Learning from Little: Comparison of Classifiers Given Little Training

George Forman and Ira Cohen

Hewlett-Packard Research Laboratories
1501 Page Mill Rd., Palo Alto, CA 94304
{ghforman,icohen}@hpl.hp.com

Abstract. Many real-world machine learning tasks are faced with the problem of small training sets. Additionally, the class distribution of the training set often does not match the target distribution. In this paper we compare the performance of many learning models on a substantial benchmark of binary text classification tasks having small training sets. We vary the training size and class distribution to examine the *learning surface*, as opposed to the traditional *learning curve*. The models tested include various feature selection methods each coupled with four learning algorithms: Support Vector Machines (SVM), Logistic Regression, Naive Bayes, and Multinomial Naive Bayes. Different models excel in different regions of the learning surface, leading to meta-knowledge about which to apply in different situations. This helps guide the researcher and practitioner when facing choices of model and feature selection methods in, for example, information retrieval settings and others.

1 Motivation and Scope

Our goal is to advance the state of meta-knowledge about selecting which learning models to apply in which situations. Consider these four motivations:

1. Information Retrieval: Suppose you are building an advanced search interface. As the user sifts through the each page of ten search results, it trains a classifier on the fly to provide a ranking of the remaining results based on the user's positive or negative indication on each result shown thus far. *Which learning model should you implement to provide greatest precision under the conditions of little training data and a markedly skewed class distribution?*

2. Semi-supervised Learning: When learning from small training sets, it is natural to try to leverage the many unlabeled examples. A common first phase in such algorithms is to train an initial classifier with the little data available and apply it to select additional predicted-positive examples and predicted-negative examples from the unlabeled data to augment the training set before learning the final classifier (e.g. [1]). With a poor choice for the initial learning model, the augmented examples will pollute the training set. *Which learning model is most appropriate for the initial classifier?*

Table 1. Summary of test conditions we vary.

P = 1..40	Positives in training set	Feature Selection Metrics:	
N = 1..200	Negatives in training set	IG	Information Gain
FX= 10..1000	Features selected	BNS	Bi-Normal Separation
Performance Metrics:		Learning Algorithms:	
TP10	True positives in top 10	NB	Naive Bayes
TN100	True negatives in bottom 100	Multi	Multinomial Naive Bayes
F-measure	2×precision×recall÷(precision+recall)	Log	Logistic Regression
	(harmonic avg. of precision & recall)	SVM	Support Vector Machine

3. Real-World Training Sets: In many real-world projects the training set must be built up from scratch over time. The period where there are only a few training examples is especially long if there are many classes, e.g. 30–500. Ideally, one would like to be able train the most effective classifiers at any point. *Which methods are most effective with little training?*

4. Meta-knowledge: Testing all learning models on each new classification task at hand is an agnostic and inefficient route to building high quality classifiers. The research literature must continue to strive to give guidance to the practitioner as to which (few) models are most appropriate in which situations. Furthermore, in the common situation where there is a shortage of training data, cross-validation for model selection can be inappropriate and will likely lead to over-fitting. Instead, one may follow the *a priori* guidance of studies demonstrating that some learning models are superior to others over large benchmarks.

In order to provide such guidance, we compare the performance of many learning models (4 induction algorithms × feature selection variants) on a benchmark of hundreds of binary text classification tasks drawn from various benchmark databases, e.g, Reuters, TREC, and OHSUMED.

To suit the real-world situations we have encountered in industrial practice, we focus on tasks with small training sets and a small proportion of positives in the test distribution. Note that in many situations, esp. information retrieval or fault detection, the ratio of positives and negatives provided in the training set is unlikely to match the target distribution. And so, rather than explore a learning *curve* with matching distributions, we explore the entire learning *surface*, varying the number of positives and negatives in the training set independently of each other (from 1 to 40 positives and 1 to 200 negatives). This contrasts with most machine learning research, which tests under conditions of (stratified) cross-validation or random test/train splits, preserving the distribution.

