1-1-2015

Applications Of Machine Learning In Biology And Medicine

Saied Haidarian Shahri
Wayne State University,

APPLICATIONS OF MACHINE LEARNING IN BIOLOGY AND MEDICINE

by

SAIED HAIDARIAN SHAHRI

DISSERTATION

Submitted to the Graduate School

of Wayne State University,

Detroit, Michigan

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

2015

MAJOR: COMPUTER SCIENCE

Approved by:

_____

Advisor Date

_____

_____

_____

_____

DEDICATION

To

my parents Mohammad and Maryam, and my brother Hamid

without whom none of this would have been possible!

ACKNOWLEDGMENTS

I feel delighted to express my gratitude to all the individuals who encouraged and supported me throughout my graduate career.

First and foremost, I thank my graduate advisor, Professor Sorin Draghici. Most of what I have learned about how to ask the right questions and put it down in writing, I learned from him. His unrivaled commitment and endless enthusiasm in research will always be an inspiration.

I thank Dr. Alper Murat for his wonderful course on nonlinear optimization, which was the starting point on some of my later research, and also for being on my graduate committee. I thank the rest of my graduate committee, Dr. Loren Schwiebert and Dr. Nathan Fisher, for their reviews and suggestions. I thoroughly enjoyed Dr. Schwiebert's course on Design and Analysis of Algorithms, and his masterful skill in breaking down and simplifying difficult problems.

Many thanks to the members of the Intelligent Systems and Bioinformatics Laboratory (ISBL) who made this journey all the more enjoyable. We truly had many adventures together. Thank you Calin Voichita, for all the collaborations we had together. Thank you Michele Donato, for reviewing and correcting my writings. I learned a lot from you on how to handle difficult situations and where to find authentic Italian cuisine. Thank you Rebecca Tagett, for your tremendous and detailed knowledge of biology and your willingness to share it anytime.

Finally, it goes without saying that none of this would have been possible without the continuous and loving support of my family.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 INTRODUCTION

Machine learning as a field is defined to be the set of computational algorithms that improve their performance by assimilating data. Since modern computation and storage becomes cheap at a very fast rate, both knowledge discovery in scientific endeavors and developing cost effective solutions in industry, increasingly rely on machine learning. As such, the field as a whole has found applications in many diverse disciplines from robotics and communication in engineering to economics and finance, and also biology and medicine.

It should not come as a surprise that many popular methods in use today have completely different origins. Modern machine learning methods usually find their roots in statistics, optimization and computer science. In some instances the same method can be analyzed and interpreted from different views. Despite this heterogeneity, different methods can be divided into standard tasks, such as supervised, unsupervised, semi-supervised and reinforcement learning.

Supervised learning is the setting where the available data is provided with predetermined outcomes, and the goal is to learn the best model that can predict the outcome in some measurable sense. When the outcome is discrete, the task is known as classification, and when it is continuous the task is called regression. Due to it's usefulness, this setting is the most studied task in machine learning and prevalent in many domains such as speech, vision and text. Unsupervised learning is the setting where the data is provided without any labels, and the goal is to organize the data in groups or find some structure in it. Clustering, density estimation and community detection are instances of this setting. Semi-supervised learning as the name implies is when the data is partially labeled, and unlabeled data is usually much more abundant than labeled data. Finally, reinforcement learning or credit assignment problem is the setting where the agent receives delayed reward or punishment for any action, and the goal is to learn a policy that maximizes the reward.

Although machine learning as a field can be formalized as methods trying to solve certain

standard tasks, applying these tasks on datasets from different fields comes with certain caveats, and sometimes is fraught with challenges. In this thesis, we develop general procedures and novel solutions, dealing with practical problems that arise when modeling biological and medical data.

Cost sensitive learning is an important area of research in machine learning which addresses the widespread and practical problem of dealing with different costs during the learning and deployment of classification algorithms. There are various ways to assign costs to the learning algorithms. To name a few misclassification costs and data acquisition costs are two fundamental notions of cost assignment. In many applications such as credit fraud detection, network intrusion and specifically medical diagnosis domains, prior class distributions are highly skewed, which makes the training examples very much unbalanced. Combining this with uneven misclassification costs renders standard machine learning approaches useless in learning an acceptable decision function. In Chapter 2 we experimentally show the benefits and shortcomings of various methods that convert cost blind learning algorithms to cost sensitive ones. The empirical results are produced using controlled experiments with large amounts of data to ensure validity.

Using the results and best practices found for cost sensitive learning, we design and develop a machine learning approach to ontology mapping. Due to the nature of ontology mapping problem and the *human in the loop* requirement, the ontology mapping framework should consider different costs involved. Although machine learning techniques have been used earlier in many semantic integration approaches, dependence on precision recall curves to preset the weights and thresholds of the learning systems has been a serious bottleneck. By recasting the mapping problem to a classification problem we try to automate this step and develop a robust and extendable meta learning algorithm. The implication is that we can now extend the same method to map the ontology pairs with different similarity measures which might not be specialized for the specific domain, yet obtain results comparable to the state of the art mapping algorithms that exploit machine learning methods. Interestingly we

see that as the similarity measures are diluted, our approach performs significantly better for unbalanced classes. We test our approach using several similarity measures and two real world ontologies, and the test results we discuss validate our claim. We also present a discussion on the benefits of the proposed meta learning algorithm.

Next, we present a novel approach to deal with uncertainty in classification when costs are unknown or otherwise hard to assign. Support Vector Machines (SVM) are considered to be among the most successful approaches for classification. However prediction of instances near the decision boundary depends more on the specific parameter selection or noise in data, rather than a clear difference in features. In many applications such as medical diagnosis, these regions should be labeled as uncertain rather than assigned to any particular class. Furthermore, instances may belong to *novel disease subtypes* that are not from any previously known class. In such applications, declining to make a prediction could be beneficial when more powerful but expensive tests are available. In Chapter 4, we focus on finding the regions in which there is a high uncertainty in predictions. A natural way for determining these regions when related costs are known is using posterior probabilities. However, in practice, the true costs are rarely available, and large margin classifiers do not compute posterior probabilities. This makes it difficult to combine the cost with the posterior and identify the uncertain regions. Previous methods have shown that thresholding SVM output can be effective, however they are limited by the requirement of a predefined *extended* cost function, i.e. an *a priori* definition for the misclassification and rejection costs, which is even harder to define in practice. The original contribution of this part is an approach for automatically finding uncertain regions without the need to *pre-define* relative misclassification and rejection costs. We develop a novel approach for optimal selection of the threshold and show its successful application on three biological and medical datasets.

The last part of this thesis provides novel solutions for handling high dimensional data usually found in biological and medical experiments. Technological advances have revolutionized data collection and processing. Although high-dimensional data is ubiquitously found

in many disciplines, current life science research almost always involves high-dimensional genomics/proteomics data. The "omics" data provide a wealth of information and have changed the research landscape in biology and medicine.

However, these data are plagued with noise, redundancy and collinearity, which makes the discovery process very difficult and costly. Any method that can accurately detect irrelevant and noisy variables in omics data would be highly valuable. In Chapter 5, we present Robust Feature Selection (RFS), a randomized feature selection approach dedicated to low-sample high-dimensional data. RFS combines an embedded feature selection method with a randomization procedure for stability.

Recent advances in sparse recovery and estimation methods have provided efficient and asymptotically consistent feature selection algorithms. However, these methods lack finite sample error control due to instability. Furthermore, the chances of correct recovery diminish with more collinearity among features. To overcome these difficulties, RFS uses a randomization procedure to provide an accurate and stable feature selection method. We thoroughly evaluate RFS by comparing it to a number of popular univariate and multivariate feature selection methods and show marked prediction accuracy improvement of a diagnostic signature, while preserving a good stability.

CHAPTER 2 COST SENSITIVE LEARNING AND THE CLASS IMBALANCE PROBLEM

Numerous learning algorithms have been developed since the appearance of modern computation. Learning algorithms can be defined as programs that improve their performance according to one or more criteria through assimilation of data/experience. Classification is a branch of learning algorithms that tries to classify a set of instances into a finite set of discrete categories. Classification algorithms are usually designed to optimize a certain loss function and in most cases this loss function is their expected number of errors. However, in real world application scenarios misclassification errors are rarely of equal cost. To make things a little worse, it is usually the case that costly errors happens to be on classes which we have little information to work with. That is, errors made upon the classes that we have few training examples are usually the costly ones. In the machine learning community, the first problem is called *cost sensitive learning*, and the second one is called the *class imbalance problem*. There are many types of costs that can be associated to learning and deployment of a particular algorithm [106]. Misclassification cost, data acquisition cost, active learning cost and computation cost are to name a few, however, our focus in this paper is only on misclassification cost. For other work related to different types of cost please refer to [105, 121, 69].

2.1 Introduction

In this work we review different ways of developing cost sensitive classifiers and how much these approaches are influenced by the class imbalance problem. Unless the prior class distributions are not highly skewed, the class imbalance problem is not always a problem per se, since some learning algorithms do consider the prior class distributions in their learning

process [120]. However, combining this with asymmetric misclassification costs can make a classifier completely useless [70].

Whenever there is sufficient credence to define costs, they are dealt with using three basic approaches. The first approach uses Bayes risk theory to minimize the expected cost [27, 121]. In this approach a classifier is trained to output class conditional probability estimates, which is coupled with the cost to define decision boundaries with minimum expected cost. The advantage of this approach is that costs need not be known at the training time. Thus, we can come up with different decisions simply by changing the cost matrix at deployment time. However, for this to happen, we need to have accurate conditional probability estimates, which are usually harder to obtain than the classification or discriminant function. Probability Estimation Trees and Lazy Option Trees [71], Bagging [17], and Calibrating [83, 122] are among the few methods to obtain accurate probability estimates. The second approach is that of making specific classifiers cost sensitive. Cost sensitive versions of Naive Bayes, Decision Trees [64, 15, 103], Support Vector Machines [47] and Neural Networks [49] fall into this category. The third approach is algorithm independent, where it alters the distribution of the training examples either by reweighting each instance or by resampling the whole training set. Some algorithm specific approaches fall into this category as well. The simplest case is random oversampling of the minority class and/or random undersampling of the majority class [72]. Experiments done with decision trees show that random oversampling usually defeats the validation process and thus makes the model overfit the data, but random undersampling helps improve the correct classification rate to some extent [30]. To help overcome the problem of creating independent samples from the altered distribution while still avoiding overfitting, instead of bootstrapping, rejection sampling methods has been successfully used and also further improved by bagging several trained models over the resampled data. This method is known as Costing [123]. There are also variants of resampling that make informed oversampling by employing k-nearest neighbors as a local estimator combined with random undersampling [20]. This method proves successful in var-

ious cases, however sometimes over-generalizes the minority class especially when the data contains a fair amount of outliers. Some attempts have been made to solve this problem by identifying the so called noise and borderline regions of the space to generate synthetic samples [53].

In the rest of Chapter 2, we compare direct cost sensitive classifiers to wrapper approaches in terms of their success in handling different levels of imbalanced data and asymmetric costs. As proven in [4], theoretically we can use a technique called Data Space Expansion to convert any multiclass classification problem to its binary counterpart. Although this would be practical for multi-class problems with few classes, methods that consider large number of classes is beyond the scope of this work. Thus, from now on without loss of generality, we will only consider binary classification problems, i.e. classification problems with only two classes. First we discuss the basics of cost matrices and selecting the base rate. Then we go on to describing some of the resampling and reweighting approaches. Finally we compare these methods experimentally by applying them to several well-known classification algorithms. The large amounts of data and cross validation over a wide range of parameter settings ensure good confidence in the reported results.

