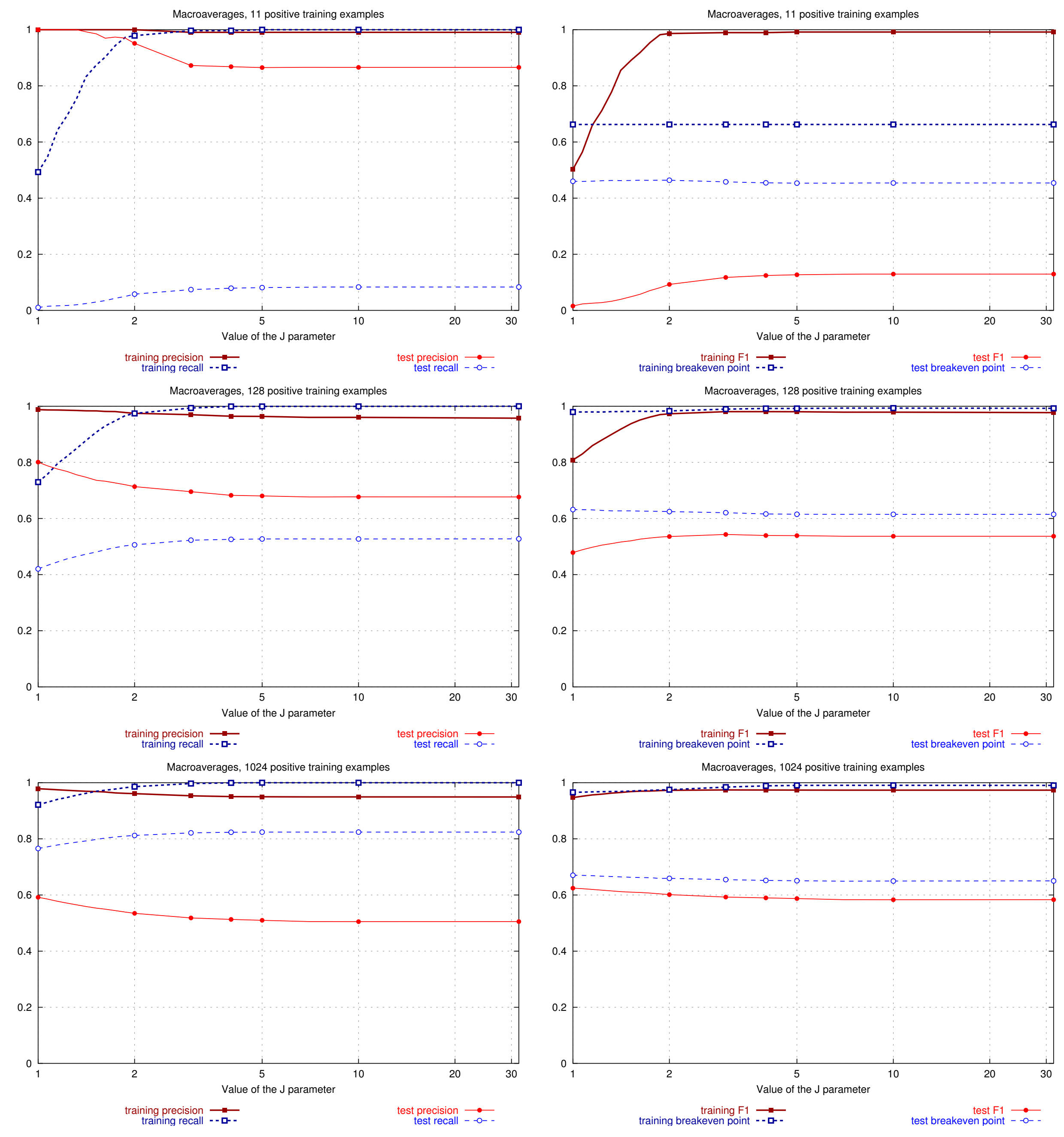


Figure 1: The dependence of macroaveraged performance measures on the cost of false negatives (the j parameter). Number of positive training examples: top row, 11 (0.1 %); middle row, 128 (1 %) bottom row, 1024 (10 %). Charts in the left column show precision and recall, and those in the right column show F_1 and the break-even point. Precision and recall behave similarly on both the training and the test set: precision decreases and recall increases with increasing values of j. F_1 increases with j, except when there are many positive training examples. Interestingly, the breakeven point hardly changes with the increase of j.

increases in j beyond that value does not influence the learning process.

The BEP hardly changes with the increase of j. On the training set it remains practically unchanged. On the test set it remains almost unchanged for small number of positive examples but shows decrease in performance for larger number of positive examples. Since BEP is only affected by the orientation of the dividing hyperplane, this suggests that the parameter j is not effective in adjusting the orientation of the hyperplane in order to improve the classification performance. As an illustration, in experiments with 11 positive examples, increasing j does change the orientation of the normal by 13,6° on average (average over the 16 categories, that is). Thus, the main effect of j on the precision, recall, and F_1 is through the change in the score threshold b.

The following sections provide detailed analysis of the effect that j has on the classifier performance for $P = 11$, 128, and 1,024 positive documents in the training corpus, which correspond to 0.1 %, 1 %, and 10 % of the training data. (Figure 1.)

Influence of j for very few positive examples ($P = 11$, i.e., 0.1 % of the training set). Recall at $j = 1$ is quite low even when applied to the training set (49.3 %), and extremely low (1.0 %) on the test set. In other words, the scarcity of positive training examples yields a very conservative classifier; on the test set, with the natural distribution of positive examples, it hardly labels any document as belonging to the positive class. However, increasing the values of j quickly improves recall over the training data, reaching recall of 97.9 % at $j = 2$ and 100 % at $j = 5$. Note that from that point on there are no misclassified positive examples and thus increasing j further does not seriously affect the learning process. In fact, the data shows that the separating hyperplane itself typically does not change from $j = 8$ onwards.

Increase in recall on the test set is much less dramatic, reaching 5.8 % at $j = 2$ and 8.1 % at $j = 5$. At the same time, precision for $j = 1$ is 100 % for both the training and the test set as the classifier is very conservative. It decreases to 99.1 % on the training set and 86.5 % on the test set as j is increased. The net result is, in some respect, captured by F_1. It increases from 50.3 % to 99.1 % on the training set and from 1.6 % to 12.9 % on the test set.

It is also instructive to observe the precision-recall breakeven point (BEP), which helps us analyze the effect of j on the orientation of the hyperplane ($\mathbf{w}$) as it is independent of the threshold b. BEP on the training data remains completely unchanged (at 66.3 %), and decreases on the test data from 46.0 % to 45.4 %. It is, thus, clear that most of the classification performance change, captured by other measures, is due to the adjustment of the score threshold.

A closer look at the data shows that changing the j parameter caused the orientation of the hyperplane to be adjusted by 13.6° on average but without a positive effect. On the other hand, the average score threshold value for 16 categories increased from 1.08 to 1.23 and the average $||\mathbf{w}||$ increased from 3.06 to 7.36. This implies that the learner has been willing to accept a narrower margin (which is always $2/||\mathbf{w}||$ units wide) as j increased. It turns out that the hyperplane has moved closer towards the origin of the coordinate system by 0.192 units on average (which is reasonable, because it makes the classifier more liberal as the threshold b was positive). This is, in fact, a rather modest shift in the hyperplane position when compared with those observed in the threshold adjustment experiments presented further in the paper. Holding the hyperplane normal fixed (as learned for $j = 1$) and tuning b to maximize F_1 over the test data moves the plane by 0.302 units on average. Setting the value of b using cross-validation on the training set moves the plane by 0.287 units on average while choosing b to be the score value where BEP occurs on the training set moves the plane by 0.232 units on average.

Thus we conclude that incrementing j has not succeeded in finding a better orientation of the hyperplane nor modifying the score threshold b appropriately to produce an effective SVM classifier.

Furthermore, comparing the weights of individual features in the resulting normals for $j = 1$ (say $\mathbf{w}$) and $j = 8$ (say $\hat{\mathbf{w}}$) shows high overlap in features with nonzero weights in the two vectors. Indeed, hardly any new features have been introduced into the normal by changing j: $\mathbf{w}$ has 4,829 nonzero components on average while $\hat{\mathbf{w}}$ has 5,097. Furthermore, if the newly-introduced features are discarded from $\hat{\mathbf{w}}$, the angle $\angle(\mathbf{w}, \hat{\mathbf{w}})$ hardly changes.

A more detailed analysis of the normals shows that practically all positive features in $\mathbf{w}$ remain positive in $\hat{\mathbf{w}}$. The features that were dropped (i.e., are zero in $\hat{\mathbf{w}}$ but not in $\mathbf{w}$) were all negative and almost all of the newly introduced features have negative weights. Furthermore, hardly any weights have changed their sign: on average 10 from negative to positive, and 12 from positive to negative. The absolute values of the weights in $\mathbf{w}$ and $\hat{\mathbf{w}}$ probably shouldn't be compared directly as length of the vector $\hat{\mathbf{w}}$ is about 2.4 times larger, on average (because of the narrower margin). However, if we normalize both normals to unit length, we observe that the number of weights with large absolute values has not changed by much, but that $\hat{\mathbf{w}}$ has more smaller weights.

The error analysis shows that for $j = 1$ there were on average 8.625 false negative classifications per category for the training set. Thus, most of the 11 positive training examples have been misclassified as negative. For $j = 8$ there are no false negatives on the training set. This suggests that the weights of positive support vectors (i.e., the corresponding α_i coefficients) have increased considerably. Generally, this could cause too many false positive classifications and, in order to prevent that, the SVM learner includes features with negative weights into $\mathbf{w}$, as it has been observed above.

Moderate number of positive examples ($P = 128$, i.e., 1 % of the training set). When the number of positive examples in the training set is $P = 128$ we observe that for $j = 1$ the recall is better and precision worse when compared with the performance for $P = 11$. Thus, having 1 % instead of 0.1 % of positive examples in the training data produces SVM classifiers that are less conservative in assigning the positive class label to documents. The recall on the training set now increases from 72.9 % for $j = 1$ to 99.9 % for $j = 4$ (note that it takes values of j between 10 and 30 to reach 100 % recall). On the test set it increases from 42.0 % to 52.7 %. At the same time, test precision drops from 80.0 % to 67.6 %, while F_1 grows from 47.8 % to 53.6 % on the test set. As before, most of these changes occurs between $j = 1$ and $j = 2$.

Abundance of positive examples ($P = 1,024$,i.e., 10 % of the training set). For $P = 1{,}024$ the performance for $j = 1$ is already relatively good with 92.1 % recall on the training and 76.6 % on the test set. Thus, the number of positive examples is sufficiently large to ensure training of effective classifiers. Increasing j to 4 yields further increase in recall, to 99.9 % on the training set and to 82.3 % on the test set. Precision on the test set decreased from 59.2 % to 50.5 % causing F_1 to *decrease* from 62.4 % to 58.3 %. Note that this is in contrast with the performance for small P where increasing j improved recall noticeably at the expense of precision but F_1 was nevertheless increased, despite the precision loss. This is consistent with the fact that the value of F_1 tends to reflect the lesser of precision and recall. For example, for small P, the recall on the test set was rather low while the precision was very high. Increasing j improved recall noticeably at the expense of precision but F_1 was nevertheless increased despite the loss in precision. For large P, the recall is not sufficiently low to begin with and the increase in recall was not sufficient to counterbalance the decrease in precision.