The learning models we evaluate are the cross product of four popular learning algorithms (Support Vector Machines, Logistic Regression, Naive Bayes, Multinomial Naive Bayes), two highly successful feature selection metrics (Information Gain, Bi-Normal Separation) and seven settings for the number of top-ranked features to select, varying from 10 to 1000.

We examine the results from several perspectives: precision in the top-ranked items, precision for the negative class in the bottom-ranked items, and F-measure, each being appropriate for different situations. For each perspective, we determine which models consistently perform well under varying amounts of training data. For example, Multinomial Naive Bayes coupled with feature selection via Bi-Normal Separation can be closely competitive to SVMs for precision, performing significantly better when there is a scarcity of positive training examples.

The rest of the paper is organized as follows. The remainder of this section puts this study in context to related work. Section 2 details the experiment protocol. Section 3 gives highlights of the results with discussion. Section 4 concludes with implications and future directions.

1.1 Related Work

There have been numerous controlled benchmark studies on the choice of feature selection (e.g. [2]) and learning algorithms (e.g. [3]). Here, we study the cross-product of the two together, and the results bear out that different algorithms call for different feature selection methods in different circumstances. Further, our study examines the results both for maximizing F-measure and for maximizing precision in the top (or bottom) ranked items – metrics used in information retrieval and recommenders.

A great deal of research is based on 5-fold or 10-fold cross-validation, which repeatedly trains on 80–90% of the benchmark dataset. In contrast, our work focuses on learning from *very small* training sets – a phase most training sets go through as they are being built up. Related work by [4] shows that Naive Bayes often surpasses Logistic Regression in this region for UCI data sets. We extend these results to the text domain and to other learning models.

We vary the number of positives and negatives in the training set as independent variables, and examine the learning surface of the performance for each model. This is most akin to the work by [5], in which they studied the effect of varying the training distribution and size (an equivalent parameterization to ours) for the C4.5 decision tree model on a benchmark of UCI data sets, which are not in the text domain and do not require feature selection. They measured performance via accuracy and area under the ROC curve. We measure the performance at the middle (F-measure) and both extreme ends of the ROC curve (precision in the top/bottom scoring items) – metrics better focused on practical application to information retrieval, routing, or semi-supervised learning. For example, when examining search engine results, one cares about precision in the top displayed items more than the ROC ranking of *all* items in the database.

The results of studies such as ours and [5] can be useful to guide research in developing methods for learning under greatly unbalanced class distributions, for which there is a great deal of work [6]. Common methods involve over-sampling the minority class or under-sampling the majority class, thereby manipulating the class distribution in the training set to maximize performance. Our study elucidates the effect this can have on the learning surface for many learning models.

Table 2. Description of benchmark datasets.

Dataset	Cases	Features	Classes	Dataset	Cases	Features	Classes
Whizbang/Cora	1800	5171	36	TREC/fbis	2463	2000	17
OHSUMED/Oh0	1003	3182	10	TREC/La1	3204	31472	6
OHSUMED/Oh5	918	3012	10	TREC/La2	3075	31472	6
OHSUMED/Oh10	1050	3238	10	TREC/tr11	414	6429	9
OHSUMED/Oh15	913	3100	10	TREC/tr12	313	5804	8
OHSUMED/ohscal	11162	11465	10	TREC/tr21	336	7902	6
Reuters/Re0	1504	2886	13	TREC/tr23	204	5832	6
Reuters/Re1	1657	3758	25	TREC/tr31	927	10128	7
WebACE/wap	1560	8460	20	TREC/tr41	878	7454	10
				TREC/tr45	690	8261	10

2 Experiment Protocol

Here we describe how the study was conducted, and as space allows, why certain parameter choices were made. Table 1 shows the parameter settings we varied and defines the abbreviations we use hereafter.

Learning Algorithms: We evaluate the learning algorithms listed in Table 1 using the implementation and default parameters of the WEKA machine learning library (version 3.4) [7].