2.2 Basics of Cost Sensitive Learning

We formally define a cost sensitive learning algorithm, one that minimizes the expected cost. Costs can be modeled in three major ways; namely, example independent cost, example dependent cost [121] and attribute acquisition cost [69]. The first two methods model misclassification cost while the latter models data acquisition cost. Example dependent costs are useful when misclassifying different examples of the same class would incur different costs. This is only possible in rare cases, such as credit card fraud detection or gathering donation through mail solicitation, when the actual case dependent cost or benefit can be easily obtained. The more widely used cost model that is also used in this paper is the example

independent misclassification cost, which we call misclassification cost or simply cost from now. In binary classification problems misclassification cost can be defined by a cost matrix C, as in Table 2.1. The extension of the cost matrix to multiclass problems is trivial.

Table 2.1: Cost Matrix For Binary Classification Task

	Actual Negative	**Actual Positive**
Predicted Negative	$C(0,0)$ or True Negative (TN)	$C(0,1)$ or False Negative (FN)
Predicted Positive	$C(1,0)$ or False Positive (FP)	$C(1,1)$ or True Positive (TP)

Let the confusion matrix H of a classifier be as in Table 2.2.

Table 2.2: Confusion Matrix of a Binary Classifier

	Actual Negative	**Actual Positive**
Predicted Negative	tn	fn
Predicted Positive	fp	tp

Then the performance of this classifier can be calculated from the expected loss, which is the sum of the Hadamaard (element-wise) product of the cost matrix by the confusion matrix divided by the total sample size,

$$\frac{C \bullet H}{\sum_{i,j} H_{i,j}} = \frac{tn \cdot C(0,0) + fp \cdot C(1,0) + fn \cdot C(0,1) + tp \cdot C(1,1)}{tn + fp + fn + tp} \tag{2.1}$$

Now suppose the algorithm classifies according to the estimated conditional probabilities $P(j|\mathbf{x})$, where j is the class and $\mathbf{x}$ is the attribute vector. Then according to the minimum Bayes risk theory, the decision boundary with respect to the expected risk $R(i,\mathbf{x}) = \sum_j P(j|\mathbf{x})C(i,j)$, should be

$$P(-1|\mathbf{x}) \cdot C(1,0) + P(1|\mathbf{x}) \cdot C(1,1) \leq P(-1|\mathbf{x}) \cdot C(0,0) + P(1|\mathbf{x}) \cdot C(0,1) \tag{2.2}$$

which is equivalent to

$$P(-1|\mathbf{x}) \cdot (C(1,0) - C(0,0)) \leq P(1|\mathbf{x}) \cdot (C(0,1) - C(1,1)) \tag{2.3}$$

Therefore adding a constant to a column of the cost matrix will not modify the optimal decision boundary. Accordingly, we simplify the cost matrix from now on such that all the elements except the main diagonal become zero. Although this cost matrix has two degrees of freedom, as far as the decision boundary is concerned, it only has one degree of freedom, since the fact that "the total sum of class conditional probabilities has to sum up to 1", adds another constraint. Thus, effectively the optimal p^* threshold that makes the decision boundary is given by

$$p^* = \frac{C(1,0)}{C(1,0) + C(0,1)} \tag{2.4}$$

In cost blind classifiers that misclassification costs are all equal, p^* is inherently set to 0.5. Thus a simple way to correct for different cost matrices is changing the threshold to the optimal base rate 2.4. Although threshold correction is a simple way of making classifiers cost sensitive, it is only effective if the classifier can estimate conditional probabilities accurately. However, this is not the case for many well-known classifiers. Classic decision trees and support vector machines can only output a class number and ranking respectively which is sufficient for classification but not for probability estimation. One way to account for this problem is threshold correction by empirical evaluation and cross validation [92]. We will discuss other approaches to handling this problem next.

For the comparison of various learning algorithms, we use the Area Under the ROC Curve (AUC) as our performance measure. AUC can be calculated using Equation 2.5 for a binary classifier. In this equation r_i is the rank of i^{th} positive example based on its probability or score belonging to the positive class sorted in ascending order, and n_+ and n_- are the number of positive and negative training samples respectively.

$$\hat{AUC} = \frac{\sum_i r_i - n_+(n_+ + 1)/2}{n_- n_+} \tag{2.5}$$

There are several intuitive reasons for choosing AUC over other performance measures such as accuracy, precision, recall and F_1 [56]. For example, AUC has increased sensitivity

in Analysis of Variance (ANOVA) tests, is independent of the decision threshold, and thus invariant to a priori class probability distributions [16]. The AUC measure is effectively equivalent to the nonparametric Wilcoxon-Mann-Whitney statistic [54]. We also note that expected loss (2.1) is a good performance measure when we have the actual true cost, which is rarely the case. In other words, the expected loss shows the performance of the classifier for a specific *operating point* (cost matrix and prior class distribution), as compared to AUC which shows the aggregate performance over the entire range of *operating points* on the ROC curve.

2.3 Wrapper Approaches to Cost Sensitive Learning

As previously mentioned, some learning algorithms do not produce accurate probability estimates although they do provide a discriminant function. Wrapper approaches to cost sensitive learning try to make cost blind classifiers cost sensitive, without manipulating the learning algorithm itself. It has been shown that we can achieve the desired base rate by rebalancing the input to the training algorithm, in proportion to the cost matrix [35, 123]. This is basically achieved either by resampling or reweighting the training data. If the base algorithm can handle instance weights, reweighting is always preferred to resampling, since it does not lose any information. Resampling or reweighting is always done according to the optimal base rate, thus the ratio of positive instances to negative instances is in proportion to ($\propto$),

$$\frac{\#positive_{new}}{\#negative_{new}} \propto \frac{P(1)C'(0,1)}{P(-1)C'(1,0)} \tag{2.6}$$

where $P(-1)$ and $P(1)$ are the prior distribution of the positive and negative examples of the training data and C' is the simplified cost matrix.

Since this ratio is only a proportion, it can be achieved by undersampling the majority class, oversampling the minority class or a combination of both. Previous work [30, 123] has shown that undersampling almost always outperforms oversampling in decision trees.

Learning decision trees involves a process known as pruning [37], which stops the growth of the tree somewhere along the path (pre-pruning) or eliminates the leaf nodes of the tree after full growth (post-pruning). In either case, these methods need a validation set which is chosen from the training set; therefore oversampling the minority class creates a biased training sample (with many identical samples in the validation and training set) that defeats the internal validation process when pruning the decision tree. To overcome this problem Zadrozny [123] proposed the so called *Costing* method, which is a rejection sampling method that effectively keeps all samples of the minority class and undersamples the majority class. By bagging several trees learned through resamplings of this form, they develop a very strong cost sensitive wrapper.

If sampling is done deterministically, there is always the danger of injecting bias from the sampling process; therefore random sampling is usually preferred. However, there are methods which introduce synthetic data through a deterministic clustering process. Synthetic Minority Oversampling Technique (SMOTE) [20] is one such method that creates new synthetic samples along the k-nearest neighbor lines adjoining each example of the minority class, and also undersamples the majority class randomly. This method has also been successful to some extent in situations where the training set is highly unbalanced.

2.4 Specifics of the Covered Algorithms

With all the various methods of making classifiers cost sensitive, selecting the right approach can be a tedious task. In order to find which combination works, we empirically study the performance of the most common combinations through controlled experiments. As discussed before, minimum Bayes risk, reweighting, resampling and smoting are the four major wrapper based approaches. In this work we have employed Costing [123] as the sampling method of choice, which according to previous studies has been the most successful.

The first base learning algorithm is the simple Naive Bayes (NB) classifier. NB esti-

mates conditional probabilities for each class from sufficient statistics of the input data as follows,

$$\hat{P}(y|\mathbf{x}) = \frac{\hat{P}(y)\hat{P}(\mathbf{x}|y)}{\hat{P}(\mathbf{x})} = \frac{\hat{P}(y)\prod_{i=1}^{k}\hat{P}(x_i|y)}{\prod_{i=1}^{k}\hat{P}(x_i)} \tag{2.7}$$

Therefore, it can easily handle instance weights by summing up the weight fractions instead of whole numbers when gathering the sufficient statistics. Accordingly, four variations of cost sensitive NB can be imagined: NB with Bayes risk, NB with Smoting, NB with Costing and NB with cost proportionate weighting.

Average One Dependence Estimators (AODE) are a class of generative classifiers proposed by Webb [117]. Naive Bayes has been widely popular in classification because it is simple and efficient. It delivers optimal classification when the estimation of the class conditional probabilities which it relies upon are accurate and the constraints of its simplifying attribute independence assumption truly hold. Although some violations of the attribute independence assumption can be tolerated [28], the accuracy deteriorates as more dependency is introduced. There is an increasing body of work, developing techniques to retain Naive Bayes' simplicity and efficiency while alleviating the problems of the attribute independence assumption [61].

AODE is an efficient technique that utilizes a weaker attribute independence assumption than Naive Bayes, thereby improving prediction accuracy without undue computational overhead, making it suitable for rapid online applications. In AODE, an aggregate of one-dependence classifiers are learned and the prediction is produced by averaging the predictions of all these qualified one-dependence classifiers. For simplicity, a one-dependence classifier is first built for each attribute, in which the attribute is set to be the parent of all other attributes. Then, AODE directly averages the aggregate, consisting of many special Tree Augmented Naive Bayes' [45]. AODE classifies an instance using the following equation,

$$\underset{y}{\operatorname{argmax}} \left(\sum_{i:1\leq i\leq n \wedge F(x_i)\geq m} \hat{P}(y, x_i) \prod_{j=1}^{n} \hat{P}(x_j|y, x_i) \right) \tag{2.8}$$

$F(x_i)$ enumerates the number of training examples which have the value x_i. This function is used to enforce the limit m placed on the support needed in order to accept a conditional probability estimate. The estimation error can be expected to fade towards the mean when the estimates are unbiased. If there are no attributes with support greater than m, AODE would be equivalent to Naive Bayes. AODEsr [125] is AODE with subsumption resolution which entails Lazy Elimination, i.e. it eliminates highly related features during classification without the computational overheads in wrapper feature selection methods.

As for the decision trees, there are many different variations of decision tree induction algorithms. However, the most practical one is Quinlan's C4.5 method [85]. Decision trees can also easily handle instance weighting the same way Naive Bayes' can, since they also induce trees from sufficient statistics of data. Thus, we have four variations of decision trees as well. Previous studies [84] have shown that unpruned decision trees yield more accurate probability estimates. Since here we are rather interested in probability estimates when computing expected cost with Bayes risk, we have chosen unpruned decision trees with Laplace smoothing (or the so called C4.4/Probability Estimation Trees (PET)) as the base algorithm for Costing. This combination works best since unpruned decision trees yield high variance low bias classifiers, and Costing which is a Bagging approach mainly dampens the variance. The rest of the approaches use the standard C4.5 method.

Standard Support Vector Machines (SVM) do not estimate class conditional probabilities. Thus in order to use SVMs as the base algorithm there are two natural extensions. Either use reweighting and employ the rank score for classification directly, or use calibrated probability outputs of SVM scores combined with minimum Bayes risk threshold. For the first case we can use the cost proportionate weights of each training example to vary the importance c_i

of each slack variable ξ_i in SVMs ordinary formulation as in

$$\begin{aligned} \min_{\xi_i, \mathbf{w}, b} \quad & \tfrac{1}{2}\mathbf{w}^\intercal \mathbf{w} + C \sum_{i=1}^{n} c_i \xi_i \\ \text{s.t} \quad & y_i \left(\mathbf{w}^\intercal \mathbf{x}_i + b\right) \geq 1 - \xi_i \,, \; i = 1, \cdots, n \\ & \xi_i \geq 0. \end{aligned} \tag{2.9}$$

where $h(\mathbf{x}) = \text{sign}\left(\mathbf{w}^\intercal \mathbf{x} + b\right)$ is the decision function and $\sum_{i=1}^{n} c_i \xi_i$ is an upper bound on the total loss.