The fact that we are not gaining anything by increasing j in this situation is confirmed by the BEP values, which grow slightly on the training set, from 96.6 % $\rightarrow$ 99.0 % but decrease on the test set from 67.0 % $\rightarrow$ 64.9 %.

5 Score distributions and class probabilites

In the previous section, we have seen that incrementing the j parameter affects the classification performance through modifying the score threshold value b. However, we also observed that this modification was not adequate. Thus, we focus our effort on adjusting the score threshold directly. This calls for an analysis of score distributions $\mathbf{w}^T\mathbf{x}$ of positive and negative examples, both in the training and test data sets.

In this section we first investigate the score distribution of positive and negative examples with respect to the default threshold obtained by SVM over the training data. Then we estimate the class probabilities, i.e., the probability that a document with a particular score belongs to a class. Based on the findings we propose a thresholding heuristic based on class probability distributions over the training data given the SVM normal $\mathbf{w}$ in addition to the more obvious ones such as the threshold that leads to maximum F_1 measure over the training data or the threshold recommended by cross validation over the training data.

5.1 Analysis of score distributions

In order to study the distribution of values $b(\mathbf{x}) := \mathbf{w}^T\mathbf{x}$ for a given normal $\mathbf{w}$ and documents $\mathbf{x}$ from the training and test sets, we discretized the distribution by dividing the range $[\min_{\mathbf{x}} \mathbf{w}^T\mathbf{x}, \max_{\mathbf{x}} \mathbf{w}^T\mathbf{x}]$ into 50 subintervals of equal width. We recorded the number of positive and negative examples in each score interval. Figure 2 shows the resulting distributions (*C183*).

We are particularly interested in score distribution obtained by the SVM learner with $j = 1$ over the training data and the score value b^* that corresponds to the maximum F_1 measure performance over the test data. The objective of our thresholding methods is to adjust b_0 automatically to values close to b^*.

Of interest are also the score values of $b_0 - 1$ and $b_0 + 1$ where we expect concentration of support vectors.

From Figure 2 we observe that, for $P = 1024$ and 128 positive documents in the training data, SVM scores of positive and negative documents lead to similar distributions over the positive and negative documents. The distributions are not necessarily separable but the consistency in distributions provides solid foundation for performance prediction using score thresholds.

We note that the distribution of negative training documents tends to reach maximum around $b = b_0 - 1$, at the location of negative support vectors. In fact, the actual peak is typically at some b strictly below $b_0 - 1$. This is consistent with the SVM optimization strategy: any data instance positioned within the margin entails a cost, measured by the slack variable ξ_i.

Similarly, the distribution of positive training examples tends to have a spike at $b = b_0 + 1$ because of the concentration of positive support vectors at the unit distance from the hyperplane. Interestingly, $b = b_0 + 1$ is typically the peak of the distribution in contrast to the score distribution of negative examples. We also observe that the score distributions of positive documents have more symmetrical shape than that of the negative documents. In the future we will investigate whether this characteristic of score distributions is correlated with the relative proportion of positive and negative examples in the training set.

We note that in case of $P = 128$, the score distribution for positive test documents is much broader than the one for positive training documents, indicating that many positive test examples are assigned low values of $\mathbf{w}^T\mathbf{x}$ and will therefore be misclassified as negative. This is the sign of overfitting to the one percent of positive examples.

In the case of extremely imbalanced data, e. g., $P = 11$ (the top row of Figure 2), we see that the learner separates the positive and negative training examples well but is much less effective in doing so on the test data. The classifier suffers from serious overfitting. Furthermore, estimation of positive score distribution over the test data is unreliable with very few data points. Thus, it is hard to modify the threshold based on score distribution solely. However, from the experiments we also observe that, in the case of very few positive training examples (in this case 0.1 %), the SVM learner typically selects a threshold so that practically all of the positive documents are support vectors; some actually lie within the margin, but usually none lie on the negative side of the margin (i.e. below the plane $\mathbf{w}^T\mathbf{x} = b_0 - 1$). This suggests another strategy selecting a better threshold by using $b = b_0 - 1$ instead of b_0. This simple heuristic is to a certain degree incorporated in the method we propose in the following section.

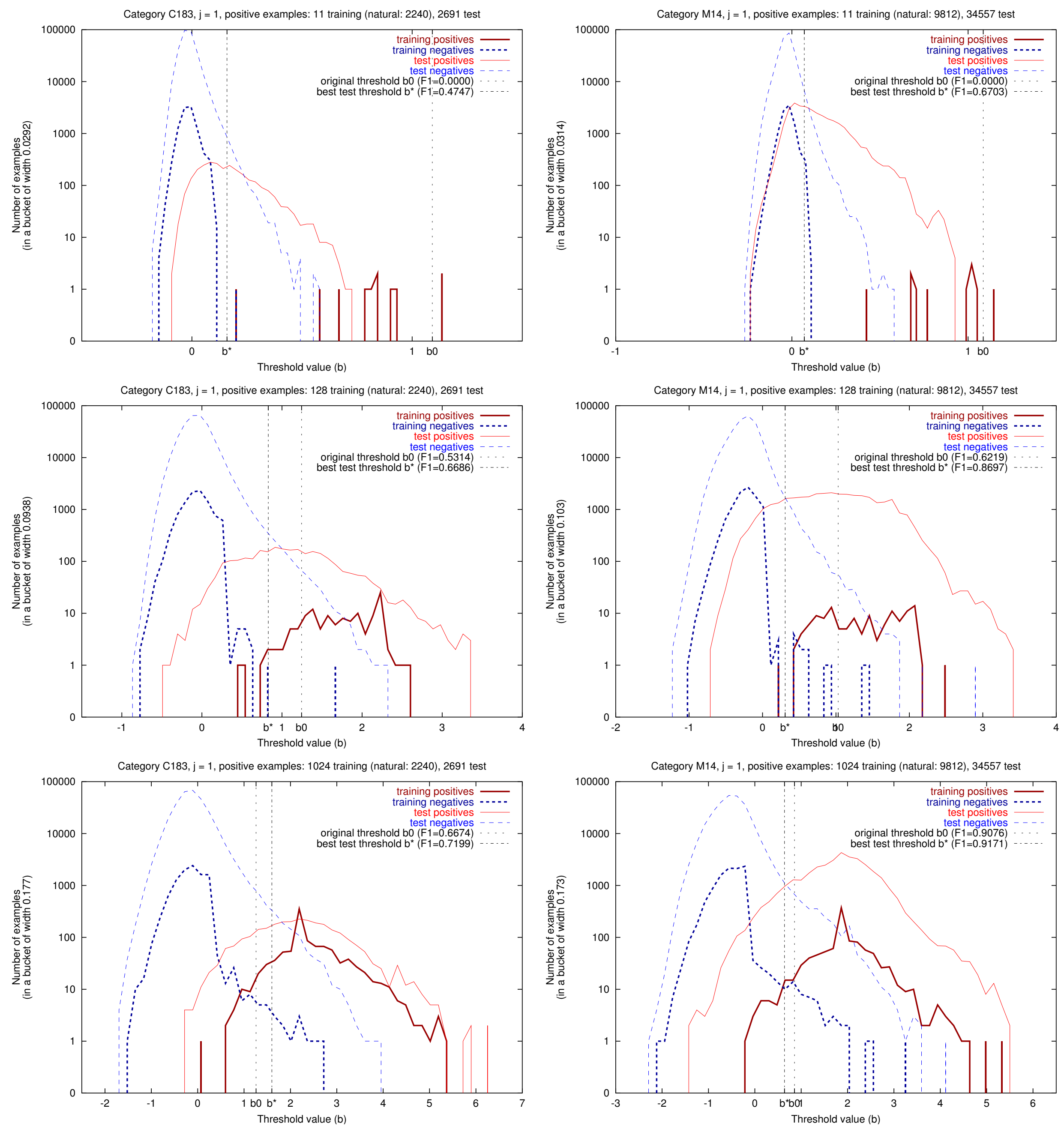


Figure 2: These charts show how many documents fall at a certain distance from the plane. For illustration, we show the analysis for two categories: *C183* is shown on the left as an example of a smaller category, and *M14* on the right as an example of a large category (charts for other categories can be found in the appendix, pp. 21–23). Number of positive training examples: top row, 11; middle row, 128; bottom row, 1024. The two vertical lines show the original threshold obtained by training SVM and the hypothetical best threshold calculated from the test data, just for comparison. The bottom two graphs show the analysis for a very small number of positive examples ($P = 11$): note how the distribution of positive test examples has hardly any relation to that of the positive training examples.

5.2 Analysis of the class probabilities

We explore the possibility of using class probabilities to identify successful score thresholds in case of highly imbalanced classes.

Given the orientation of the hyperplane $\mathbf{w}$ and some value b we estimate the probability that a document $\mathbf{x}$ assigned the score $\mathbf{w}^T\mathbf{x}$ belongs to the positive class as $P(\oplus|b) := P(y = +1|\mathbf{w}^T\mathbf{x} = b)$. We apply similar methods as in the previous section. We divide the range of values $\mathbf{w}^T\mathbf{x}$ into subintervals (about three times as long as those in the analysis of score distributions) and record the relative frequency of positive examples among all the documents with scores within a given interval.

The distribution curves (Figure 3) have a familiar sigmoidal shape. For training examples the estimated $P(\oplus|b)$ is 0 for low scores b and grows rapidly towards 1, the value that is maintained for the rest of the document scores, confirming that SVM scoring of training examples does not introduce false positives among highly scored documents.

On the graphs we indicate the default score threshold b_0 that SVM produces during the learning process and the threshold b^* that maximizes the F_1 performance measure on the test set. We note that b^* usually occurs close to the value of b where the probability $P(\oplus|b)$ for the training set first begins to grow in value, i.e., the score interval with significant number of positive examples is encountered. This is consistent with our observation about the distribution of scores in the previous section: in case of highly imbalanced training data it seems suitable to adjust the default threshold to the value of $b_0 - 1$ (thus moving the separating plane to the lower edge of the original margin area).

We, thus, suggest a simple heuristic for selecting a new score threshold. We choose the smallest value of b for which the probability that an instance is positive, given that its score belongs to the corresponding interval, is above a certain small predefined threshold, e.g., $\eta = 2\,\%$. The exact value of η should not be very important, considering how immediate and large the first increase in $P(\oplus|b)$ appears to be on these distribution charts for $P = 11$.