Naive Bayes is a simple generative model in which the features are assumed to be independent of each other given the class variable. Despite its unrealistic independence assumptions, it has been shown to be very successful in a variety of applications and settings (e.g. [8, 9]). The multinomial variation of Naive Bayes was found by [10] to excel in text classification.

Regularized logistic regression is a commonly used and successful discriminative classifier in which a class a-posteriori probability is estimated from the training data using the logistic function [11]. The WEKA implementation is a multinomial logistic regression model with a ridge estimator, believed to be suitable for small training sets.

The Support Vector Machine, based on risk minimization principles, has proven well equipped in classifying high-dimensional data such as text [2, 12, 3]. We use a linear kernel and varied the complexity constant C; all of the results presented in this paper are for C=1 (WEKA's default value); a discussion of the results when we varied C is given in the discussion section. (The WEKA v3.4 implementation returns effectively boolean output for two-class problems, so we had to modify the code slightly to return an indication of the Euclidean distance from the separating hyperplane. This was essential to get reasonable TP10 and TN100 performance.)

Feature Selection: Feature selection is an important and often under estimated component of the learning model, as it accounts for large variations in performance. In this study we chose to use two feature selection metrics, namely Information Gain (IG) and Bi-Normal Separation (BNS). We chose these two based on a comparative study of a dozen features ranking metrics indicating that

Table 3. Experiment procedure.

```
1 For each of the 19 multi-class dataset files:
2  For each of its classes C
       where there are >=50 positives (C) and >=250 negatives (not C):
3   For each of 5 random split seeds:
4    Randomly select 40 positives and 200 negatives for set MaxTrain,
         leaving the remaining cases in the testing set.
5    For P = 1..40:
6     For N = 1..200:
7      Select as the training set the first P positives
           and the first N negatives from MaxTrain.
8      // Task established.  Model parameters follow. //
9      For each feature selection metric IG, BNS:
10       Rank the features according to the feature selection metric
              applied to the training set only.
11       For FX = 10,20,50,100,200,500,1000:
12         Select the top FX features.
13         For each algorithm SVM, Log, NB, Multi:
14           Train on the training set of P positives and N negatives.
15           Score all items in the testing set.
16           For each performance measure TP10, F-measure, TN100:
17             Measure performance.
```

these two are top performers for SVM; classifiers learned with features selected via IG tended to have better precision; whereas BNS proved better for recall, overall improving F-measure substantially [2]. We expected IG to be superior for the goal of precision in the top ten.

Benchmark Data Sets: We used the prepared benchmark datasets available from [2, 13], which stem originally from benchmarks such as Reuters, TREC, and OHSUMED. They comprise 19 text datasets, each case assigned to a single class. From these we generate many binary classification tasks identifying one class as positive vs. all other classes. Over all such binary tasks, the median percentage of positives is ∼5%. We use binary features, representing whether the (stemmed) word appears at all in the document. The data are described briefly in Table 2. For more details, see Appendix A of [2].

Experiment Procedure: The experiment procedure is given in Table 3 as pseudo-code. Its execution consumed ∼5 years of computation time, run on hundreds of CPUs in the HP Utility Data Center. Overall there are 153 2-class data sets which are randomly split five times yielding 765 binary tasks for each P and N. The condition on the second loop ensures that there are 40 positives available for training plus at least 10 others in the test set (likewise, 200 negatives for training and at least 50 for testing). Importantly, feature selection depends only on the training set, and does not leak information from the test set.