In the second case we have used Platt's scaling [83] to obtain accurate probability estimates from SVM ranking scores. The SVMs use an RBF kernel optimized by a 3 level deep grid search over the complexity range of $C = 2^{-22\sim 7}$ and gamma range of $\gamma = 2^{-10\sim 3}$.

To have a fair comparison of the range of algorithms, we also included another highly successful classification method which is the boosting approach. We used Adaboost [41] to create an ensemble of unpruned decision trees (C4.4). Three fold internal cross validations were performed to choose the optimal number of iterations (10~1000). We also used Logitboost [42] with decision stumps trained up to 2000 iterations and internal cross validation to find the optimal number of iterations. All the cross validations were stratified and prior instance weights were set in proportion to their cost. In the next section we discuss the experimental setup and data sets.

2.5 Empirical Results

To obtain low variance estimates of the performance measure, we have performed a 10 fold stratified cross validation over 24 data sets from the UCI Machine Learning Repository [7]. Multi-class and continuous class data sets were discretized and converted to binary classification problems. For a fair comparison, all data sets were discretized by the same supervised MDL method [39] and binarized according to the **Condition** column of Table 2.3 in a preprocessing step. The details of the data sets are shown in Table 2.3. As can be seen from

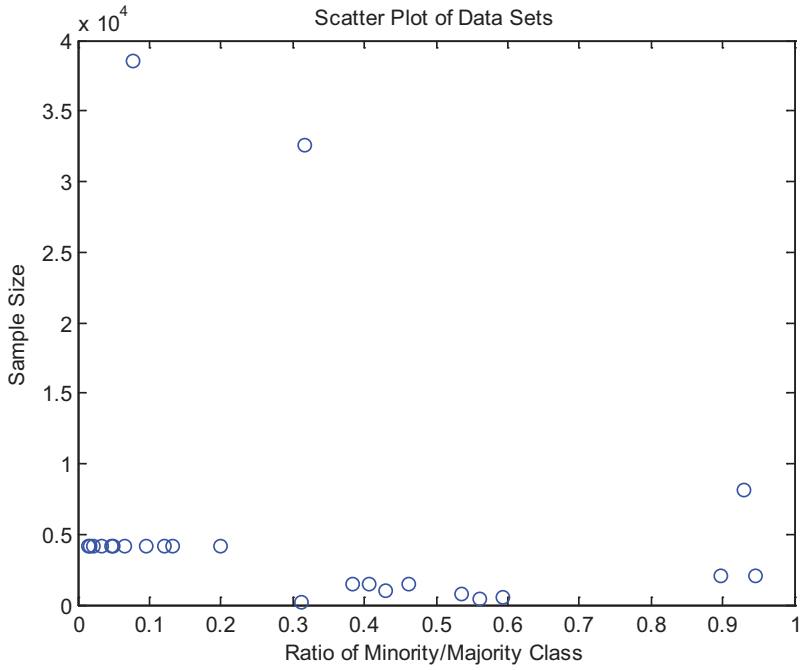


Figure 2.1: Scatter plot of data set minority weight ratio versus sample size.

the scatter plot of Figure 2.1, the data sets were chosen such that the weight ratio of the minority class over the majority class would cover the range of possible weights. A weight ratio of 1 means that the two classes have equal sample sizes and costs. For each data set the cost of misclassifying the minority class is set equal to the sample size for the majority class and vice versa. Thus, misclassifying the minority class would be relatively more costly than the majority class. We believe that assigning only one cost matrix to each data set should be enough when performance is computed using AUC, since AUC does not depend on prior class distributions and costs. Nevertheless we also compared the algorithms based on average misclassification cost, to see how well each method performs when optimizing for a single operating point.

Table 2.4 shows the ranking of each algorithm based on win/loss counts as measured by a two tailed t-test over AUC at 0.05 significance rate. Similarly, Table 2.5 shows another ranking based on win/loss counts of average misclassification cost as computed in Equation (2.1), also by a two tailed t-test at 0.05 significance rate.

Table 2.3: Detailed description of datasets.

Row	Data Set	Condition	Sample Size	Minority Ratio	Attributes
1	abalone0	Age ≤ 4	4177	0.018	8
2	abalone1	Age ≤ 5	4177	0.0474	8
3	abalone2	Age ≤ 6	4177	0.1201	8
4	abalone3	Age>12	4177	0.1989	8
5	abalone4	Age>13	4177	0.1329	8
6	abalone5	Age>14	4177	0.0955	8
7	abalone6	Age>15	4177	0.0666	8
8	abalone7	Age>16	4177	0.0487	8
9	abalone8	Age>17	4177	0.0337	8
10	abalone9	Age>18	4177	0.023	8
11	abalone10	Age>19	4180	0.0158	8
12	adult		32561	0.3172	14
13	mushroom		8124	0.9306	21
14	covertype0	Ponderosa Pine vs Cotton-wood/Willow	38481	0.0769	54
15	german		1000	0.4286	20
16	ionosphere		351	0.56	34
17	pima Indians diabetes		768	0.536	8
18	satellite0	2 vs 7	1515	0.4624	36
19	satellite1	1 vs 3	2010	0.898	36
20	satellite2	4 vs 7	1414	0.407	36
21	satellite3	5 vs 7	1455	0.3844	36
22	satellite4	3 vs 7	2020	0.9461	36
23	wisc diag breast cancer		569	0.5938	30
24	wisc prog breast cancer		198	0.3113	33

Table 2.4: Win/Loss ranking ordered by first column based on AUC as performance measure.

Win-Loss	Win	Loss	Algorithm
223	227	4	AODEsr with cost proportionate weighting
202	214	12	AODE with cost proportionate weighting
152	190	38	Naive Bayes with cost proportionate weighting
150	188	38	Costing 100 Naive Bayes
138	163	25	Costing 100 unpruned J48 trees
53	129	76	Adaboost: up to 1000 unpruned J48 trees with cost proportionate weighting
31	95	64	Logitboost: up to 2000 Stumps with Cross Validation
-62	57	119	SVM: 3 level deep grid search with RBF kernel $\gamma = 2^{-10\sim3}$, $C = 2^{-22\sim7}$, Platt's Scaling and Bayes Risk
-99	71	170	5-NN SMOTED Naive Bayes with Bayes Risk
-126	32	158	Multi Layer Perceptrons with cost proportionate weighting
-128	31	159	J48 tree with cost proportionate weighting
-131	54	185	Naive Bayes with Bayes Risk
-140	18	158	SVM: 3 level deep grid search with RBF kernel $\gamma = 2^{-10\sim3}$, $C = 2^{-22\sim7}$ and cost proportionate weighting
-155	24	179	5-NN SMOTED J48 tree with Bayes Risk
-195	8	203	J48 tree with Bayes Risk

Table 2.5: Win/Loss ranking ordered by first column based on average loss as performance measure.

Win-Loss	Win	Loss	Algorithm
268	273	5	SVM: 3 level deep grid search with RBF kernel $\gamma = 2^{-10\sim3}$, $C = 2^{-22\sim7}$ and cost proportionate weighting
259	263	4	SVM: 3 level deep grid search with RBF kernel $\gamma = 2^{-10\sim3}$, $C = 2^{-22\sim7}$, Platt's Scaling and Bayes Risk
38	78	40	Logitboost: up to 2000 Stumps with Cross Validation
30	78	48	AODEsr with cost proportionate weighting
21	72	51	Costing 100 unpruned J48 trees
19	71	52	AODE with cost proportionate weighting
16	68	52	Multi Layer Perceptrons with cost proportionate weighting
8	58	50	Adaboost: up to 1000 unpruned J48 trees with cost proportionate weighting
1	60	59	J48 tree with cost proportionate waiting
-6	60	66	J48 tree with Bayes Risk
-39	52	91	5-NN SMOTED J48 tree with Bayes Risk
-137	14	151	Costing 100 Naive Bayes
-145	13	158	Naive Bayes with Bayes Risk
-148	11	159	Naive Bayes with cost proportionate weighting
-185	6	191	5-NN SMOTED Naive Bayes with Bayes Risk

By comparing Tables 2.4 and 2.5 we can see that there is a distinctively different pattern of ranking between the two measures. The family of Bayesian approaches, such as Naive Bayes and AODE, significantly outperform SVM and boosting in AUC, however, the situation is reversed when ranking is based on average loss. These results might seem a little surprising and hard to explain at first sight, but the a closer inspection suggests the following explanation.

There is an apparent tradeoff between a general decision function which is optimized over all possible costs and prior distributions (conditional probability), and a decision function which is optimized for a single operating point pivoted on a cost matrix and prior class distribution (classification/discriminant function).

The results of Table 2.5 shows that the classical SVM and boosting approaches, which have been previously proven highly successful in classification, can be tailored to different operating points. However, since we might not always know the exact operating point in our application, which is usually the case, it is beneficial to develop an algorithm that could find the optimal decision over a range of operating points. That is, in some applications, neither of the two extreme cases in Table 2.4 and 2.5 is a good choice. For example the P-norm Ranking [88] approach is one such technique that concentrates on accurate ranking of the top portion of the list.

2.6 Comparison of Multiple Classifiers

To delve a little deeper and gain some insight into the properties of each algorithm, we employ a statistical hypothesis testing approach that is more precise in terms of *power* and *replicability* for classifier performance measures, rather than the win/loss counts based on t-tests usually found in the literature. The stringent battery of tests which follows would reveal the true performance and significance of each algorithm.

When comparing multiple classifiers, two concerns need to be addressed. Firstly, how

accurate can the performance measures be estimated? Meaning that, how robust are our estimates of the performance measures in the face of variability of the original data generation process? The second question is, how accurate and powerful are the comparisons being made? In other words, can we compute the significance of differences in the computed performance measures (AUC, expected loss, and etc.)?

Clearly, these two concerns are not independent. The lower the quality of our estimates of the performance, the less power we would have when testing significance.

These concerns have been addressed by Demšar [25], where he suggested using the Friedman test [43, 44] and its post-hoc Nemenyi tests [80], instead of win/loss counts based on paired t-tests. Friedman test is a non-parametric form of ANOVA with repeated measures as opposed to ranksum test which is a non-parametric form of ANOVA where there are no repeated measurements or correlation among test subjects.

The problem with using win/loss counts based on t-tests is two folds. Firstly, one cannot compute the significance of differences in "win-loss" that is usually measured. Secondly, it assumes that any difference in the performance less than the critical value at α percentile as measured by a t-test is non-significant. That is, whether we compare two algorithms 10 times or a 1000 times, the result would be insignificant when the difference is less than a certain amount, even if one algorithm always outperforms the other, which is totally absurd.

The reason Friedman test was chosen over ANOVA with repeated measures is that ANOVA assumes *normarlity* and *sphericity* of the distribution of the test subjects. Although the normality assumption is usually overlooked by statisticians, the sphericity assumption can seriously hurt the power and replicability of the test and especially the post-hoc t-tests. The performance measures calculated for classifiers, such as AUC and expected loss, adhere neither to normality nor sphericity assumptions. This is especially true when facing asymmetric loss functions and unbalanced prior distributions, which happens to be the case in cost sensitive settings.