6 Experiments with threshold modification

In this section we evaluate several methods for setting score thresholds b. We assume that the SVM learner has provided a hyperplane $\mathbf{w}^T\mathbf{x} = b_0$ under the default setting of $j = 1$. We intend to modify only the threshold (replace b_0 by some other value b), keeping its normal ($\mathbf{w}$) unchanged. This, therefore, preserves the orientation of the plane but influences the level of assignment of the positive labels to the test documents; a higher (lower) threshold leads to more conservative (more liberal) assignment. Success of the thresholding mechanism is measured by comparison with the optimal value of the F_1 measure for the test data.

6.1 Comparison of different thresholding heuristics

Several methods for modifying the threshold are considered: selecting the score (1) that maximizes the value of F_1 over the training data; (2) cross-validation over subsets of training data, observing the F_1 measure on individual subsets; and (3) based on a score that corresponds to a specified value of positive class probability as described in Section 5.2. For the sake of comparison, we cosider the threshold that maximizes F_1 over the test data which is, of course, unavailable in practice. The results of these experiments are summarized in Figure 5 and Table 2.

We observe that although the performance of the original threshold obtained by the SVM learner is very poor when the number of positive training examples was low, the same model can be considerably improved by choosing a different threshold. In fact, by choosing a threshold using cross-validation on the training set, we get to within 70 % of the optimal threshold, the one that maximizes the F_1 performance measure on the test data. Additionally, placing a threshold on the training break-even point or using the threshold that maximizes training F_1 also turns out to be a reasonably successful strategy; it achieves about 50 % of the optimal test F_1.

The following is a detailed analysis of these results

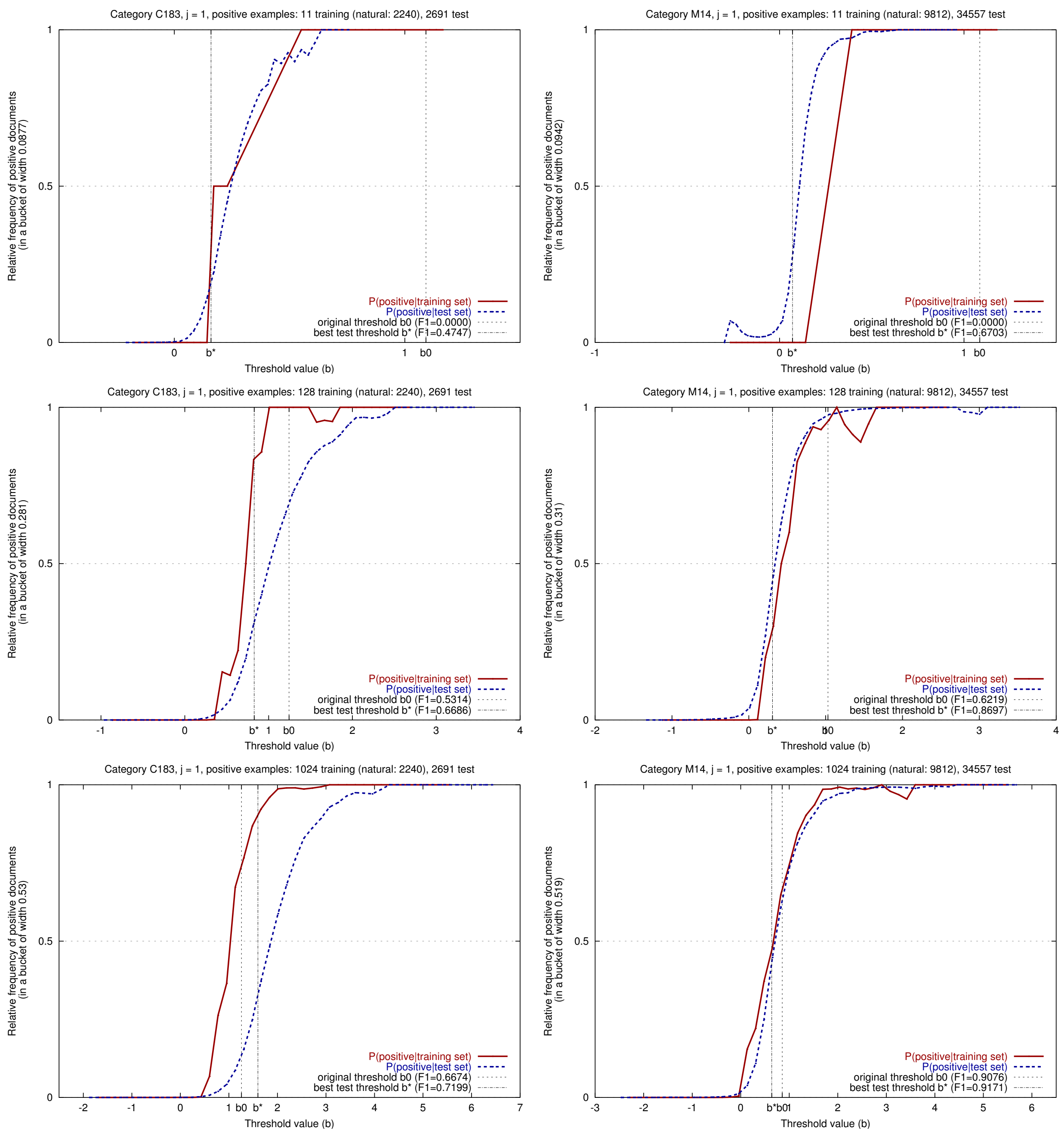


Figure 3: These charts show the proportion of positive documents among those that fall at a certain distance from the plane. For illustration, we show the analysis of two categories: *C183* is shown on the left as an example of a smaller category, and *M14* on the right as an example of a large category (charts for other categories can be found in the appendix, pp. 24–26). Number of positive training examples: top row, 11; middle row, 128; bottom row, 1024. The two vertical lines show the original threshold obtained by training SVM and the hypothetical best threshold calculated from the test data, just for comparison.

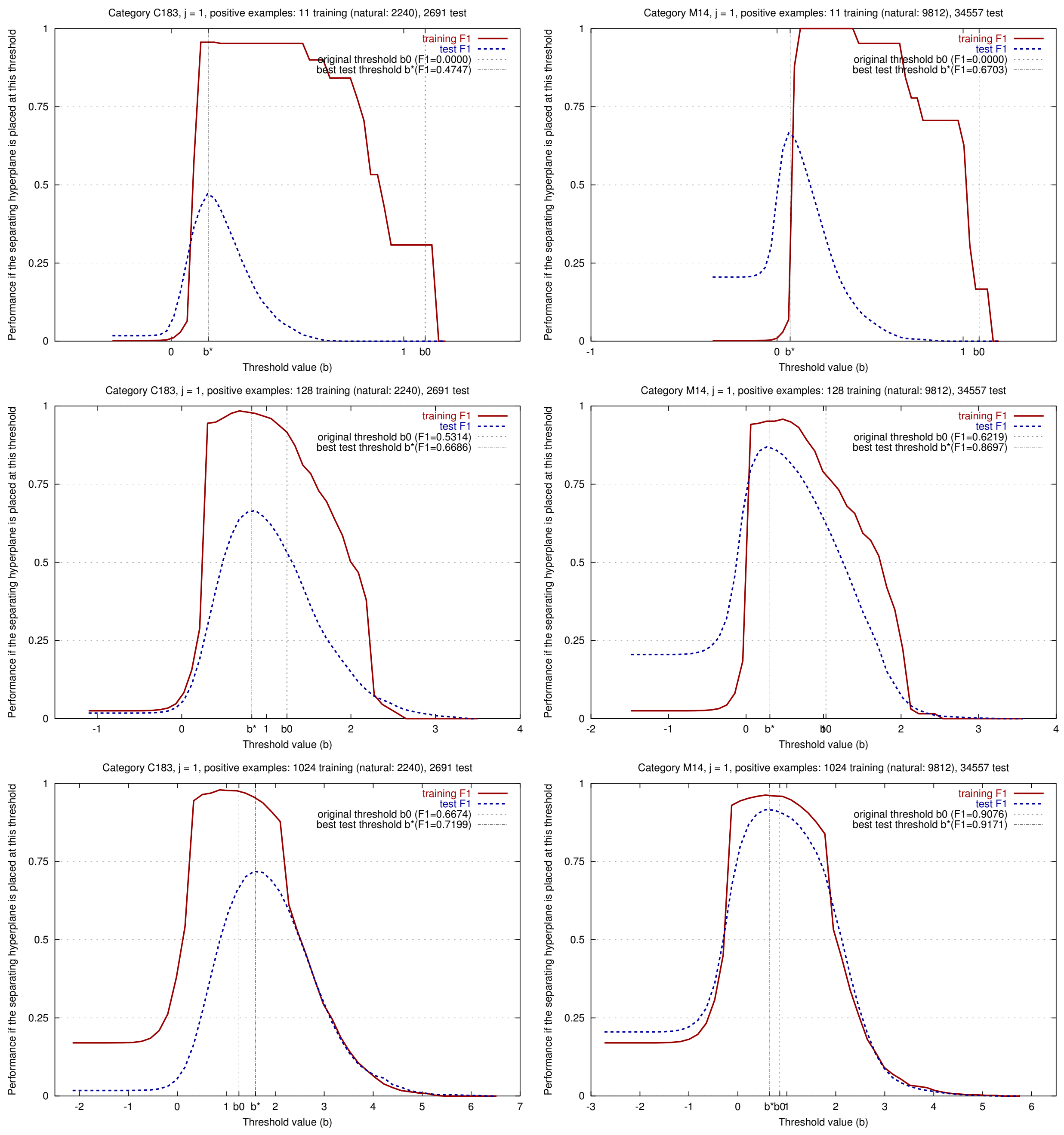


Figure 4: These charts show how F_1 depends on the threshold b of the separating hyperplane $\mathbf{w}^T\mathbf{x} = b$ while its normal ($\mathbf{w}$) remains unchanged. *C183* is shown on the left as an example of a smaller category, and *M14* on the right as an example of a large category.