Because the order of items in MaxTrain is random, later selecting the *first* P or N cases amounts to random selection; this also means that the performance for (P=3,N=8) vs. (P=**4**,N=8) represents the same test set, and the same train-

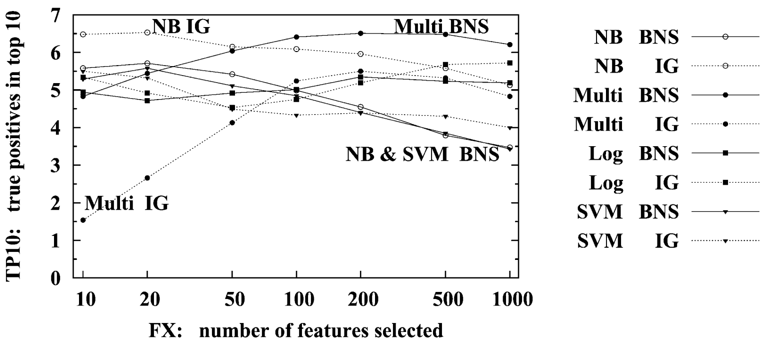

Fig. 1. Average TP10 performance for each learning model given P=5 positives and N=200 negatives, varying the number of features selected FX. *(For more readable color graphs, see http://www.hpl.hp.com/techreports/2004/HPL-2004-19R1.html)*

ing set with one additional positive added at random, i.e. they may be validly interpreted as steps on a learning curve.

Performance Metrics: We analyze the results independently for each of the following metrics:

1. The TP10 metric is the number of true positives found in the 10 test cases that are predicted most strongly by the classifier to be positive.

2. The TN100 metric is the number of true negatives found in the 100 test cases most predicted to be negative. Because of the rarity of positives in the benchmark tasks, scores in the upper 90's are common (TN10 is nearly always 10, hence the use of TN100).

3. F-measure is the harmonic average of precision and recall for the positive class. It is superior to grading classifiers based on accuracy (or error rate) when the class distribution is skewed.

Different performance metrics are appropriate in different circumstances. For recommendation systems and information retrieval settings, where results are displayed to users incrementally with the most relevant first, the metric TP10 is most appropriate. It represents the precision of the first page of results displayed. For information filtering or document routing, one cares about both the precision and the recall of the individual hard classification decisions taken by the classifier. F-measure is the metric of choice for considering both together. For semi-supervised learning settings where additional positive (negative) cases are sought in the unlabeled data to expand the training set, the appropriate metric to consider is TP10 (TN100). Maximum precision is called for in this situation, or else the heuristically extended training set will be polluted with noise labels.

3 Experiment Results

We begin by examining an example set of results for the average TP10 performance over all benchmark tasks, where the training set has P=5 positives and N=200 negatives. See Fig.1. We vary the number of features selected along the logarithmic x-axis.

We make several observations. First, two models rise above the others by ∼15% here: Naive Bayes using IG with a few features, tied with Multinomial Naive Bayes using BNS with several hundred features. Second, note that using the opposite feature selection metric for each of these Bayes models hurts their performance substantially, as does using the 'wrong' number of features. This illustrates that the choices in feature selection and the induction algorithms are interdependent and are best studied together. Third, SVM, which is known for performing well in text classification, is consistently inferior in this situation with positives greatly under represented in the training set, as we shall see.

Condensing FX Dimension: These results are for only a single value of P and N. Given the high dimensionality of the results, we condense the FX dimension hereafter, presenting only the best performance obtained over any choice for FX. Because this maximum is chosen based on the *test* results, this represents an upper bound on what could be achieved by a method that attempts to select an optimal value of FX based on the training set alone. Condensing the FX dimension allows us to expose the differences in performance depending on the learning algorithm and feature selection metric. Furthermore, for the practitioner, it is typically easy to vary the FX parameter, but harder to change the implemented algorithm or feature selection metric.

Visualization: With this simplification, we can vary the number of positives P and negatives N in the training set to derive a *learning surface* for each of the 8 combinations of algorithm and feature selection metric. We begin by illustrating a 3-D perspective in Fig.2a showing the learning surfaces for just three learning models: Multinomial Naive Bayes, SVM and Logistic Regression, each with BNS feature selection. The performance measure here is the average number of true positives identified in the top 10 (TP10). From this visualization we see that Log-BNS is dominated over the entire region, and that between the remaining two models, there are consistent regions where each performs best. In particular, the SVM model substantially under performs the Mulinomial Naive Bayes when there are very few positives. The two are competitive with many positives and negatives.