Let r_i^j be the rank of the j-th of k algorithms on the i-th of N data sets. The Friedman

test compares the average ranks of algorithms, $R_j = \frac{1}{N}\sum_i r_i^j$. Under the null-hypothesis, which states that all the algorithms are equivalent and so their ranks R_j should be equal, the Friedman statistic

$$\chi_F^2 = \frac{12N}{(k(k+1))}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right] \tag{2.10}$$

is distributed according to χ_F^2 with $k-1$ degrees of freedom, when N and k are big enough (as a rule of a thumb, $N > 10$ and $k > 5$). For a smaller number of algorithms and data sets, exact critical values have been computed [93, 124]. Iman and Davenport [57] showed that Friedman's χ_F^2 is undesirably conservative and derived a better statistic

$$F_F = \frac{(N-1)\chi_F^2}{N(K-1)-\chi_F^2} \tag{2.11}$$

which is distributed according to the F-distribution with $k-1$ and $(k-1)(N-1)$ degrees of freedom. This is the statistic used in the current study. If the null-hypothesis is rejected, we can proceed with a post-hoc Nemenyi test similar to the Tukey test for ANOVA, which is used to compare all classifiers to each other. The performance of two classifiers is significantly different if the corresponding average ranks differ by at least the critical difference

$$CD = q_\alpha\sqrt{\frac{k(k+1)}{6N}} \tag{2.12}$$

where critical values q_α are based on the Studentized range statistic divided by $\sqrt{2}$.

Using the described statistical testing procedure, we conducted the same comparisons of different cost sensitive approaches on 24 datasets. The results are shown in Figures 2.2 and 2.3 for 1-AUC, and in Figures 2.4 and 2.5 for average loss, as performance measures respectively.

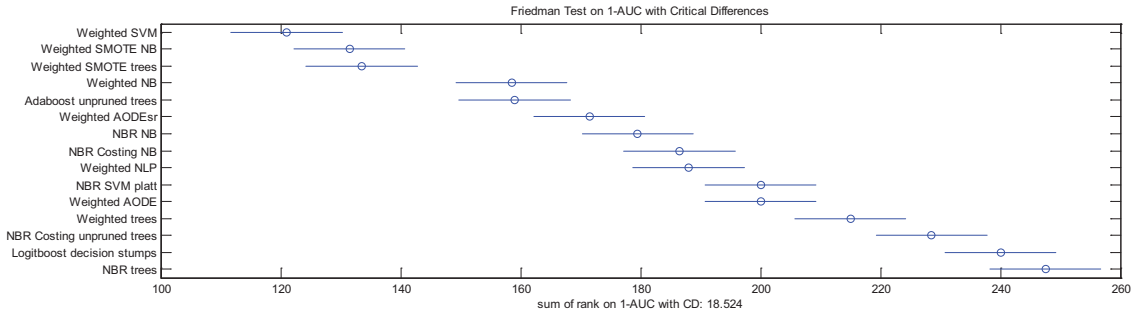


Figure 2.2: Different methods sorted from top to bottom according to sum of rank on 1-AUC, with critical difference computed using Equation (2.12).

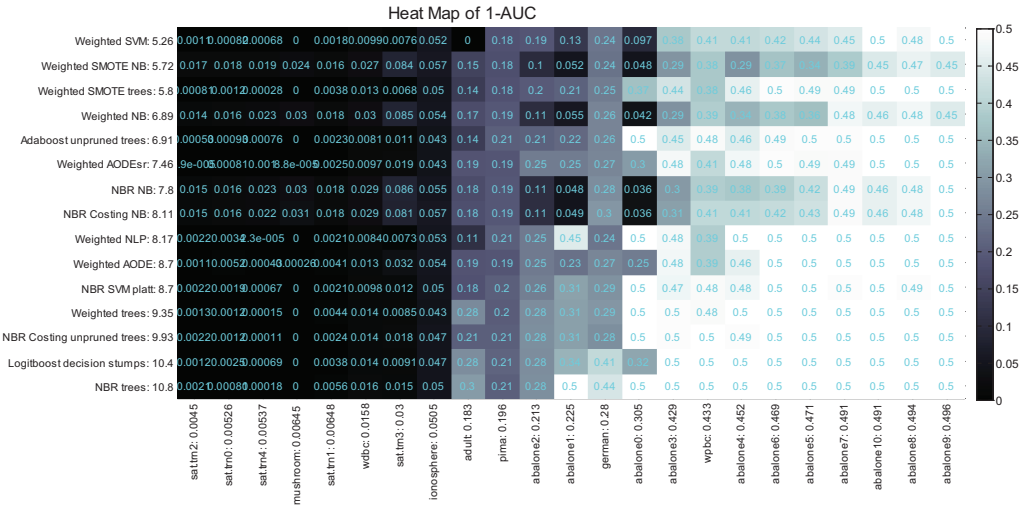


Figure 2.3: Different methods sorted from top to bottom according to average rank on 1-AUC, and datasets from left to right according average difficulty (average 1-AUC over all methods).

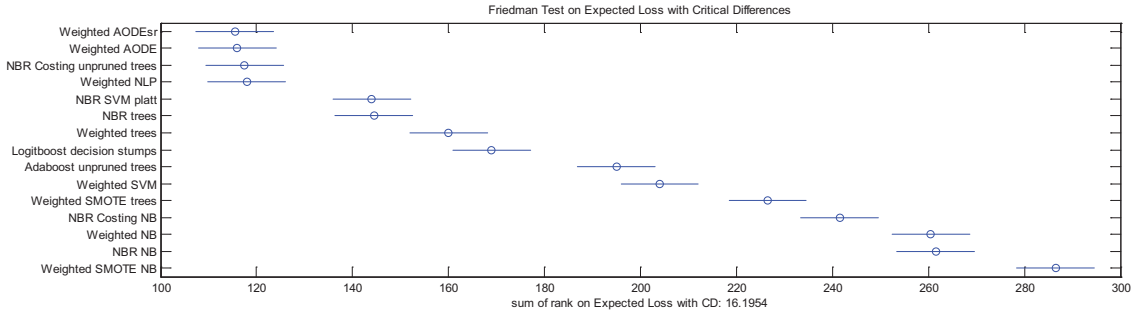


Figure 2.4: Different methods sorted from top to bottom according to sum of rank on average loss, with critical difference computed using Equation (2.12).

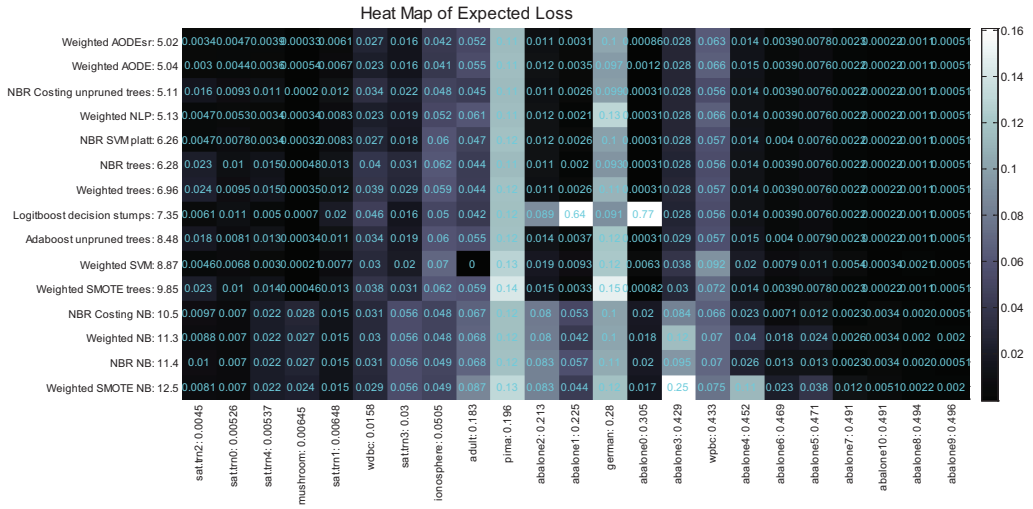


Figure 2.5: Different methods sorted from top to bottom according to average rank on expected loss, and datasets from left to right according average difficulty (average 1-AUC over all methods).

2.7 Conclusion

In many practical problems in the machine learning domain, such as advertisement and digital marketing (click through rate prediction), credit assignment, network intrusion and medical diagnosis, the cost or utility of different actions vary significantly. In such problems, it is vital to take advantage of the information about different costs when learning the decision function. There is an inherent tradeoff between learning the conditional probability over the complete range and learning a binary decision function that outputs two values.

This tradeoff is made apparent from the experiments conducted in Sections 2.5 and 2.6. The results show that maximum margin methods such as SVM and Boosting excel at learning a binary decision function, while Bayesian methods are better at predicting probabilities. Depending on the problem domain and whether information about various costs exist, one should take appropriate choices for the base learning method and its cost sensitive extension to achieve optimal results.

CHAPTER 3 AN EXTENDABLE META-LEARNI ALGORITHM FOR ONTOLOGY MAPPING*

3.1 Introduction

As means of communication grows with the growth of the web and the technologies provide abundance of tools for creating a wealth of information, the problem of information dissemination and reconciliation worsens far more rapidly than the problems faced in creating the knowledge in the first place. This issue is prominently recognized at the advent of Semantic Web, and therefrom the problem of information integration, which has been a subject of intense research for more than a couple of decades, still lies at the heart of the present day research.

On the Semantic Web, the information at the highest level is provided through ontologies, and inevitably the same problems of heterogeneous data sources that arose with the web surfaced again. In this chapter we try to tackle a specific problem in ontology mapping, a part of information integration, which is not addressed in the current literature.

A large body of work in ontology integration, including ontology mapping and ontology merging, has focused on developing algorithms and heuristics [94] to mitigate the burden of aligning large ontologies manually. However, in many cases, the exact place where machine learning can play a key role is intermixed with heuristics of defining and fine tuning various thresholds [107], or several layers of learning [26], that along the way, has made deploying this technique rather difficult. In this chapter we strive to develop a framework to push the heuristic parts of the ontology mapping problem exactly where it belongs, i.e. in defining the similarity measures and not in the learning paradigm, and propose a simple yet extendible

*This chapter has been published in [91].

framework that is comparable to the state of the art techniques, i.e. can easily be improved as better similarity measures are developed and still be comparable.

There are several advantages in using such an approach. One is that the current techniques of machine learning can be used to automatically learn the parameters, and hence no threshold setting or fine tuning interventions would be necessary on the part of the user. The other is that the total outcome would solely depend on how well the similarity measures are defined and the data sets are prepared. Of course, one could argue that the latter benefit is a disadvantage, since creating informative similarity measures can be a challenge for many domains. However, handcrafting ad hoc learning methods that work well for a set of specific similarity measures in each specific ontology alignment problem would not make the original problem any easier. Therefore, if we put the effort on defining informative similarity measures, this framework can utilize the information therein in a coherent manner, by only plugging in the measure. The implication would be that, since no further improvement could be done automatically based on the available training data and defined similarity measures, we can turn our attention to creating better similarity measures. Furthermore, when the class distributions are highly unbalanced, as in the ontology mapping problem, it turns out that our approach has better precision recall curves.

In this work, we address the problem of ontology mapping, which specifically is, finding corresponding entities in pairs of ontologies that appear to be an approximate match. Broadly speaking, an ontology is a set of axioms used to formally define *concepts* and *relations* that exist in a certain domain and assert information about *individuals* in that domain. From this definition three emphasized keywords can be extracted that can be informative for comparison of two ontologies. In the rest of the chapter we lay out the foundation of the machine learning paradigm for ontology mapping in the hope of addressing the automatic threshold selection. Therefore, this work is not trying to develop any novel similarity measures as it might be highly dependent to each domain.