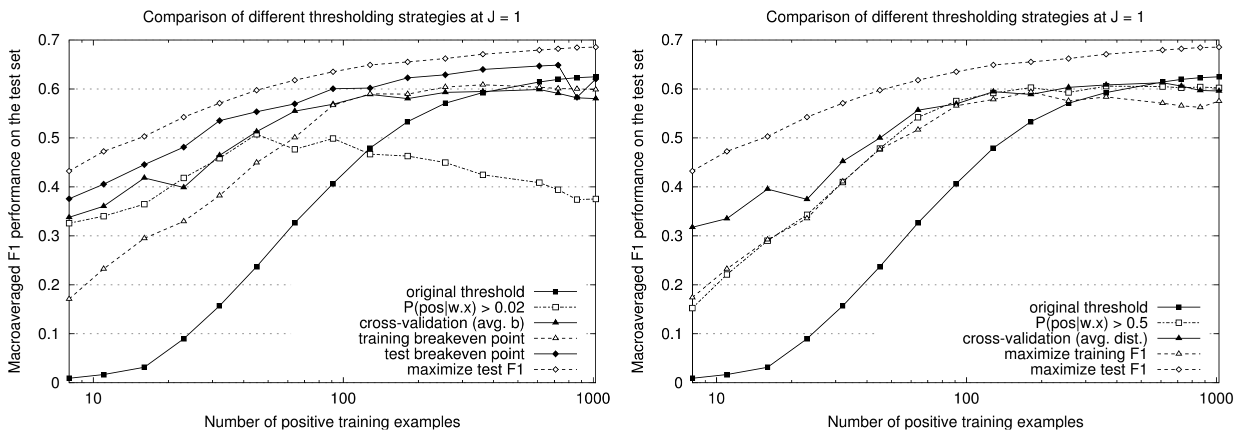


Figure 5: A comparison of various strategies for modifying the threshold of the separation hyperplane obtained by the SVM learner. Note that the macroaveraged test F_1 performance measure is shown in all cases. The explanations in the legend refer to the thresholding strategy used to modify the threshold of the hyperplane found by the SVM learner.

for $P = 11$ positive training examples. The threshold as proposed by the SVM learner is usually much too high, already for the training set but even more so for the test set (as we can see from Figure 5; recall: 21.6 % on the training set, 1.7 % on the test set; precision: 56.3 % and 18.8 %). It is interesting that while many categories are linearly separable problems and a threshold exists that would achieve 100 % accuracy on the training set, the SVM prefers to choose a much higher threshold that misclassifies most positive training examples as negative. The reason is probably that this gives it the illusion of a much wider margin (the misclassified positive examples are ignored for the purposes of margin classificaton and while their slack variables do penalize the plane a little, this is apparently not enough to overcome the attraction of the wide margin).[2]

6.1.1 Maximizing training F_1

Threshold modification by choosing b to be the value that maximizes F_1 on the training set improves the macroaverage F_1 performance from and 1 % to 23.2 % on the test set (Figure 5; on the training set, macroaveraged value rises to 99.7 %).

An alternative to this heuristic might be to place the threshold at the break-even point, i.e., so that the precision and recall on the training set are roughly the same. It is not unreasonable to expect a good F_1 performance around the break-even point, because the F_1-measure tends to rewards circumstances where precision and recall are similar rather than those when one of them has been increased at the expense of the other. It turns out that the thresholds placed at the breakeven point are very close to those at maximum training F_1, and the resulting performance on the test set is very similar as well.

6.1.2 Cross-validation

We are aware that by considering the maximum F_1 value over the whole training we run the risk of overfitting the value of b to the training data. In order to reduce that risk we apply stratified 5-fold cross-validation, hoping to identify a more robust score threshold. We train the SVM classifier on 80 % of the training examples and modify the threshold to maximize F_1 on the remaining 20 % of the training set. In the end, each of the five iterations during

[2]To avoid this, one might be tempted to increase the overall misclassification cost, i.e., the C parameter. However, Section 8 will show that this is not helpful.

cross-validation proposes a possibly different normal $\mathbf{w}_i$ and score threshold b_i. As our final model, we use the normal trained over the entire training set but calculate the threshold b from the five b_i parameters obtained from cross-validation. The average, $b := \frac{1}{5}\sum_{i=1}^{5} b_i$, is one obvious choice. However, we might be concerned by the fact that b is not really expressed in any absolute units but is relative to the width of the margin. The role of b, as far as the classifier is concerned, is really to define the distance between the hyperplane and the origin of the coordinate system. For the plane $\mathbf{w}^T\mathbf{x} = b$, this distance is $b/||\mathbf{w}||$. Thus, we can average the distances, $d := \frac{1}{5}\sum_{i=1}^{5} b_i/||\mathbf{w}_i||$, and set the threshold of the final plane so as to place the plane at this distance from the origin: $b := d \cdot ||\mathbf{w}||$. It turns out that the thresholds proposed by these two approaches are usually quite similar. Averaging the values of b gives better performance at $P = 11$ ($F_1 = 36.0\,\%$ compared to $33.5\,\%$ obtained from averaging the distance), while at $P = 128$ and $P = 1024$ averaging the distance is slightly more successful, although the differences are small.

6.1.3 Thresholding from class probabilities

Based on the observations in section 5.2, we consider the class probability $P(\oplus|b)$ that $\mathbf{x}$ is positive given $b = \mathbf{w}^T\mathbf{x}$, and place the score threshold at the smallest b where this probability grows beyond some small positive value η. In this case we used $\eta = 0.02$.

As Figure 5 shows, this heuristic is very successful when very few positive training examples are available. However, from $P = 64$ upwards, this heuristic leads to relatively poorer performance, indicating that the resulting classifier is too liberal in assigning the positive class values. This is reasonable, given that the probability $P(\oplus|b)$ as a function of b no longer has such a sudden and steep increase from 0 but instead assumes a more continuous sigmoidal shape (see the bottom two rows of Figure 3).

If we take the "Bayesian" approach, i.e. choosing the threshold b such that at $P(\oplus|b) = 1/2$, in order to classify each instance into the class with the greater a posteriori probability, we see that the resulting classifier is less successful for small values of P. However, for higher values of P this approach improves the performance and for $P > 64$ it matches the effectiveness of the cross-validation method (Sec. 6.1.2).

6.1.4 Maximizing test F_1

In order to put the above results into a perspective, it is informative to look at the maximum F_1 that can be achieved on the test data just by modifying the threshold. If the threshold is chosen optimally for each category, the macroaveraged F_1 achieves the value of $47.2\,\%$ on the test set. Incidentally, if the threshold is placed where the precision-recall breakeven point occurs on the test set, the resulting classifier is too conservative and its F_1 is on average about 0.05 less than the optimal F_1.

6.2 Range of good thresholds

As a way of assessing how difficult it is to select a good score threshold, we examine the distribution of scores over the test data. We determine the range of score thresholds that, if chosen, would lead to a specified performance over the test data, e.g. $80\,\%$ of the optimal F_1 value on the test set. The narrower this range is, more difficult it is to choose a good threshold. In other words, we are interested in the width of the bell-shaped portion of the curves in Figure 4.

Given a hyperplane normal $\mathbf{w}$, suppose that b^* is the threshold that maximizes test F_1 for this $\mathbf{w}$. Let a constant a specified the level of F_1 performance to be achieved. We define:

$$b_1 := \max\{b < b^* : F_1(\mathbf{w}, b) < aF_1(\mathbf{w}, b^*)\}$$
$$b_2 := \min\{b > b^* : F_1(\mathbf{w}, b) < aF_1(\mathbf{w}, b^*)\}$$

and let $b_{\min} := \min_i \mathbf{w}^T\mathbf{x}_i$, $b_{\max} := \max_i \mathbf{w}^T\mathbf{x}_i$, where i goes over all test documents. Then we can regard $\kappa_a(\mathbf{w}) := (b_2 - b_1)/(b_{\max} - b_{\min})$ as an indicator of the difficulty of choosing a good threshold for the particular classifier under consideration. The measure is relative to $b_{\max} - b_{\min}$ to make it easier to compare different classifiers. Alternatively, we might convert $b_2 - b_1$ to Euclidean distance between the hyperplanes $\mathbf{w}^T\mathbf{x} = b_1$ and $\mathbf{w}^T\mathbf{x} = b_2$, i.e. $\kappa'_a(\mathbf{w}) := (b_2 - b_1)/||\mathbf{w}||$. The values of these indicators, shown in Table 1 for $a = 0.8$, suggest that,

j	P	Macroaveraged values $F_1(b^*)$	$\kappa_{0.8}$	$\kappa'_{0.8}$
1	11	0.4724	0.1136	0.0540
1	128	0.6490	0.2186	0.0650
1	1024	0.6855	0.2159	0.0586

Table 1: The $\kappa_{0.8}$ and $\kappa'_{0.8}$ values indicate how wide an interval of thresholds (b) around the optimal threshold b^* (i.e., the threshold that maximizes F_1 on the test set) result in F_1 that is at least 80 % of the optimal. The lower these values, the more difficult it is to choose a good threshold without looking at the test set. See the text for a formal definition of κ and κ'.

		Macroaveraged F_1 on the test set					
j	P	At original threshold	At training BEP	Cross-validation (avg. b)	Cross-validation (avg. dist.)	At optimal threshold	Thr. from $P(\oplus\|\mathbf{w}^T\mathbf{x})$ distr.
1	11	0.0166	0.2324	0.3603	0.3351	0.4724	0.3334
1	128	0.4795	0.5901	0.5889	0.5944	0.6490	0.4668
1	1024	0.6249	0.5987	0.5803	0.5958	0.6855	0.3753
100	11	0.1311	0.0916	0.3500	0.3336	0.4651	0.2997
100	128	0.5375	0.2595	0.5666	0.5853	0.6334	0.3750
100	1024	0.5841	0.5590	0.5666	0.5853	0.6661	0.6025

Table 2: The performance of classifiers based on different thresholding strategies.

indeed, it is more difficult to choose a good threshold in situations when there were few positive training examples (i.e., small value of P), although the results, particularly for κ', are somewhat inconclusive.

7 Using both j and b

In this section, we consider applying both cost-based learning (via the j parameter) and threshold modification (the b value), using several thresholding heuristics.

If we decide to retain the original threshold as learned by the SVM, we saw (in Section 4) that increasing the value of j improves performance if the number of positive examples is small (e.g. $P = 11$ and 128) but degrades performance slightly if the number of positive examples is large enough: at $P = 1024$, $F_1 = 62.5\,\%$ for $j = 1$ but 58.4 % for $j = 100$.

Due to the scarcity of positive examples on the training set, our classifiers are too conservative. The threshold b needs to be lowered to improve the performance on the test set. For $j = 1$, setting the value of b to the score of the BEP for the training data makes the classifier more liberal. For $j = 100$ it actually increases the threshold and makes the classifier more conservative, leading to performance that is worse than that of the original threshold.

In the models obtained at $j = 100$, all the 11 positive training examples become unbounded support vectors, i.e., they have $0 < \alpha_i < jC$ and lie on the plane $\mathbf{w}^T\mathbf{x} = b_0 + 1$, parallel to the separating hyperplane $\mathbf{w}^T\mathbf{x} = b_0$. The negative examples all lie below the plane $\mathbf{w}^T\mathbf{x} = b_0 + 1$; some of them are unbounded (i.e., lie on $\mathbf{w}^T\mathbf{x} = b_0 - 1$), some are bounded (i.e. have $\alpha_i = C$) and thus lie above $\mathbf{w}^T\mathbf{x} = b_0 - 1$; some of these latter even lie above $\mathbf{w}^T\mathbf{x} = b_0$, meaning that they are misclassified by this model. None lie above $\mathbf{w}^T\mathbf{x} = b_0 + 1$, however.

Now suppose that the threshold of the plane is modified from b_0 to some new value b. For values of $b > b_0 + 1$, the preceding paragraph shows that all the training examples would be predicted negative by such a plane, resulting in 0 % recall and (by definition) 100 % precision; for $b < b_0 + 1$, recall would be 100 % as all positive examples would be predicted positive.