This 3-D perspective visualization becomes difficult if we display the surfaces for each of the eight learning models. To resolve this, we plot all surfaces together and then view the plot from directly above, yielding the topo-map visualization shown in Fig.2b. This reveals only the best performing model in each region. The visualization also indicates the absolute performance (z-axis) via isoclines, like a geographical topo-map.

While a topo-map shows the model that performed *best* for each region, it does not make clear *by how much* it beat competing models. For this, we show two z-axis cross-sections of the map near its left and right edges. Figures 2c–d fix the number of negatives at N=10 and N=200, comparing the performance of all eight models as we vary the number of positives P. Recall that by design the test set for each benchmark task is fixed as we vary P and N, so that we may view these as learning curves as we add random positive training examples.

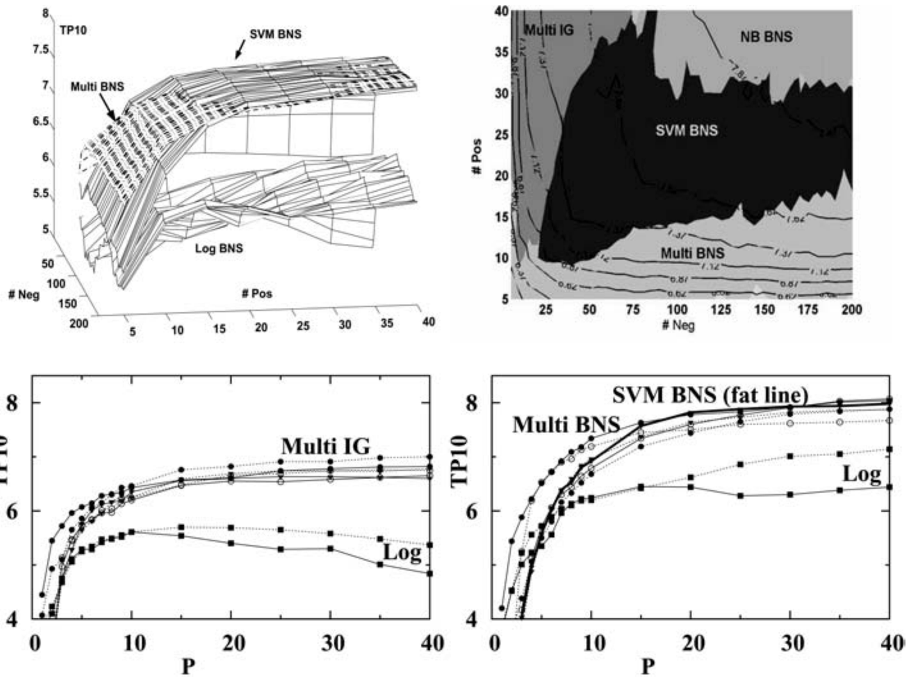

Fig. 2. TP10 performance. (a) Learning surfaces for three models (SVM-BNS, Multi-BNS and Log-BNS) as we vary the number of positives and negatives in the training set. (b) Topo-map of best models – 3D surfaces of all models viewed from above. Isoclines show identical TP10 performance. (c) Cross-section at N=10 negatives, varying P positives. (d) Cross-section at N=200 negatives, varying P positives. (Legend in Fig.1.)

TP10 Results: Our initial impetus for this study was to determine which learning models yield the best precision in the top 10 given little training data. We find that the answer varies as we vary the number of positives and negatives, but that there are consistent regions where certain models excel. This yields meta-knowledge about when to apply different classifiers. See Fig.2b. We observe that BNS is generally the stronger feature selection metric, except roughly where the number of positives exceeds the number of negatives in the training set along the y-axis, where Multi-IG dominates.