3.2 The Web Ontology Language (OWL)

In this section we give a short review of how ontologies are represented on the web by introducing the current de facto standard (OWL), and present a pair of ontologies as an example to depict the information that can be extracted from this domain in order to build the similarity measures.

The sole purpose of ontologies is to make unstructured information embedded in documents machine understandable. The latest advancement in ontology representation is OWL which can be used to explicitly define entities in terms of vocabularies and the relationships between those entities. This representation of entities and their interrelationships is called an ontology. OWL is more a expressive language than XML, RDF and RDF-S and thus better capable of representing machine readable content. For a detailed overview of the language readers are referred to [3].

From a modeling and semantic point of view, OWL shares a strong correspondence with Description Logics borrowing many logical constructs from it. OWL comes in three increasingly expressive sub-languages or "species", OWL-Lite, OWL-DL and OWL-Full.

OWL-Lite: The reason behind OWL Lite is to support users that require a hierarchy of entities with simple constraints. These expressivity limitations ensure that it provides a minimal useful subset of language features, which are relatively straightforward for tool developers to support.

OWL-DL: OWL-DL supports those users who want the maximum expressiveness of the language without losing decidability. It includes all the OWL language specifications, but they can only be used under certain restrictions.

OWL-Full: OWL-Full has the same vocabulary as OWL DL but it allows the free, unrestricted use of RDF constructs (e.g., classes can be instances). OWL-Full is thus identical in syntax to OWL-DL with an extended semantics of RDF, but is undecid-

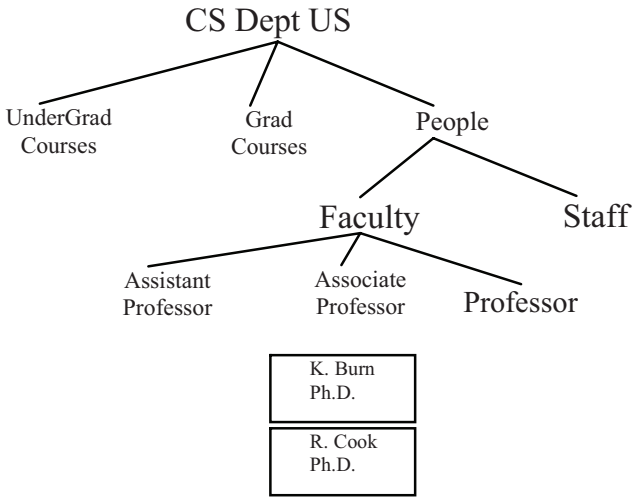
able.

We will be using OWL-DL constructs to be able to include similarity measures that exploit inference capabilities of OWL representation from each ontology in finite time (decidability). As previously stated, there are three broad categories of entities that can be extracted from an ontology, namely the concepts, the relations (properties) and the instances, also known as individuals. To elaborate on the definition and meaning of these entities we introduce an example adapted from [26]. In Figure 3.1 two computer science department ontologies are shown from two universities with different taxonomies. As can be seen from this figure, entities like *People, Courses* and *Staff* refer to classes which are concepts in the corresponding ontologies. The lines between each class represent the relations among classes, such as *Faculty* and *Staff* which are related to *People* in Figure 3.1(a). Lastly, entities like *"R. Cook"* and *"K. Burn"* are instances of class *Associate Professor* in the ontology.

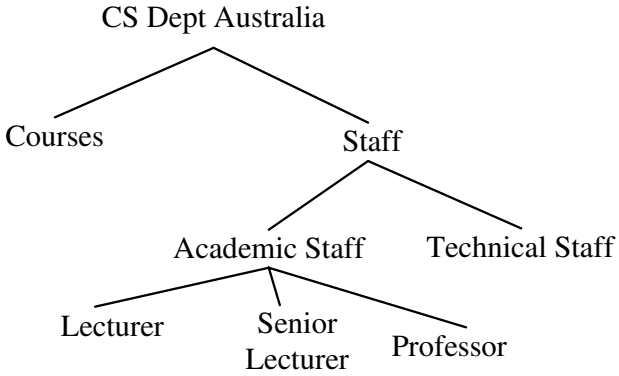
The two CS departments represent two ontologies created by different people, and therefore, although they describe a similar domain, some entities do not match precisely. For example *Assistant Professor* in *"CS Dept US"* is equivalent to *Lecturer* in *"CS Dept Australia"*, and also some entities do not appear in the same hierarchical relation, such as *"UnderGrad Courses"* and *"Grad Courses"* which directly appear under *"CS Dept US"*, but if they were to exist in *"CS Dept Australia"* they would have come under *"Courses"* in Figure 3.1(b).

3.3 Machine Learning Approach to Ontology Mapping

To motivate the use of machine learning approaches to ontology mapping, let us reformulate the mapping problem in terms of a classification problem [31] in which we want to have a machine which classifies the input patterns into different classes. In this section we will formally describe supervised learning, an area that specifically addresses this problem, and present the current best method suitable for handling ontology mapping problem requirements.



(a)



(b)

Figure 3.1: A Computer Science Department Ontology

3.3.1 Formulating Ontology Mapping as a Supervised Learning Problem

Supervised learning is a machine learning technique for creating a discriminant function from a set of training data. The training data consists of pairs of input vectors, and desired outputs. The output of the function can be a continuous value (called *regression*), or a discrete class label of the input vector (called *classification*). The task of the supervised learner is to predict the value of the function for any valid input object after having seen a number of training examples (i.e. pairs of input and desired output). To achieve this, the learner has to generalize from the presented data to unseen situations in a "reasonable" way (i.e. with low variance and bias).

There are two broad categories of classifiers. The generative classifiers learn a model of the joint probability distribution, $P(\mathbf{x}, y)$, of the inputs $\mathbf{x}$ and the label y and make their predictions by using Bayes rule to calculate $P(y|\mathbf{x})$, and then picking the most likely label y. Discriminative classifiers, on the other hand, model the posterior $P(y|\mathbf{x})$ directly, or learn a direct map from the inputs $\mathbf{x}$ to class labels. Contrary to the widely held belief that the discriminative classifiers are almost always to be preferred, there can be two distinct regimes of performance as the training size is increased. Previous studies [81] show that, while discriminative learning has a lower asymptotic error, a generative classifier may also approach its (higher) asymptotic error much faster. Because of this property and noting that creating training examples for the ontology mapping problem is a tedious task (usually there are not many examples available), we opted for the generative classifier models. There are also various other benefits which will be discussed in Section 3.3.3.

To formulate the ontology mapping problem as a supervised learning task we have to define the input vector $\mathbf{x}$ and output class variable y. The input vector $\mathbf{x}$ can be defined as a set of similarity measures computed from the information extracted from the pair of ontologies. The class label y can be *true* if the two compared entities are actually the same, and *false* otherwise. The similarity scores and the approximate string matching techniques

employed to compute the similarity measures are described in Section 3.4. In the next subsection the proposed classifier is described.

3.3.2 Average One Dependence Estimators (AODE)

Average one dependence estimators are a class of generative classifiers [117]. To understand the internal workings of this algorithm, first we have to describe the Naive Bayes (NB) Classifier and the extensions of this algorithm. Naive Bayes has been widely used in classification due to its simplicity, and effectiveness. It delivers optimal classification when the estimation of the class conditional probabilities on which it relies are accurate and the constraints of its simplifying attribute independence assumption truly hold. Although some violations of the attribute independence assumption can be tolerated [29], the accuracy deteriorates as more dependency is introduced, and there is an increasing body of work developing techniques to retain Naive Bayes' desirable simplicity and efficiency while alleviating the problems of the attribute independence assumption [61].

Previous Studies [115] show that Lazy Bayesian Rules (LBR) [126] has demonstrated accuracy comparable to boosting decision trees [127], and Super Parent Tree Augmented Naive Bayes (SP-TAN) [63] has comparable accuracy to Lazy Bayesian Rules, both of which are statistically superior to Naive Bayes. However, these two techniques have high computational overheads, SP-TAN having high computational complexity at training time and Lazy Bayesian Rules having high computational complexity at classification time. Since we are interested in a scalable and extendible method, they do not serve our purpose. However, Average One Dependence Estimator is an efficient technique that utilizes a weaker attribute independence assumption than Naive Bayes, thereby improving prediction accuracy without undue computational overheads, making it suitable for rapid online applications. We first describe Naive Bayes and subsequently describe Average One Dependence Estimator.

Let $\mathbf{x} = \{x_1 \ldots, x_n\}$ be an example from a training set, where x_i is the value of the i^{th} similarity measure. We want to predict the class $y \in c_1, \ldots, c_k$ from this set. If we had the

true distribution $P(y|\mathbf{x})$ we could optimally predict y by selecting $\underset{y}{\operatorname{argmax}}\, P(y, \mathbf{x})$. However since $P(y|\mathbf{x})$ is not available, we try to estimate it based on the training set. This estimate is usually denoted by $\hat{P}(y|\mathbf{x})$.

One way to estimate $\hat{P}(y|\mathbf{x})$ is from $P(y, \mathbf{x})$, since by definition,

$$P(y|\mathbf{x}) = P(y, \mathbf{x})/P(\mathbf{x}) \tag{3.1}$$

$$\propto P(y, \mathbf{x}) \tag{3.2}$$

Therefore, $\underset{y}{\operatorname{argmax}}\, P(y|\mathbf{x}) = \underset{y}{\operatorname{argmax}}\, P(y, \mathbf{x})$. Now if we ever need $P(y|\mathbf{x})$ we can always use Equation 3.3 to calculate this value,

$$\frac{P(y, \mathbf{x})}{P(\mathbf{x})} \approx \frac{\hat{P}(y, \mathbf{x})}{\sum_{i=1}^{k} \hat{P}(c_i, \mathbf{x})} \tag{3.3}$$

This is because $P(\mathbf{x}) = \sum_{i=1}^{k} \hat{P}(c_i, \mathbf{x})$. However, since we are only interested in predicting the class y, we only need $\hat{P}(y, \mathbf{x})$.

When the training sample is small, as in our case, or the number of similarity measures is large, the frequency of which any event $\mathbf{x}$ happens would be very small and therefore $\hat{P}(y, \mathbf{x})$ would not be a reliable estimate of $P(y, \mathbf{x})$. However, there is no way to eliminate this problem considering this model and the data we have, and in practice, only a minor adjustment such as Laplace correction can be made. That is, a constant number is added to every event not occurred in order to have a non zero probability. Also, another way of overcoming this problem is by changing the model and assuming independency among all similarity measures. This make our base estimates to be from $\hat{P}(y, x)$ instead of $\hat{P}(y, \mathbf{x})$, which is more likely to occur in our training set.

Using the product rule of probabilities we have the following.

$$\hat{P}(\mathbf{x}|y) = \prod_{i=1}^{n} \hat{P}(x_i|y) \tag{3.4}$$

where $\hat{P}(x_i|y) = \hat{P}(y, x_i)/\hat{P}(y)$, and $\hat{P}(y)$ is the prior probability of any class $y = c_i$. This technique is widely known as the Naive Bayes assumption. Therefore Naive Bayes classifies by selecting

$$arg\ max_y \left(\hat{P}(y) \prod_{i=1}^{n} \hat{P}(x_i|y) \right) \tag{3.5}$$

where $\hat{P}(y)$ and $\hat{P}(x_i|y)$ are estimates of the respective probabilities derived from the frequency of their respective arguments in the training sample, with possible corrections such as the Laplace estimate.