Let $b_\ominus := \max_{i:y_i=-1} \mathbf{w}^T\mathbf{x}_i$ be the lowest threshold below which all the negative training examples are located. For values of $b \in (b_\ominus, b_0 + 1)$, all positive examples are declared positive, so precision is still 100 %, but, as b drops below $b_\ominus$, more and more negative training examples are misclassified as positive

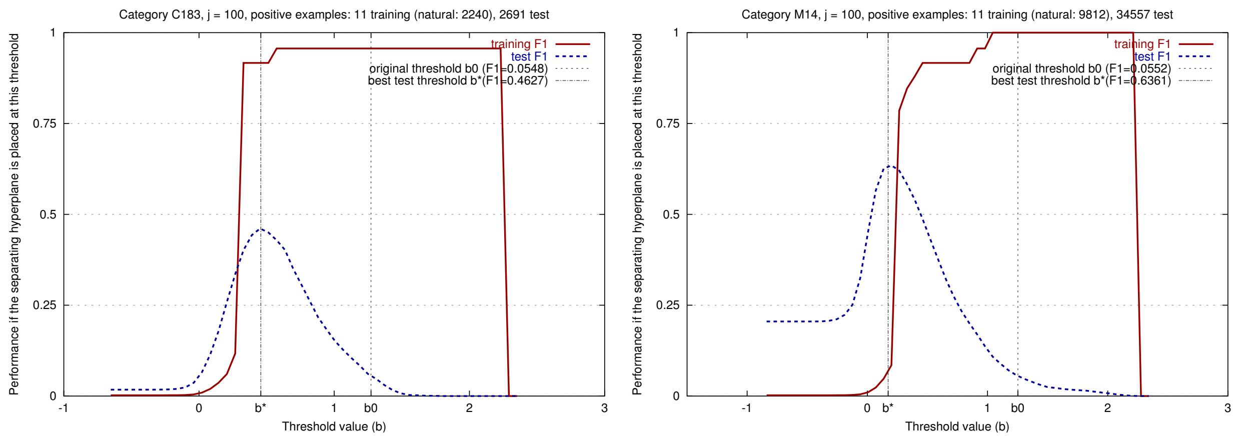


Figure 6: The dependency of F_1 on the threshold b while the normal $\mathbf{w}$, obtained by learning at $j = 100$, remains unchanged.

and precision begins to decrease.

Thus we see that precision and recall are both 100 % for all $b \in (b_\ominus, b_0 + 1)$; for $b > b_0 + 1$, precision is greater than recall, and for $b < b_\ominus$, recall is greater than precision. If we decide to choose b so as to maximize the F_1-measure on the training set, or to place it on the precision-recall break-even point, both criteria give the same proposal: b must lie somewhere in $(b_\ominus, b_0+1)$, but they cannot be more specific than that.

The results shown in Table 2 above use the middle of the range thus proposed, i.e., $b := (b_\ominus + b_0 + 1)/2$. Note that the existence of bounded negative support vectors implies that $b_\ominus > b_0 - 1$ and hence $b > b_0$. Thus our newly proposed threshold b actually makes the classifier more conservative than the original threshold at b_0. (Even if C were increased, thereby forcing the learner to avoid placing negative training examples above the plane $\mathbf{w}^T\mathbf{x} = b_0 - 1$, $b_\ominus$ would still be $\geq b_0 - 1$ and the new threshold proposed by these heuristics would still be $\geq b_0$, thus not making the classifier any more liberal.) Indeed it is hard to expect a heuristic to propose a useful new threshold for a hyperplane that cannot distinguish between different positive training examples at all (because they are all equally distant from it, i.e., all on $\mathbf{w}^T\mathbf{x} = b_0 + 1$): the new threshold can really only tell how far from the positive examples should the new hyperplane be placed, and it is not obvious how to do this in a principled way and so as to achieve as much as possible for performance on the test set.

If cross-validation on the training set is used to modify the threshold, the performance is roughly the same in both cases (for $j = 1$ and $j = 100$), and insofar as there is a difference it is in favour of $j = 1$. The same is true for the optimal threshold (i.e., the one that maximizes the F_1 measure on the test set). Both of these results suggest that the *orientation* of the original plane (the one obtained for $j = 1$) is actually slightly better than of the one for $j = 100$; in a sense, $j = 100$ caused the learner to slightly overfit the training set.

The general conclusion of this section is that it confirms the observations of Section 4 that the hyperplanes obtained at large values of j do not really have a better orientation than those obtained at $j = 1$; that modifying the threshold is a much more successful way of improving the original classifier; and that, if one decides to modify the threshold, the j parameter can be left at its original value of 1 and there is no need to tune it.

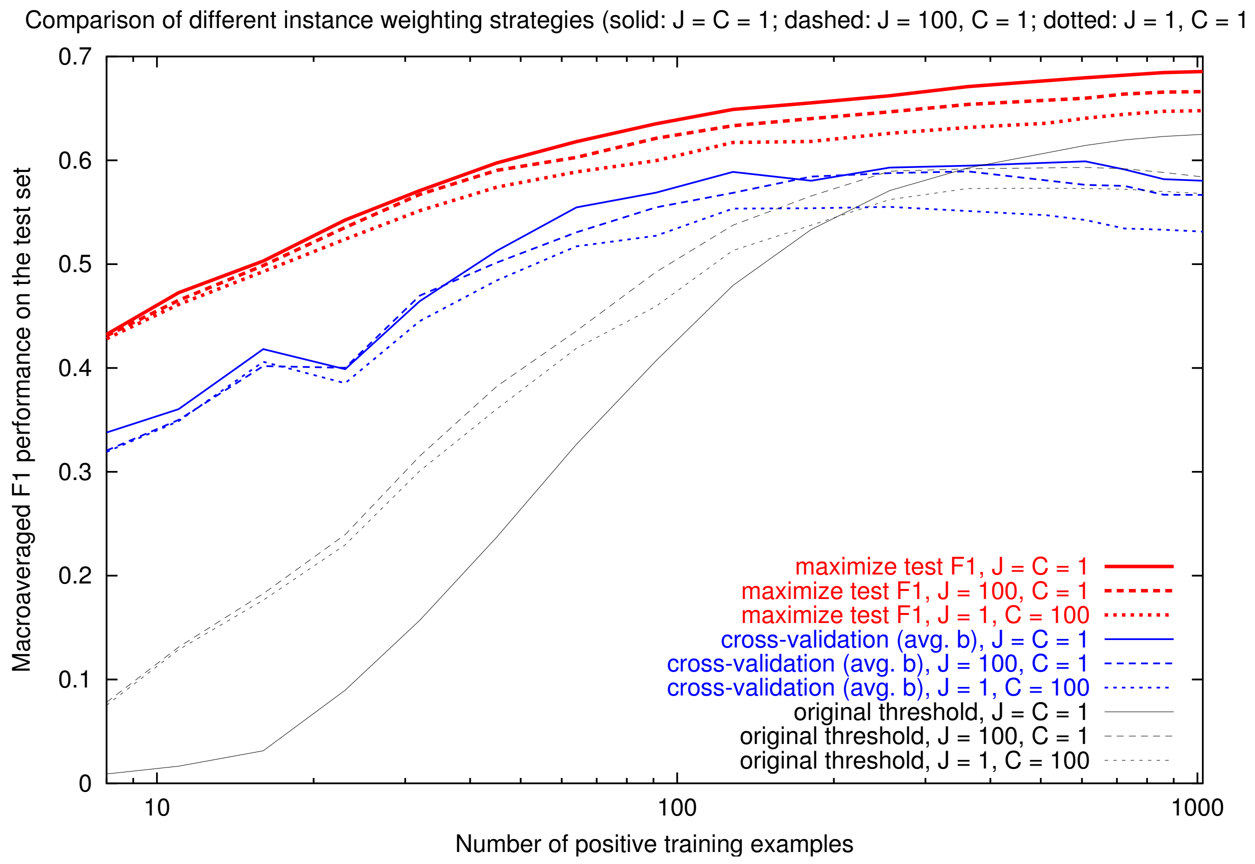


Figure 7: A comparison of various strategies of increasing the influence of misclassified training examples on the criterion function optimized by the SVM learner. $C = j = 1$ is the default setting; $C = 1, j = 100$ increases the influence of positive examples by a factor of 100; $C = 100, j = 1$ increases the influence of all examples by a factor of 100. The margin maximization term of the criterion function remains the same in all cases.

8 A comparison of weighting strategies

Recall that the criterion function (1) optimized by SVM during training requires the learner to seek a tradeoff between maximizing the margin and classifying the training examples correctly. The relative importance of these two goals is determined by the C parameter: the greater the C, the more important it will seem to classify the training examples correctly. In effect, this can be seen as a kind of noise handling.

In Section 6, we have observed that the poor performance of the classifiers obtained by SVM on a small number of positive examples is chiefly due to the fact that optimization leads to the maximum margin at a low cost of placing the positive examples within the margin. Indeed, at the default value $C = 1$ it can afford to practically ignore most of the positive training examples when computing the width of the margin.

The most straightforward way of forcing the SVM learner to take the training examples more seriously is by increasing the C parameter. This can be seen as an alternative to the j parameter: instead of increasing the weight of errors on positive training examples only, C affects the weight of errors on both positive and negative examples. The influence of the margin maximization criterion on the learner's choice of the model is thus correspondingly smaller.

Our experiments show that increasing C is not more effective than increasing j. This is illustrated by the chart in Figure 7. If extremely few positive training examples are available, increasing C has practically the same effect as increasing j, whereas in all other circumstances increasing C is in fact slightly less successful than increasing j (this can be seen by comparing the dashed and dotted lines in Figure 7). Both these approaches are inferior to leaving all the weights at their default values and adjusting the threshold b using cross-validation on the training set. In other words, increasing the influence of misclassification costs above the default values caused SVM to produce hyperplanes of a slightly less successful orientation.

9 Conclusions

We have explored the use of SVM for text categorization in cases where only a small percentage of positive examples are available in the training set. Our experiments indicate that, although the straightforward cost-based approach yields an increase in performance, this improvement is of limited extent compared to the effect of modifying the threshold b. For example, the macroaveraged F_1 increases from 0.0157 to 0.1295 using the cost-based approach while direct modification of the threshold achieves F_1 performance of 0.3603. The increase in performance brought about by the cost-based approach is, in fact, due largely to threshold adjustment rather than improvement in the orientation of the separating hyperplane found by the SVM learner.

In the case of few positive training examples it is difficult to infer the distribution of the scores of positive test examples (Section 5.1). Thus, it seems unlikely that one can successfully make use of the properties of score distributions to modify the threshold. As the number of positive training examples increases this is much easier. However, by observing the class probabilities (Section 5.2) we find that setting the score threshold at the point where the probability of the positive class begins to grow beyond negligibly small values (e.g., at $P(\oplus|b) = 0.02$) is more effective for cases where very few positive training examples are available, while placing the threshold at $P(\oplus|b) = 0.5$ (a kind of maximum a posteriori predictor) is very effective for large positive classes.