Recall that the test distribution has only a few percent positives, as is common in many tasks. So, a random sample would fall in the region near the x-axis having <10% positives where Multi-BNS dominates. Observe by the horizontal isoclines in most of this region that there is little to no performance improvement for increasing the number of negatives. In this region, the best action one can take to rapidly improve performance is to provide more *positive* training examples (and similarly near the y-axis).

The isoclines show that the best TP10 performance overall can be had by providing a training set that has an over representation of positives, say P>30

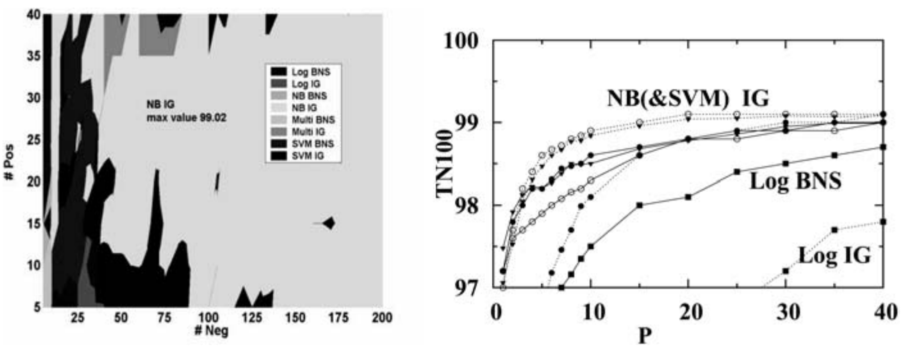

Fig. 3. TN100 performance. (a) Topo-map of best models for TN100. (b) Cross-section given N=200 negatives, varying P positives. (Legend in Fig.1.)

and N>100. Here, NB-BNS dominates, but we can see by the mottling of colors in this region that Multi-BNS is closely competitive. More generally, the cross-section views in Figs.2c–d allow us to see how competitive the remaining learning models are. In Fig.2c, we can now see that with enough positives, Multi-BNS, SVM-BNS and NB-BNS are all competitive, but in the region with <=15 positives, Multi-BNS stands out substantially.

TN100 Results: We also determined which learning model yields the most true negatives in the bottom-ranked list of 100 test items. Although no model is dominant everywhere, NB-IG is a consistent performer, as shown in Fig.3a, especially with many negatives. In the N=200 cross-section shown in Fig.3b, we see that it and SVM-IG substantially outperform the other models. Overall, performance is very high (over 98% precision) – a fact that is not surprising considering that there are many more negatives than positives in the test sets. Unfortunately, no further performance improvement is attained after ~20 positives (though one may speculate for P>40).

F-Measure Results: Next we compare the learning models by F-measure. Figure 4a shows the topo-map of the best performing models over various regions. Observing the isoclines, the greatest performance achieved is by SVM-BNS, with appropriate oversampling of positives. If random sampling from the test distribution, most labels found will be negative, and put us in the region of poor F-measure performance along the x-axis, where NB-IG dominates. We see in the cross-section in Fig.4b with P=5 fixed and varying the number of negatives N, that NB-IG dominates all other models by a wide margin and that its performance plateaus for N>=30 whereas most other models experience declining performance with increasing negatives. Likewise, in Fig.4c with N=10 fixed, the performance of all models declines as we increase the number of positives. Finally, in Fig.4d with N=200 fixed, we see that substantial gains are had by SVM with either feature selection metric as we obtain many positive training examples. With few positives, such as obtained by random sampling, SVM becomes greatly inferior to NB-IG or Multi-BNS.