One of the recent works on improving Naive Bayes is *Average One-Dependence Estimators*, or simply AODE [117]. In Average One Dependence Estimators, an aggregate of one-dependence classifiers are learned and the prediction is produced by averaging the predictions of all these qualified one-dependence classifiers. For simplicity, a one-dependence classifier is firstly built for each attribute, in which the attribute is set to be the parent of all other attributes. Then, Average One Dependence Estimator directly averages the aggregate consisting of many special tree augmented Naive Bayes. Average One Dependence Estimator classifies an instance using the following equation.

$$arg\ max_y \left(\sum_{i:1 \leq i \leq n \wedge F(x_i) \geq m} \hat{P}(y, x_i) \prod_{j=1}^{n} \hat{P}(x_j|y, x_i) \right) \tag{3.6}$$

where $F(x_i)$ enumerates the number of training examples which have the value x_i, which is used to enforce the limit m placed on the support needed to accept a conditional probability estimate. If $\neg\exists i : 1 \leq i \leq n \wedge F(x_i) \geq m$, AODE would be equivalent to Naive Bayes. In the current work we used $m = 30$ suggested by [117].

3.3.3 Properties of Average One Dependence Estimators

Having described this method we will discuss the properties of Average One Dependence Estimator algorithm compared to other previous best approaches and the implications of

employing this algorithm for ontology mapping. Compared to Lazy Bayesian Rules and SP-TAN, the two previous best approaches, Average One Dependence Estimator is very efficient. Looking into Lazy Bayesian Rules and SP-TAN, one can recognize that computational cost of these two algorithms can mostly be attributed to *model selection* and *probability estimation*.

Lazy Bayesian Rules adopts learning at classification time. For each $\mathbf{x} = \{x_1, \ldots, x_n\}$ to be classified, a set of the similarity measures W, is selected, and the remaining features are assumed to be independent given W and y. Thus, every measure depends both on the class and the measures chosen for inclusion in W. W is heuristically selected in order to minimize error on the training sample. Thus, the computational effort is high when there is huge number of examples.

In contrast to Lazy Bayesian Rules, Tree Augmented Naive Bayes and SP-TAN allow every similarity measure x_i to depend upon the class and at most one other measure, $p(x_i)$, called the parent of x_i. The parent function $p(.)$ is learned at training time. Conditional mutual information is employed in Tree Augmented Naive Bayes [115] to select the parent function. A naive heuristic method that minimizes error on training samples is used in SP-TAN [63]. At training time both Tree Augmented Naive Bayes and SP-TAN generate a three-dimensional table of probability estimates for each measure-value, conditioned by each other measure-value and each class.

Unlike Lazy Bayesian Rules, where it has to choose the similarity measure set W, and SP-TAN, where it has to select $p(.)$, Average One Dependence Estimator has no model selection, except for choosing all one dependence estimators having a sufficient support set. Also, unlike Lazy Bayesian Rules, where the probability estimates are generated on the fly in the classification phase, Average One Dependence Estimator and SP-TAN compute the estimates at training time via the three dimensional probability tables, thus making it very efficient both at training and classification time.

The other major implication of using Average One Dependence Estimator is that it can

readily be incorporated in an incremental learning setup. This means that if the current training examples are not sufficient for a reasonable estimate of the probability tables, they can be later incorporated when they become available, and the probability tables re-estimated accordingly to reflect the adjustments needed to utilize the information contained in the recently included examples. Therefore, the effect would be that we can build a system for mass collaboration of users to contribute in identifying matching concepts and add the examples as they become available. This will have a major impact on the current efforts of using machine learning approaches for online ontology mapping, since the shortage of training examples is the first and foremost difficulty in applying machine learning techniques in such applications.

3.4 Creating Training Examples

By far the most crucial factor in developing a successful classifier is having ample independent identically distributed training data with informative similarity measures. The first step towards creating informative similarity measures is to understand the nature of the data being categorized.

Since developing informative similarity measures depend highly on both the representation and domain of the ontology pair, the best thing would be to use a widely employed language for representation and a set of similarity measures that work well for a group of domains. For the representation part, we have already discussed the OWL ontology language, and in this work we only consider expressive languages up to OWL-DL variant (species). Thus, we would be able to compute logical similarity measures through reasoning along with other semantic and syntactic similarity measures, without compromising decidability. However, one should note that the semantic and syntactic similarity measures are usually highly dependent on the ontology domain. Hence, we want to use a meta-learner model robust enough to be able to handle inconsistent similarity measures.

As previously mentioned, in the ontology mapping problem there are three key entities that inform us about the nature of an ontology, namely classes, relations and instances. Thus, obviously there are three categories of similarity measures which can be defined to quantify the similarity for each pair of classes: comparing the classes, comparing the relations of a pair of classes, and comparing the instances of a pair of classes. To further elaborate on this issue, we will describe one similarity measure for each category.

The most straight forward way of comparing two classes is to compare their names. There is always, however, the difficulty of two classes referring to the same entity with different names. There are two ways to approach this problem; one is to consider the similarity in the semantic space, and the other in the syntactic space, the former being comparing words in terms of being synonymous in a controlled vocabulary domain and the latter comparing words in terms of approximate string matching. In this work, the second method is adopted and some of the popular approximate string matching algorithms are discussed later in this section.

In its most general form, an ontology can be represented as a directed acyclic graph. That is, the classes in an ontology are brought together by a set of relations. Thus, one way to compare the similarity of two classes is to compare their relations in their corresponding ontologies, such as the parents, children and siblings of a class. The similarity measure used in this work is the path of classes required to traverse from the root to reach the considered class.

For the last category of similarity measures, instances can be compared in various ways. Each class has a set of instances and therefore to quantify the similarity of two classes, one could compare the set of instances for each class. Comparing two sets of instances can be approximate or exact itself, however, since our contribution as described in the introduction is not developing new similarity measures, we have only applied exact similarity functions for set comparisons using the Jaccard coefficient, described later in this section. In the rest of this section we will note some of the approximate string matching algorithms, widely

employed in the literature and describe the ones used in this work.

The Levenshtein distance [66] is a string similarity metric to measure the edit distance. The Levenshtein distance between two strings is given by the minimum number of operations needed to transform one string into the other, where an operation is an insertion, deletion, or substitution of a single character. It can be considered a generalization of the Hamming distance, which is used for strings of the same length and only considers substitution edits.

The Monge-Elkan [77] distance is a variant of the Gotoh distance which is an extension of the Smith-Waterman algorithm with affine gap costs. It is calculated based on the sum of best matching atomic substring of the compared strings with the following formula

$$match(A, B) = \frac{1}{|A|} \sum_{i=1}^{|A|} max_{j=1}^{|B|} match(A_i, B_j) \tag{3.7}$$

The Jaro distance [60] takes into account typical spelling deviations. Briefly, for two strings s and t, let s' be the characters in s that are "common with" t, and let t' be the characters in t that are "common with" s; roughly speaking, a character a in s is "in common" with t if the same character a appears in about the same place in t. Let $T_{s',t'}$ measure the number of transpositions of characters in s' relative to t'. The Jaro similarity metric for s and t is

$$Jaro(s, t) = \frac{1}{3} \left(\frac{|s'|}{|s|} + \frac{|t'|}{|t|} + \frac{|s'| - T_{s',t'}}{2|s'|} \right) \tag{3.8}$$

This approach has since been expanded to a slightly modified approach called Jaro-Winkler. This extension modifies the weights of poorly matching pairs (s, t) that share a common prefix. The output score is simply adjusted as follows

$$JW(s, t) = J(s, t) + (prefixlen \times prefixscale \times (1 - J(s, t)) \tag{3.9}$$

where *prefixlen* is the length of the common prefix at the start of string, and *prefixscale* is a

constant scaling factor for how much the score is adjusted upwards for having common prefix's. This adjustment gives more favorable ratings to strings that match from the beginning for a set prefix length.

Previous studies [22] show that although Monge-Elkan distance gives the best results, the Jaro-Winkler's distance metric achieves comparable results but is an order of magnitude faster. Thus we adopted the Jaro-Winkler score as our string similarity metric.

Set comparisons of two set of instances of each class is calculated with the Jaccard coefficient. The Jaccard coefficient of two sets A and B is computed using the following formula,

$$Jaccard(A, B) = \left(\frac{|A \cap B|}{|A \cup B|}\right) \tag{3.10}$$

3.5 Generating the Final Mapping

From the previous step we have generated the pairwise similarity measures of two concepts for each pair of concepts in two ontologies. Thus, we have a matrix of similarity measures and from this matrix we have to generate a mapping between the ontologies. Before explaining our method we have to clarify a few points.

From another view point [26] the similarity measures defined in the previous section can be classified into intrinsic and extrinsic measures, instead of concepts, relations and instances. Intrinsic similarity measures are the measures computed solely from the information obtained from the class itself, such as the name or content (instances), and extrinsic similarity measures are measures computed from any other information we have about the classes, such as its relation to other classes in the ontology structure.

To create a similarity score of two classes we have used both intrinsic and extrinsic similarity measures as discussed in the previous section. Therefore, in our approach, the single best similarity score is sufficient for generating the final mapping. Since we don't have any other information to include in the classification process, any random tie breaking scheme

seems reasonable to resolve the conflicting scores.

Another approach [26] to this problem, however, is to discard any extrinsic similarity measure in computing the similarity score and include it later to resolve the conflicting scores, and generate an improved mapping. The relaxation labeling method which has been applied successfully to similar matching problems in computer vision, natural language processing, and hypertext classification have been used for this purpose [26]. Relaxation labeling first computes the similarity scores solely based on the intrinsic features and later uses the extrinsic features in an iterative local optimization method, by changing label assignments to graph nodes until reaching a local optimum.

We argue that each approach has its own benefits and short comings and to gain a better understanding of the properties of each method, further study is required. For example, it seems that relaxation labeling can improve the final mapping for ontologies which extrinsic attributes play a more prominent role; this approach, however, cannot be employed in an incremental setup and would suffer from inconsistent scores. It is also very hard to define a good stopping criteria for relaxation labeling algorithms.

3.6 Experimental Results

In order to evaluate the proposed classification scheme, we have employed several open source software. Swoop is a hyper-media based ontology editor developed at MINDSWAP that allows creating and browsing OWL ontologies [62]. In our implementation, Pellet was used for ontology reasoning [95]. Pellet is an open source reasoner written in Java.

Our experimental method employed ontologies that are based on a pair of real world ontologies. The reported work is from two ontologies developed by separate organizations and separate goals. One ontology is from Karlsruhe [2], and is used in the Ontoweb portal. It defines terms used in bibliographic items and a university organization. The other ontology is from INRIA [1], and has been designed by Antoine Zimmermann based on the BibTeX in

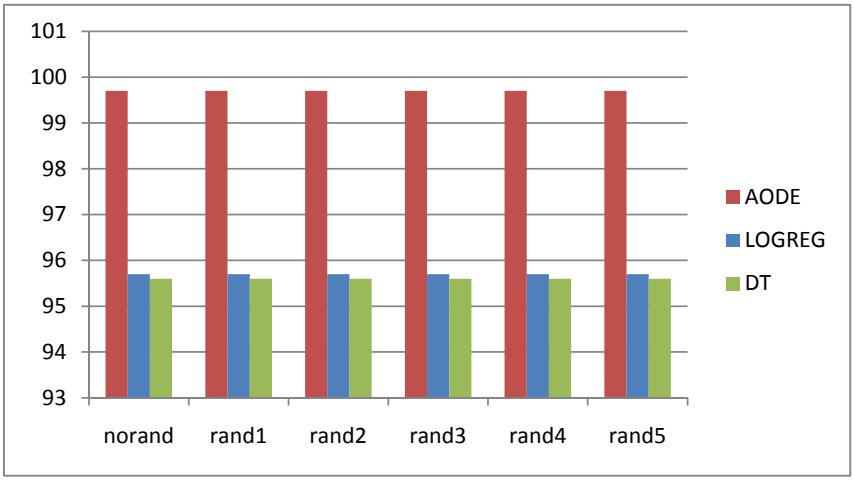
OWL ontology and the Bibliographic XML DTD. Its goal is to easily gather various RDF items. These items are usually BibTeX entries found on the web, which are transformed into RDF according to this ontology. The actual hierarchy of this ontology contains classes, which are subclasses of several other classes. The two ontologies have 24 corresponding classes. Figure 3.2(b) shows the characteristics of the ontologies in more details.