The most successful thresholding technique explored in our experiments, cross-validation over the training data, achieves more than 70 % of the best F_1 performance that could possibly be attained by without changing the orientation of the SVM determined hyperplane. For example, with the hyperplane obtained after learning with 11 positive training examples, the best threshold could achieve a macroaveraged F_1 of 0.4724, while the threshold based on cross-validation achieves $F_1 = 0.3603$.

Future work will extend this research to compare our results with alternative thresholding techniques used in information retrieval and other techniques that deal with imbalanced class distributions, such as discarding negative examples or training several classifiers and combining them with stacking. In addition, it would be interesting to explore the effects of thresholding in cases when feature selection is performed before training, as is commonly done in text categorization. And since our experiments indicate that the hyperplane found by the SVM algorithm when presented with very few positive training examples has a reasonably good orientation but an overly conservative threshold, it would be interesting to explore whether the criterion function used by SVM could be redefined so as to focus on looking for a good orientation without regard for the threshold, with the understanding that the threshold would be selected afterwards (and that efforts to force SVM to choose a good threshold by itself, e.g. via the j parameter, tend to lead it to a hyperplane with a slightly better threshold and a less useful normal). However, this may be problematic because a threshold is necessary in order to define the notion of a margin, which is one of the key components that made SVM such a successful learning algorithm.

References

[BGMM02] Janez Brank, Marko Grobelnik, Nataša Milić-Frayling, Dunja Mladenić: *Interaction of feature selection methods and linear classification models.* Proc. ICML-2002 Workshop on Text Learning, pp. 12–17, 2002. Longer version available as Microsoft Research Technical Report MSR-TR-2002-63, 12 June 2002.

[CRS02] Soumen Chakrabarti, Shourya Roy, Mahesh V. Soundalgekar: *Fast and accurate text classification via multiple linear discriminant projections.* Proc. 28th Int. VLDB Conf., Hong Kong, China, August 20–23, 2002.

[CS98] Philip K. Chan, Salvatore J. Stolfo: *Toward Scalable Learning with Non-Uniform Class and Cost Distributions: A Case Study in Credit Card Fraud Detection.* Proc. 4th Int. Conf. on Knowledge Discovery and Data Mining (KDD-98), August 27–31, 1998, New York City, New York, USA, pp. 164–168. AAAI Press.

[CV95] Corinna Cortes, Vladimir N. Vapnik: *Support-vector networks.* Machine Learning, 20(3):273–297, September 1995.

[Elk01] Charles Elkan: *The foundations of cost-sensitive learning.* Proc. 17th IJCAI, Seattle, Washington, USA, August 4–10, 2001, pp. 973–978. Morgan Kaufmann.

[Jap00] Nathalie Japkowicz: *Learning from Imbalanced Data Sets: A Comparison of Various Strategies.* In Nathalie Japkowicz (ed.), *Learning from Imbalanced Data Sets: Papers from the AAAI Workshop* (Austin, Texas, Monday, July 31, 2000), AAAI Press, Technical Report WS-00-05, pp. 10–15.

[Joa99] Thorsten Joachims: *Making large-Scale SVM Learning Practical.* In: B. Schölkopf, C. Burges, A. Smola (eds.), *Advances in Kernel Methods — Support Vector Learning.* Cambridge, MA: MIT Press, 1999, pp. 169–184.

[KM97] Miroslav Kubat, Stan Matwin: *Addressing the curse of imbalanced training sets: one-sided selection.* Proc. 14th ICML, Nashville, Tennessee, USA, July 8–12, 1997, pp. 179–186.

[Lew91] David D. Lewis: *Representation and Learning in Information Retrieval.* PhD thesis (and Tech. Rept. UM-CS-1991-093), CS Dept., Univ. of Massachusetts, Amherst, MA, USA, 1992.

[Pla99] John Platt: *Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods.* In: A. Smola, P. Bartlett, B. Schölkopf, D. Schuurmans (eds.): *Advances in large margin classifiers.* Cambridge, MA: MIT Press, 1999, pp. 61–74.

[NGL97] Hwee Tou Ng, Wei Boon Goh, Kok Leong Low: *Feature Selection, Perceptron Learning, and a Usability Case Study for Text Categorization.* Proceedings of the 20th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 67–73). Philadelphia, PA, USA, July 27–31, 1997.

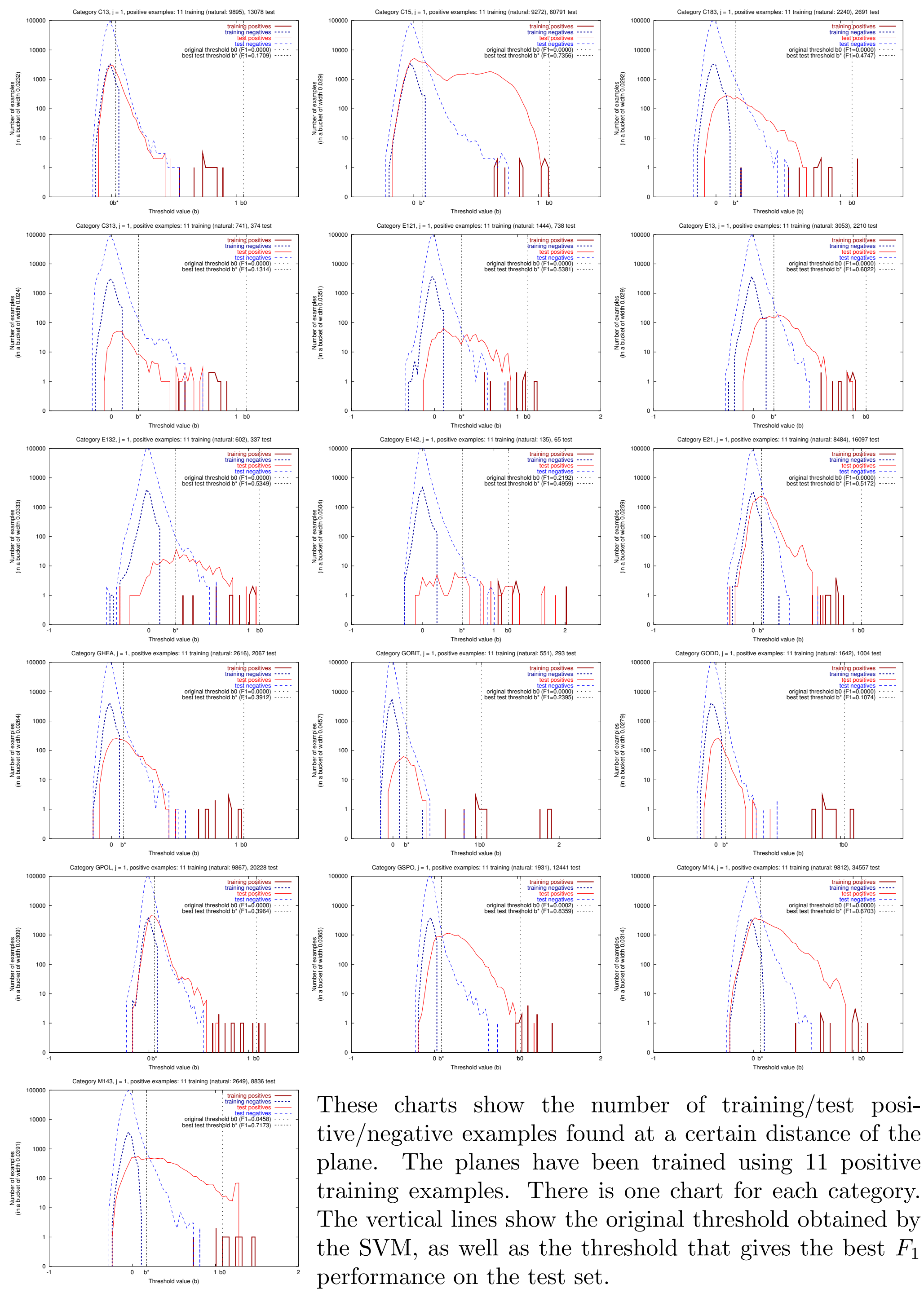
[Reu00] Reuters: *Reuters Corpus, Volume 1: English Language, 1996-08-20 to 1997-08-19.* Available from `http://about.reuters.com/researchandstandards/corpus/`

[Yang01] Yiming Yang: *A study on thresholding strategies for text categorization.* Proc. 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New Orleans, Louisiana, USA, September 9–13, 2001, pp. 137–145.

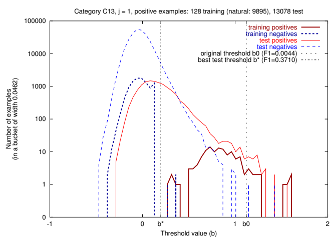
[WP01] Gary M. Weiss, Foster Provost: *The effect of class distribution on classifier learning: An empirical study.* Tech. Rept. ML-TR-44, Dept. of CS, Rutgers University. August 2, 2001.

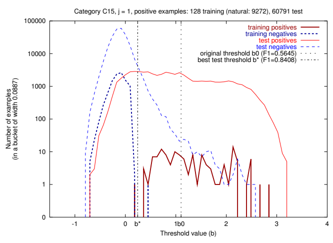
Appendix: per-category charts

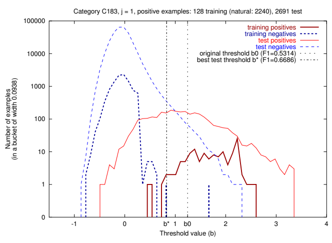
These charts are like those in figures 2 and 3, except that they are given here for all 16 categories rather than for just two.

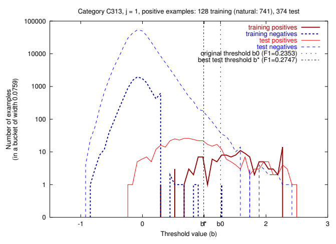


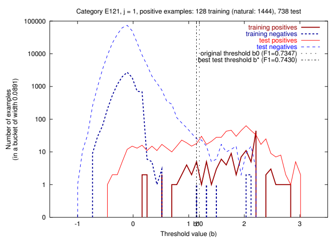
These charts show the number of training/test positive/negative examples found at a certain distance of the plane. The planes have been trained using 11 positive training examples. There is one chart for each category. The vertical lines show the original threshold obtained by the SVM, as well as the threshold that gives the best F_1 performance on the test set.