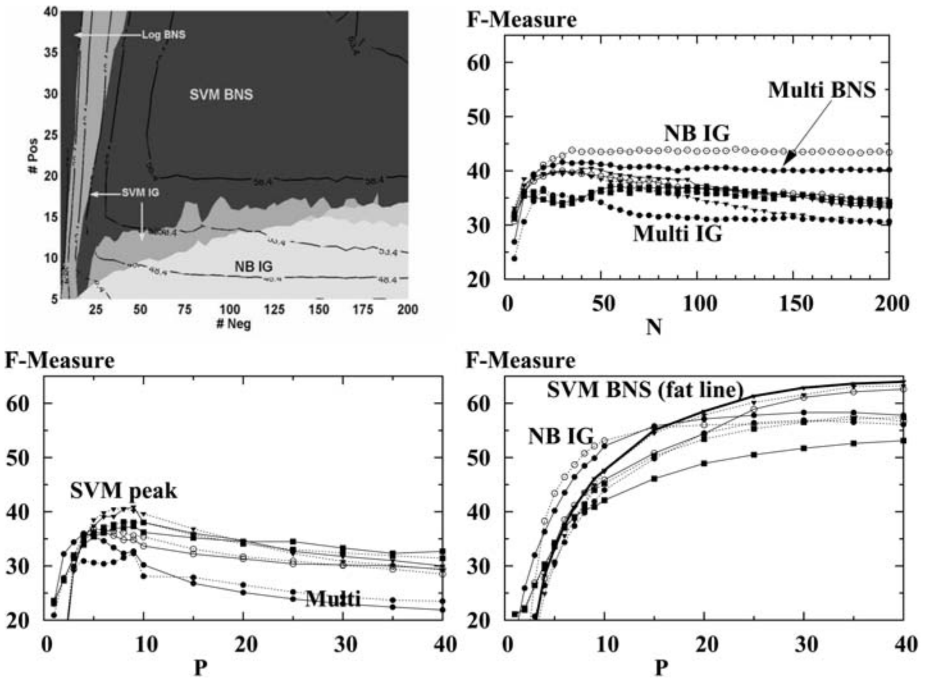

Fig. 4. F-measure performance. (a) Topo-map of best models for F-measure. (b) Cross-section at P=5 positives, varying N negatives. (c) Cross-section at N=10 negatives, varying P positives. (d) Cross-section at N=200 negatives, varying P positives. (Legend in Fig.1.)

Observing the isoclines in Fig.4a, the best approach for maximizing F-measure at all times while building a training set incrementally from scratch is to use SVM (-BNS or else -IG) and keep the class distribution of the training set at roughly 20% positives by some non-random sampling method.

3.1 Discussion

Generalizing from the notion of a learning curve to a learning *surface* proves to be a useful tool for gaining insightful meta-knowledge about regions of classifier performance. We saw that particular classifiers excel in different regions; this may be constructive advice to practitioners who know in which region they are operating. The learning surface results also highlight that performance can be greatly improved by non-random sampling that somewhat favors the minority class on tasks with skewed class distributions (however, balancing P=N is unfavorable). This is practical for many real-world situations, and may substantially reduce training costs to obtain satisfactory performance.

Because of the ubiquitous research practices of random sampling and cross-validation, however, researchers routinely work with a training set that matches the distribution of the test set. This can mask the full potential of classifiers under study. Furthermore, it hides from view the research opportunity to develop classifiers that are less sensitive to the training distribution. This would be useful practically since in industrial classification problems the class distribution of the training set is often varied, unknown in advance, and does not match the testing or target distributions, which may vary over time.

Naive Bayes models have an explicit parameter reflecting the class distribution, which is ususally set to the distribution of the training set. Hence, these models are frequently said to be sensitive to the training distribution. The empirical evidence for TP10, TN100 and F-measure shows that Naive Bayes models are often relatively insensitive to a shift in training distribution (consistent with the theoretical results by Elkan [14]), and surpass SVM when there is a shortage of positives or negatives.

Although the results showed that SVM excels overall for TP10 and F-measure if the training class distribution is ideal, SVM proves to be highly sensitive to the training distribution. This is surprising given that SVM is popularly believed to be resilient to variations in class distribution by its discriminative nature rather than being density-based.