As described previously in Section 3.4, we have defined three similarity measures with which the scoring and matching is carried out. The name similarity compares the name of two classes by Jaro-Winkler's approximate string distance metric. The content similarity measures the similarity of the set of individuals of two classes using the Jaccard coefficient. The path similarity concatenates the name of classes traversed from root to the compared class and measures their Jaro-Winkler's distance.

To emphasize the discussion in Section 3.5, we note that the first two similarity measures are of intrinsic type and the last one is an extrinsic measure. These measures are only a sample of what can be realized and for each specific domain different similarity measures might prove useful.

One of the important considerations here is that, since the similarity measures will hardly ever convey the true semantic similarity of the underlying concepts, there would always be false positives and false negatives in our predictions. Thus, to make a correct mapping a human would always be needed in the mapping loop. However, the effort put into correcting a false positive is far less than the false negative by the human expert, since if an ontology has n concepts, the human expert would have to look into at most $\mathbf{O}(n)$ possibilities to correct a false positive. However, for a false negative there are $\mathbf{O}(n^2)$ possibilities to consider.

To evaluate the utility of using Average One Dependence Estimator, we have considered diluting the similarity score by adding the most uninformative similarity measures, i.e. uniform random numbers, incrementally. We tested our proposed method using a ten-fold stratified cross validation scheme. Figure 3.2(b) shows the Area Under the Precision Recall Curve (AUPRC) for each setting. The "norand" setting corresponds to the test in which no

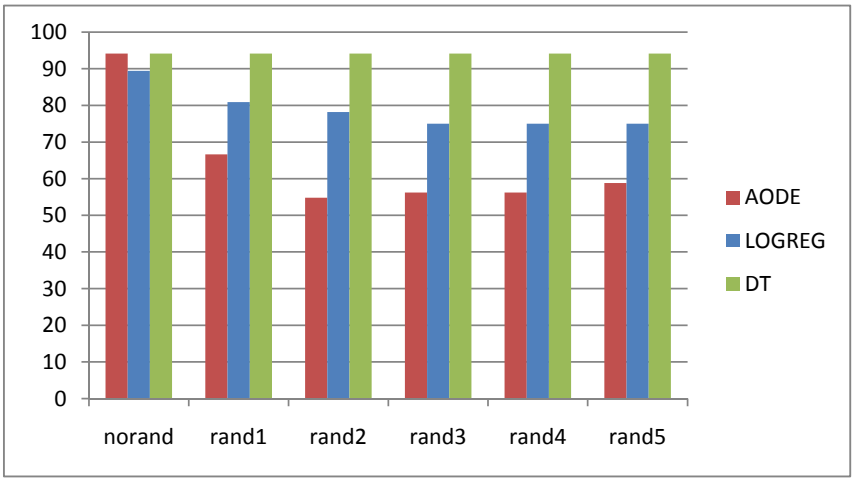


(a)

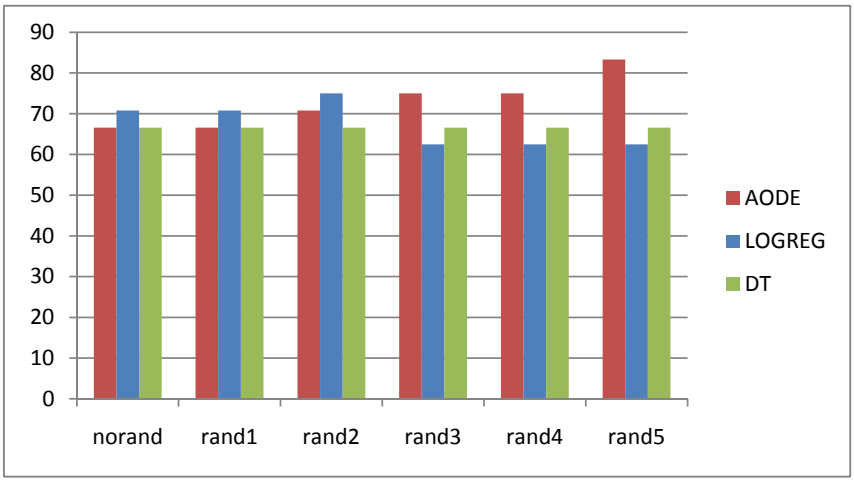
	OntoWeb	Bibliographic
# of Classes	64	68
# of Properties	72	58
# of Individuals	68	59
Min Depth of Class Tree	1	1
Max Depth of Class Tree	5	4
Min Branching of Class Tree	1	1
Max Branching of Class Tree	13	17

(b)

Figure 3.2: (a) Area Under the Precision Recall Curve (AUPRC). (b) Detailed description of two ontologies.



(a)



(b)

Figure 3.3: (a) Precision of three classification methods. (b) Recall of three classification methods.

random number has been added, and "rand1" through "rand5" are the settings where 1 to 5 extra random similarity measures has been added. The three bars represent three classification methods, Average One Dependence Estimator, logistic regression and C4.5 decision trees. Figure 3.3 shows the precision (a) and recall (b) for the same experiments. As can be seen, the AUPRC for Average One Dependence Estimator, which is a more accurate measure for evaluation, is consistently higher than other methods. Furthermore, as more inconsistent similarity measures are added, we can see less false negatives with Average One Dependence Estimator than other methods.

3.7 Conclusion

We have described a novel machine learning approach to ontology mapping. By discussing the notions of similarity, we have explained some of the common similarity measures used in the literature. In a sense, all the similarity measures can be divided into intrinsic and extrinsic measures. From an information theoretic point of view, all the similarity measures may contain valuable or contradictory information about the similarity of the concepts. By employing a probabilistic machine learning paradigm we assimilated all the valuable information and discarded the contradictions that lead to a single robust similarity score. The coherent way of integrating the similarity measures through a meta learning approach made it possible for us to propose a framework that is easily extendable. Furthermore, there is no need for precision recall curves to set the weights and thresholds of the system as this step is automated by recasting the ontology mapping problem to a classification problem. Since the similarity measures may not convey the true semantic similarity of concepts, there is always contradictory information among the measures. By diluting the measures and increasingly adding the most uninformative measures we tried to simulate this phenomenon and assess the goodness of this approach. As the results show, the meta learning algorithm is clearly superior than the other methods. This result is especially interesting since the training examples are highly unbalanced.

CHAPTER 4 AUTOMATIC DETECTION OF UNCERTAIN REGIONS IN SUPPORT VECTOR MACHINES

4.1 Introduction

In real-world classification problems, there is significant amount of uncertainty due to noise, insufficient data, and specific training and testing protocols. In many applications, such as clinical diagnosis, it is beneficial to explicitly recognize this uncertainty instead of trusting the classifier output and ignoring the problem altogether. Classification with rejection option is ideal in two scenarios:

i. For applications where test instances might be from novel classes not seen in any previous training data. For example, in microarray data where the patients gene expression profile might be from a new previously unknown cancer subtype.

ii. For applications where there are a series of increasingly more accurate but also more expensive tests to categorize instances belonging to a set of classes. The primary tests have a low cost, but produce features with low power in separating the classes, while the more complex tests would have more discriminatory power by producing better features with higher cost.

These types of setups are common in medical diagnosis. Here, the cost can be expressed not only in terms of the financial aspect but also on how invasive the test is. Obtaining a saliva or a blood sample is less costly than a biopsy. To minimize the overall cost and identify novel samples, we improve the classification of the primary classifier by introducing uncertainty. At each level, any instance that is declared uncertain requires a more powerful test.

The need for having a rejection option in classification problems has been previously recognized by us and others [21, 55, 78, 104, 112, 48, 10, 119, 50, 118]. However, an important limitation is the need to explicitly define the cost for misclassification and cost for

rejection. For most applications, coming up with exact definitions of the different costs is non-trivial, which is why classification methods with rejection option are not commonly used. In this work, we developed a procedure capable of automatically computing a reasonable data-dependent rejection threshold in lack of any further information. At the same time, this approach is flexible enough to let the user fine tune the threshold, in case expert domain specific knowledge about the related costs is available. Another benefit of using this method is that the additional computation required to set the threshold does not change the computational complexity of training a single SVM. This feature makes the approach appealing in applications that involve large data sets.

Section 4.2 is a brief introduction to support vector machines, the necessary transformations required to obtain calibrated posterior probabilities and existing work on classifying with rejection. In section 4.3 we describe how the uncertainty regions can be obtained using both posterior probability and geometric margin, and we introduce the method to automatically choose the uncertainty threshold. Section 4.4 contains experimental results on both artificial and real datasets, while the conclusions are drawn in Section 4.5.

4.2 Background

Support Vector Machines (SVMs) [14, 23] are widely used and are considered to be one of the most powerful learning algorithms in data mining. The goal of a SVM is to find a separating hyperplane that is located as far as possible from all training examples. The prediction output of SVM is not probabilitistic, and hence additional steps are required to obtain calibrated conditional probabilities.

4.2.1 Support Vector Machines

Consider the set $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$ where $\mathbf{x}_i \in \mathbb{R}^m$ is an m dimensional vector. A hyperplane is defined by $f_{\mathbf{w},b}(\mathbf{x}) = \mathbf{w}^\intercal \mathbf{x} + b$ where $\mathbf{w}^\intercal \mathbf{x}$ denotes the dot product of the two

column vectors $\mathbf{w}$ and $\mathbf{x}$. Consider the set of labels $\mathcal{Y} = \{y_1, y_2, \cdots, y_n\}$ where $y_i \in \{-1, +1\}$. In this set each y_i corresponds to the label of $\mathbf{x}_i$. The classifier $\text{sign}\left(f_{\mathbf{w},b}(\mathbf{x})\right)$ will directly classify $\mathbf{x}$ into $\{-1, +1\}$. SVMs find the slope $\mathbf{w}$ and offset b of this hyperplane for a set of linearly separable cases by solving the following constrained optimization problem,

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \tfrac{1}{2}||\mathbf{w}||^2 \\ \text{s.t} \quad & y_i\left(\mathbf{w}^\intercal \mathbf{x}_i + b\right) \geq 1,\ i = 1, \cdots, n. \end{aligned} \tag{4.1}$$

When the training set $\mathcal{S} = \{\mathcal{X}, \mathcal{Y}\}$ is not linearly separable, we relax the constraints by allowing to ignore some constraints for a penalty. This would result in the following *1-norm soft margin SVM* formulation,

$$\begin{aligned} \min_{\xi_i,\mathbf{w},b} \quad & \tfrac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i \\ \text{s.t} \quad & y_i\left(\mathbf{w}^\intercal \mathbf{x}_i + b\right) \geq 1 - \xi_i\,,\ i = 1, \cdots, n \\ & \xi_i \geq 0. \end{aligned} \tag{4.2}$$

4.2.2 Calibrating Probability Scores

After solving the optimization problem in Eq. 4.2, the $\text{sign}\left(f_{\mathbf{w},b}(\mathbf{x})\right)$ will be the prediction output of the SVM. However, this is not a probabilistic output. In some applications, having a high accuracy or a large area under the receiver operating characteristic (ROC) is not enough, and it is important to obtain accurate probability estimates. Efforts have been made to assess how well calibrated the output scores are for a given algorithm [24], and to create transformations that re-scale the scores back into calibrated probability estimates [122]. Two popular parametric and non-parametric approaches are Platt's scaling [83] and isotonic regression [86] respectively. A common feature of both methods is that the resulting transformations are monotonically increasing functions, which is what would be expected from such transformations.