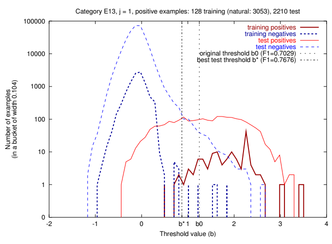


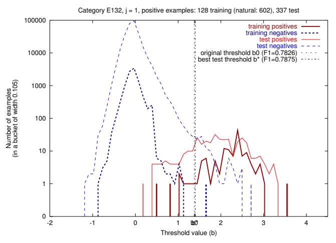


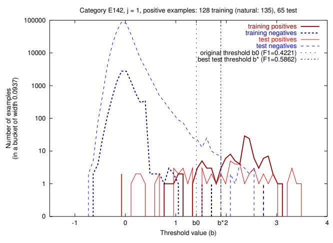


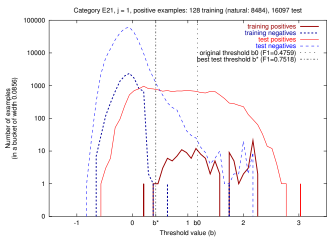


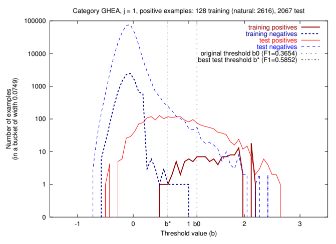


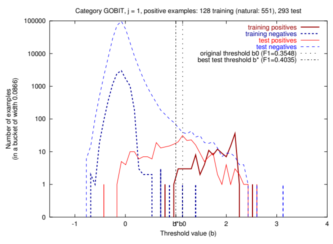


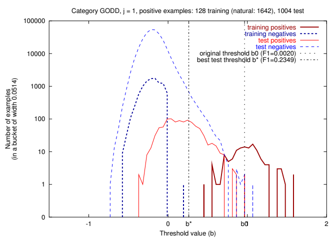


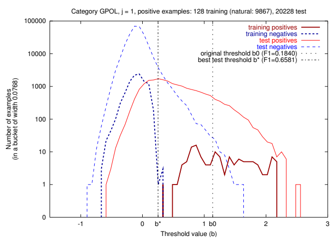


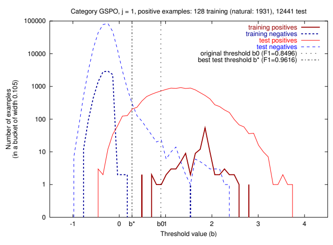


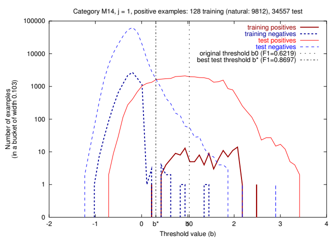


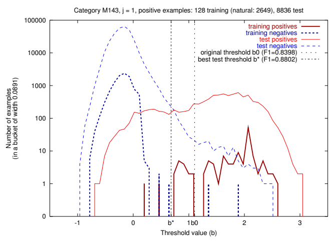




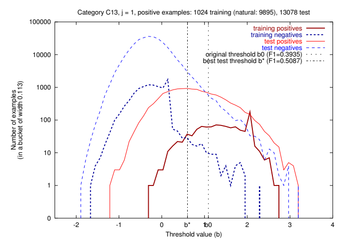


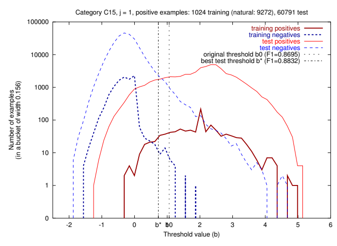


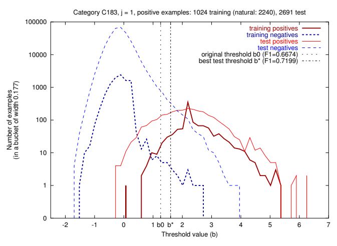


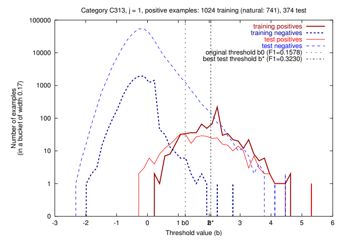


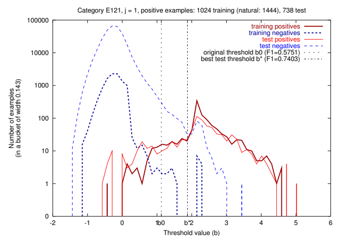
These charts show the number of training/test positive/negative examples found at a certain distance of the plane. The planes have been trained using 128 positive training examples. There is one chart for each category. The vertical lines show the original threshold obtained by the SVM, as well as the threshold that gives the best F_1 performance on the test set.

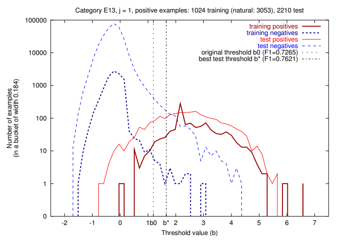


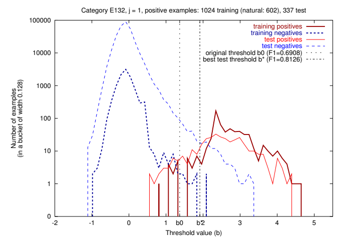


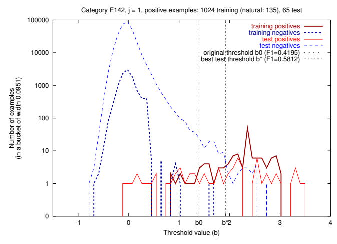


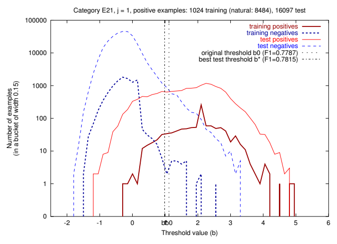


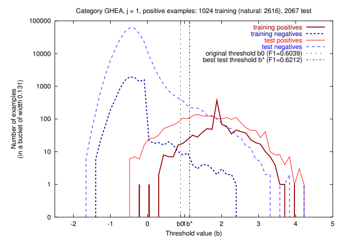


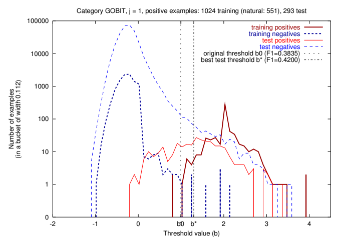


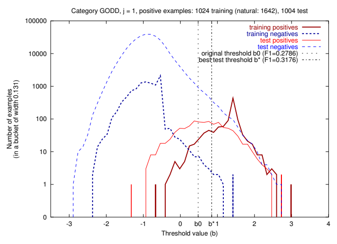


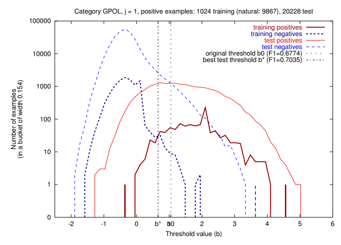


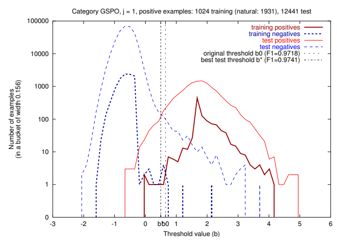


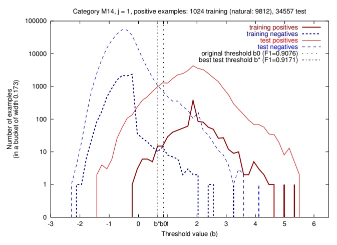


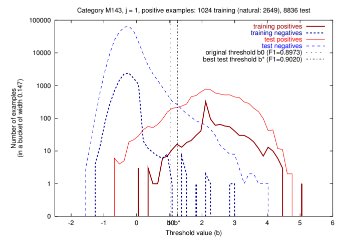




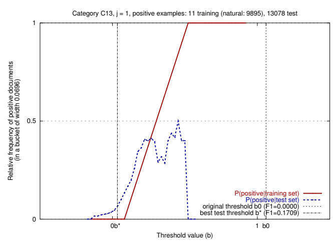


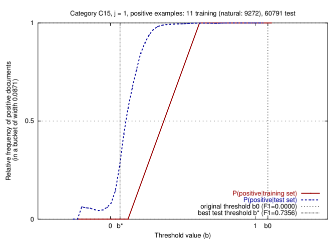


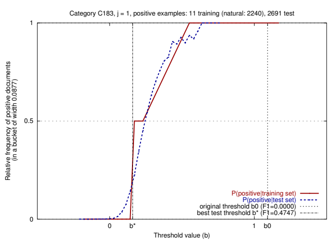


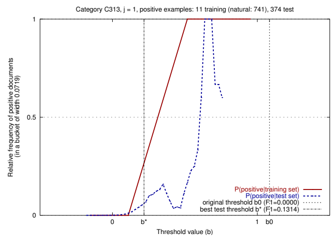


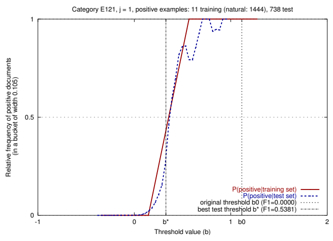
These charts show the number of training/test positive/negative examples found at a certain distance of the plane. The planes have been trained using 1024 positive training examples. There is one chart for each category. The vertical lines show the original threshold obtained by the SVM, as well as the threshold that gives the best F_1 performance on the test set.