This raises the question of varying SVM's C parameter to try to reduce its sensitivity to the training class distribution. To study this, we replicated the entire experiment protocol for eight values of C ranging from 0.001 to 5.0. For F-measure, other values of C substantially hurt SVM's performance in the regions in Fig.4a along the x- and y-axes where SVM was surpassed by other models. In the large region where SVM already dominates, however, some values of C increased performance – at the cost of making the learning surface *more* sensitive to the training distribution. For P=40 and N=200, F-measure can be increased by as much as ∼5% when C=0.1, but then its performance declines drastically if the number of positives is reduced below 20.

The additional results were similar for TP10. Refer to the topo-map in Fig.2b. No value of C made SVM surpass the performance of Multi-BNS in the region along the x-axis. At the upper right, performance could be increased by fortunate choices for C (by as much as ∼3% for P=40 and N=200, which exceeds the performance of NB-BNS slightly). This boost comes at the cost of very poor performance in the lower region along the x-axis. Finally, some values of C made SVM competitive with Multi-IG in the top left, but again made performance much worse as we decrease the number of positives.

Overall, varying C does not lead to fundamentally different conclusions about the regions of performance. It does not address the issue of making the choice of classifier insensitive to the operating region. Furthermore, in regions where performance can be improved, it remains to be seen whether the optimal C value can be determined automatically via cross-validation. With small training sets, such cross-validation may only lead to over fitting the training set, without practical improvement.

4 Summary

This paper compared the performance of different classifiers with settings often encountered in real situations: small training sets especially scarce in positive examples, different test and train class distributions, and skewed distributions of positive and negative examples. Visualizing the performance of the different classifiers using the learning surfaces provides meta-information about which models are consistent performers under which conditions. The results showed that feature selection should not be decoupled from the model selection task, as different combinations are best for different regions of the learning surface.

Future work potentially includes expanding the parameters, classifiers and datasets studied, and validating that the meta-knowledge successfully transfers to other text (or non-text) classification tasks.

References

1. Liu, B., Dai, Y., Li, X., Lee, W.S., Yu, P.S.: Building text classifiers using positive and unlabeled examples. In: Intl. Conf. on Data Mining. (2003) 179–186
2. Forman, G.: An extensive empirical study of feature selection metrics for text classification. Journal of Machine Learning Research **3** (2003) 1289–1305
3. Yang, Y., Liu, X.: A re-examination of text categorization methods. In: ACM SIGIR Conf. on Research and Development in Information Retrieval. (1999) 42–49
4. Ng, A.Y., Jordan, M.I.: On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. In: Neural Information Processing Systems: Natural and Synthetic. (2001) 841–848
5. Weiss, G.M., Provost, F.: Learning when training data are costly: The effect of class distribution on tree induction. Journal of Artificial Intelligence Research **19** (2003) 315–354
6. Japkowicz, N., Holte, R.C., Ling, C.X., Matwin, S., eds.: AAAI Workshop: Learning from Imbalanced Datasets, TR WS-00-05, AAAI Press (2000)
7. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools with Java implementations. Morgan Kaufmann, San Francisco (2000)
8. Duda, R.O., E.Hart, P.: Pattern Classification and Scene Analysis. John Wiley and Sons (1973)
9. Domingos, P., Pazzani, M.: Beyond independence: conditions for the optimality of the simple Bayesian classifier. In: Proc. 13th International Conference on Machine Learning. (1996) 105–112
10. McCallum, A., Nigam, K.: A comparison of event models for naive Bayes text classification. In: AAAI-98 Workshop on Learning for Text Categorization. (1998)
11. le Cessie, S., van Houwelingen, J.: Ridge estimators in logistic regression. Applied Statistics **41** (1992) 191–201
12. Joachims, T.: Text categorization with support vector machines: Learning with many relevant features. In: European Conf. on Machine Learning. (1998) 137–142
13. Han, E., Karypis, G.: Centroid-based document classification: Analysis & experimental results. In: Conference on Principles of Data Mining and Knowledge Discovery. (2000) 424–431
14. Elkan, C.: The foundations of cost-sensitive learning. In: International Joint Conference on Artificial Intelligence. (2001) 973–978