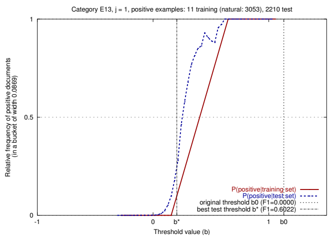


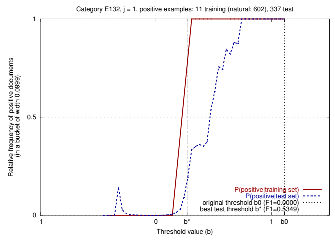


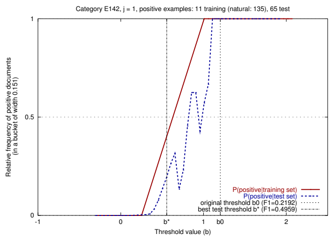


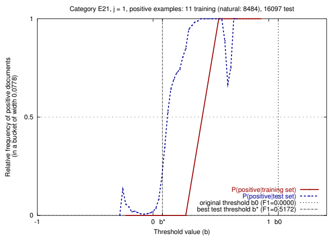


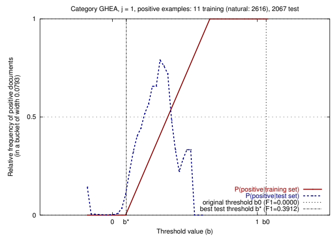


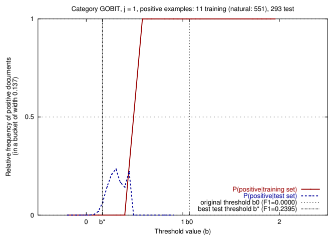


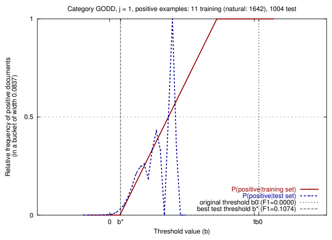


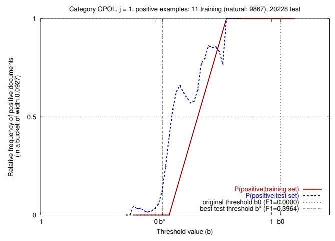


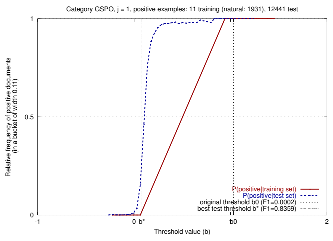


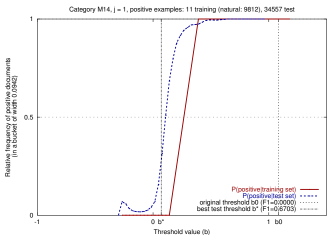


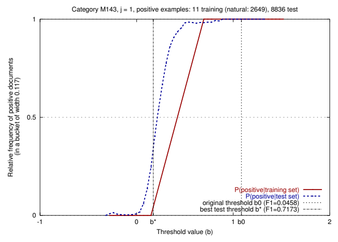




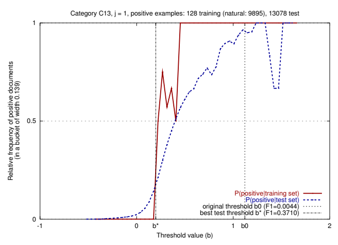


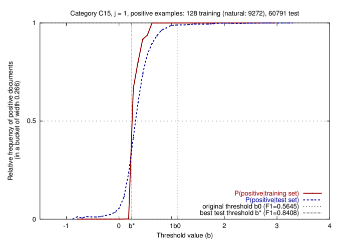


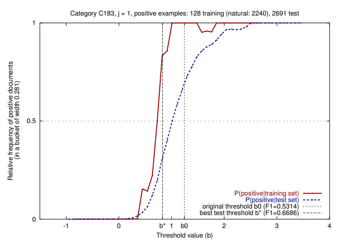


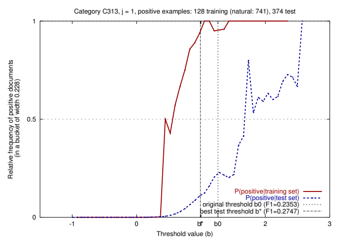


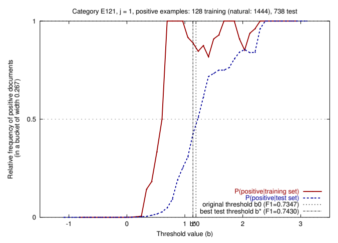
These charts show the proportion of positive examples among all examples that are found at a certain distance from the plane. The planes have been trained using 11 positive training examples. There is one chart for each category. The vertical lines show the original threshold obtained by the SVM, as well as the threshold that gives the best F_1 performance on the test set.

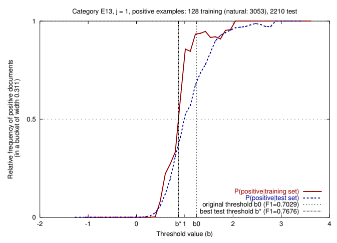


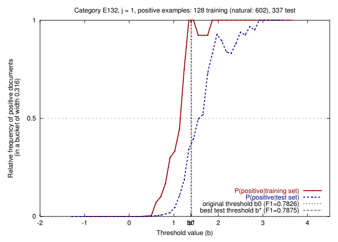


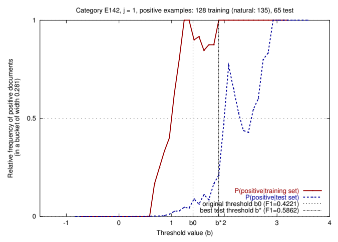


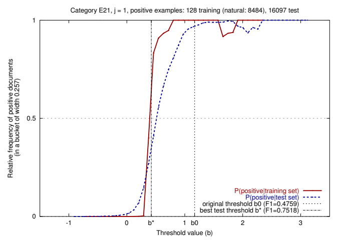


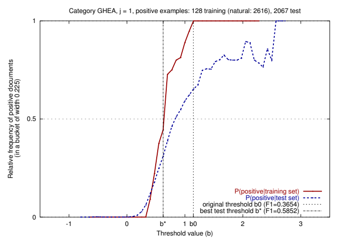


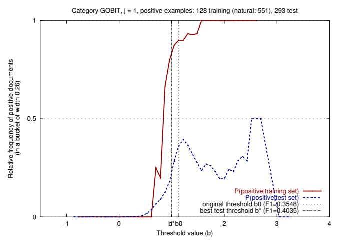


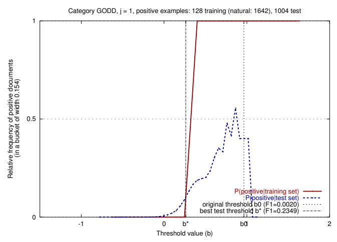


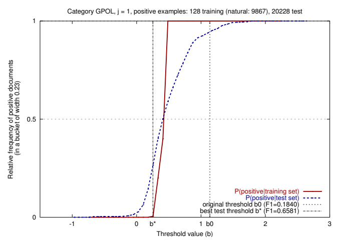


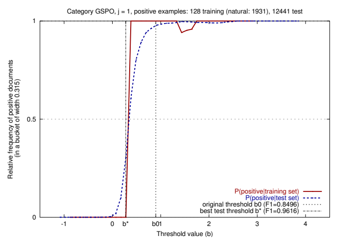


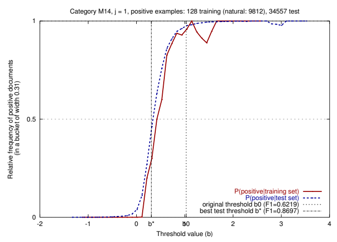


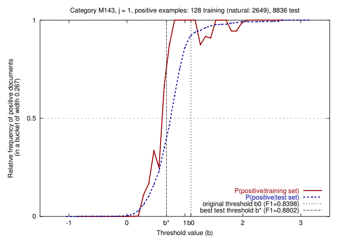




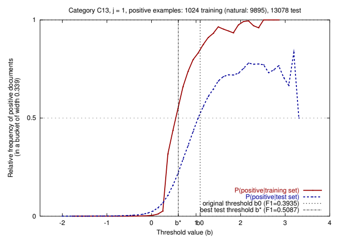


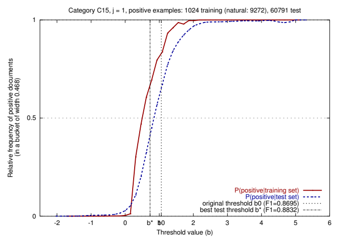


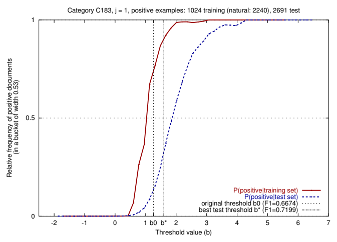


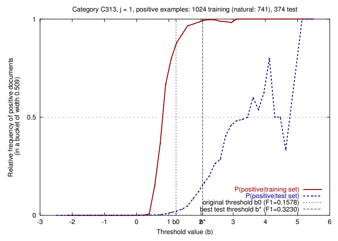


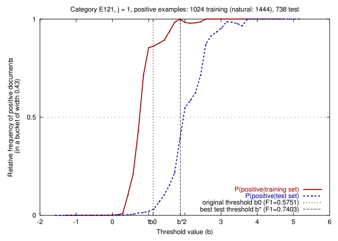
These charts show the proportion of positive examples among all examples that are found at a certain distance from the plane. The planes have been trained using 128 positive training examples. There is one chart for each category. The vertical lines show the original threshold obtained by the SVM, as well as the threshold that gives the best F_1 performance on the test set.

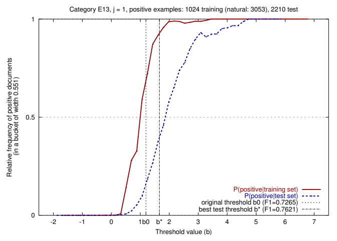


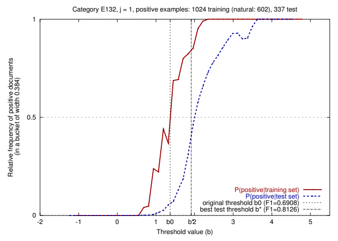


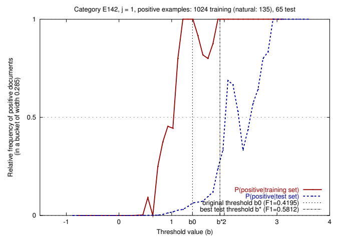


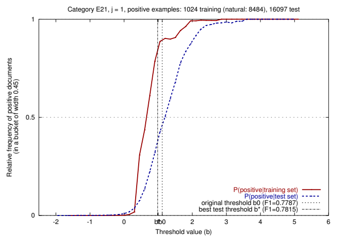


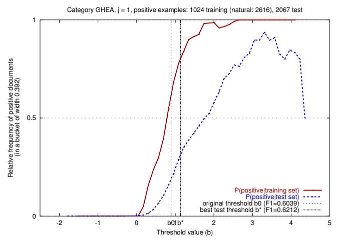


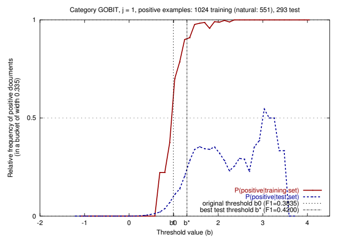


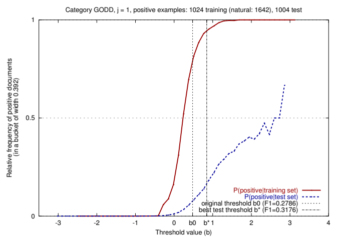


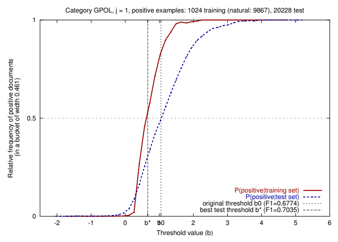


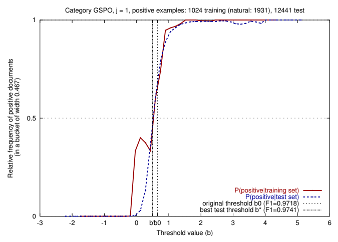


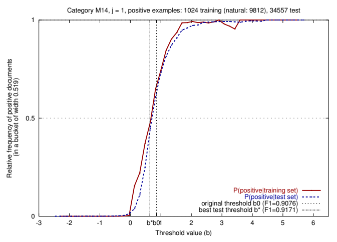


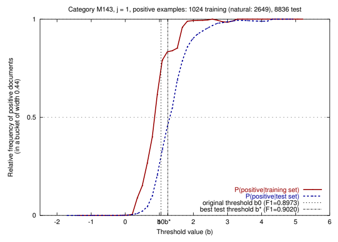












These charts show the proportion of positive examples among all examples that are found at a certain distance from the plane. The planes have been trained using 1024 positive training examples. There is one chart for each category. The vertical lines show the original threshold obtained by the SVM, as well as the threshold that gives the best F_1 performance on the test set.