Predicting good probabilities is not an easy task, since in practice the true conditional probability $\eta(\mathbf{X}) = P_{\mathbf{X},Y}(Y = 1|\mathbf{X})$ is never known. However, one sanity check is the requirement that the conditional probabilities be well calibrated. This means that for any interval of probabilities $[p_1, p_2]$, the probability of drawing a positive example given the classifier predicts $\hat{\eta}(\mathbf{X}) = \hat{P}(Y = 1|\mathbf{X}) \in [p_1, p_2]$ should also be in $[p_1, p_2]$. Previous studies have suggested that many classifiers, including naive Bayes and maximum margin methods, do not predict well calibrated probabilities [122, 83]. It has been shown that naive Bayes models that make simplistic assumptions about the probability structure push the posterior towards 0 and 1, while maximum margin methods such as SVM and boosted trees push away from the extreme probabilities [82]. In addition, it has been shown that Platt's scaling is effective for maximum margin methods, while it is less suited for naive Bayes [82]. Isotonic regression is also effective, but is inferior for smaller datasets. Since the calibration process requires internal cross validation, which makes the effective training data even smaller, Platt's scaling is preferred in SVM [82].

4.2.3 Classification with Rejection Option

Once the SVM optimization problem is solved (Eq. 4.2), the prediction of a test sample is given by $\text{sign}\,(f_{\mathbf{w},b}(\mathbf{x}))$. Hence, considering the example in Fig. 4.1, once the decision boundary is set between the two classes (crosses and circles), the two points A and B are going to be predicted as crosses. However, one should be more confident that A is a cross in comparison to B. Furthermore, the point C is more likely to be from none of the previously known classes. The prediction of B and C can be highly influenced by noise, or choice of training parameters. Currently, the output of classical SVM does not include any information allowing the user to make the distinction between A and B. Even though, as presented in previous section, the SVM output can be mapped to posterior probabilities, the question of where to draw the boundary between confident and less confident still exists.

The first seminal work that considers a rejection option, and given the true conditional

probability $\eta(\mathbf{X})$, formulates the optimal rejection rule, is the work of Chow [21]. Since then, rejection option has been considered for many probabilistic classifiers [55]. One of the early works for increasing the reliability of SVMs by thresholding can be found in [78], which computes an empirical distribution of the margin based on the training data and removes the top $\alpha\%$ of the margin closest to zero regardless of the label Y. Later applications of thresholding have confirmed the effectiveness of this approach for including a rejection option ([104, 112]). However, how to choose an optimal threshold has not been clear. Since choosing any threshold $\tau > 0$ from the output of the hinge loss is not *infinite sample consistent* (Classification-Caliberated) [9], the natural evolution was to develop surrogate convex loss functions that replace the standard hinge loss used in SVM to support a rejection option [10, 119, 50, 118]. However all existing methods require an *extended* cost matrix, or equivalently, require a threshold of the true conditional probability $\eta(\mathbf{X})$, to be explicitly defined *a priori*. Furthermore, these surrogate convex losses create twice as many constraints compared to the standard hinge loss, and as such require more samples to reach the same level of precision. Given that in most practical applications, coming up with an appropriate extended cost matrix is not trivial, it is worthwhile to forego infinite sample consistency for an adaptive threshold adjustment based on the empirical error.

4.3 Methods

All existing methods for building a classifier with a rejection option are limited by the need to *pre-define* an extended cost matrix (both the cost for misclassification and the cost for rejection). However, this process is not trivial and more often than not researchers in biomedical fields will overlook these methods and use the classification algorithms that do not require the definition of such a cost matrix. Our goal is to propose a method that will automatically detect a data-dependent threshold for rejection.

We propose two approaches to detect uncertainty regions (i.e., regions where sample

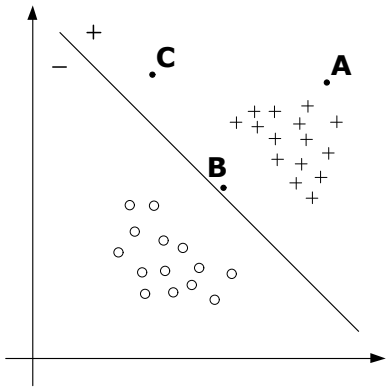


Figure 4.1: A simple example of a two dimensional binary classification problem. Once a hyperplane that separates the crosses (+) from the circles (∘) is found, any new test example falling on the positive side of the hyperplane will be classified as a cross, with no distinction made between points such as A and B. However, this is undesirable since the prediction confidence of A should be greater than that of B. Furthermore, C is likely to be from none of the two classes. A small change in the training protocol could alter slightly the position of the hyperplane and B could in fact be classified as a circle. We propose for the prediction of this type of examples to be marked as uncertain.

perdition will be rejected), one based on the geometric margin of the SVM and the other based on a posterior probability. Independent of the method to define uncertain regions, we propose an automatic method to detect the uncertainty threshold. These three approaches are described in the following subsections.

4.3.1 Uncertainty using SVM margin

Given a set of training examples $\{(\mathbf{x}_i, y_i)\}$ the *functional margin* between the instance $\mathbf{x}_i$ and the decision boundary is defined as

$$\hat{\gamma}_i = y_i \left(\mathbf{w}^\intercal \mathbf{x}_i + b\right) = y_i f_{\mathbf{w},b}(\mathbf{x}_i).$$

The confidence of the classifier's prediction of sample $\mathbf{x}_i$ is captured in the magnitude of $\hat{\gamma}_i$. However, this margin is dependent on the training data set and therefore is usually normalized. The normalization yields the *geometric margin* of $\mathbf{x}_i$ defined as

$$\gamma_i = y_i \left(\left(\frac{\mathbf{w}}{||\mathbf{w}||} \right)^\intercal \mathbf{x}_i + \frac{b}{||\mathbf{w}||} \right). \tag{4.3}$$

The geometric margin of the entire training set $\mathcal{S}$ is

$$\gamma = \min_{i=1,\cdots,n} \gamma_i. \tag{4.4}$$

Because the SVM maximal margin classifier is designed to maximize the geometric margin of the training set (γ) we proposed in [112] that all the test points that have a geometric margin smaller than γ to be considered as uncertain. Therefore, the uncertainty region is defined as

$$\mathcal{U} = \{\mathbf{x} \in \mathbb{R}^m | \ \ |\gamma_{\mathbf{x}}| < |\gamma|\}.$$

This however imposes a hard threshold which does not permit to incorporate specific costs for uncertainty or misclassification. We propose the addition of the threshold parameter τ that can be used to input the different costs using

$$\mathcal{U} = \{\mathbf{x} \in \mathbb{R}^m | \ \ |\gamma_{\mathbf{x}}| < \tau \cdot |\gamma|\}.$$

The specific value of the threshold τ above will be computed by the method described in Section 4.3.3 below.

4.3.2 Uncertainty using posterior probability

Since the output of SVM, $f_{\mathbf{w},b}(\mathbf{x})$, is not a probabilistic measure, one of the calibration methods discussed in Section 4.2.2 can be used to obtain probabilistic scores. Subsequently, the uncertainty areas can be defined using these probabilities. Here, we have opted for Platt's scaling since it is more efficient than the counterpart isotonic regression method [82]. Platt's scaling is a parametric calibration method which approximates the posterior probabilities by fitting the output of the SVM to a sigmoidal function of the form

$$P(Y = 1|\mathbf{x}) \approx P_{A,B}(f(\mathbf{x})) = \frac{1}{1 + \exp(Af(\mathbf{x}) + B)},$$

	prediction	
	incorrect	correct
uncertain	a	b
certain	c	d

Figure 4.2: To evaluate the effectiveness of using uncertainty we consider the contingency table between correct/incorrect prediction and uncertain/certain label. For us, the true positives will be a which represents the incorrect predictions that fall in the uncertainty regions. Hence, the precision, recall, and enrichment are defined in terms of a.

where $f(\mathbf{x})$ is the output of the classifier. The best parameter setting $z^* = (A^*, B^*)$ is found by minimizing the following cross entropy score (with N_+ positive and N_- negative examples)

$$\min_{z=(A,B)} \quad F(z) = -\sum_{i=1}^{n} \left(t_i \log(p_i) + (1-t_i)\log(1-p_i)\right)$$

$$\text{for} \quad p_i = P_{A,B}(f_i), \text{ and } t_i = \begin{cases} \frac{N_+ + 1}{N_+ + 2} & \text{if } y_i = +1 \\ \\ \frac{1}{N_- + 2} & \text{if } y_i = -1. \end{cases}$$

In our experiments, we used the Platt scaling [68] implementation available in the LIBSVM package [19].

We define our uncertainty regions based on the class-dependent posterior probabilities. Any sample with posterior probability bellow the threshold τ is uncertain,

$$\mathcal{U} = \{\mathbf{x} \in \mathbb{R}^m, y \in \{-1, +1\} |\ P(Y = y|\mathbf{x}) < \tau\}.$$

The specific value of the threshold τ above will be computed by the method described in Section 4.3.3 below.

4.3.3 Automatic detection of threshold

Both methods for defining uncertainty regions are dependent on the adequate selection of the threshold τ. This threshold represents the trade-off between obtaining uncertain or incorrectly classified samples, and is controlled by their respective costs. The problem we address in this section is the automatic detection of the data-dependent threshold.

In the process of selecting a threshold, our goal is to eliminate as many incorrect examples as possible (by declaring them as uncertain) without eliminating any correct samples. Hence, we define true positives as the incorrect samples that are in the uncertainty region (Fig. 4.2). The precision, in this case, is defined as the percentage of examples in the uncertain region that are incorrectly classified ($Prec = \frac{a}{a+b}$). The recall is the percentage of incorrectly classified examples in the uncertain regions as a fraction of the total incorrectly classified ($Rec = \frac{a}{a+c}$). Our goal is to simultaneously maximize the precision and recall. This can be achieved by maximizing the F_1 measure [109], which is the weighted harmonic mean of precision and recall ($F_1 = \frac{2 \cdot Prec \cdot Rec}{Prec + Rec}$). We automatically choose the optimal threshold τ, as the threshold that achieves the best F_1 measure over all possible threshold values determinable by the training set. In case of any additional domain specific knowledge available from an expert, the F_1 measure can be generalized to $F_\beta = \frac{(1+\beta^2) Prec \cdot Rec}{(\beta^2 Prec) + Rec}$, which offers the possibility of selecting a bias towards the precision or recall.

Another method for automatic selection of the threshold is to compute the enrichment of the incorrect examples in the uncertainty region. This can be achieved using the Fisher's exact test [40]. The probability of obtaining the observed number of incorrect samples in the uncertainty region just by chance follows a hypergeometric distribution

$$P(X = a) = \frac{\binom{a+b}{a}\binom{c+d}{c}}{\binom{n}{a+c}} = \frac{(a+b)!(c+d)!(a+c)!(b+d)!}{a!b!c!d!n!},$$

where $n = a + b + c + d$. The probability of observing more than a incorrect examples in the uncertainty region is $P(X > a) = 1 - \sum_{i=0}^{a} P(X = i)$. By minimizing this measure, we provide the uncertainty region that incorporates the most incorrect examples while also preserving the largest number of correctly classified examples, and therefore the best choice of threshold τ.

Regardless of the method used to automatically select the threshold, our framework determines uncertainty regions in the SVM prediction. We will refer to it as SVM with uncertainty

or USVM.

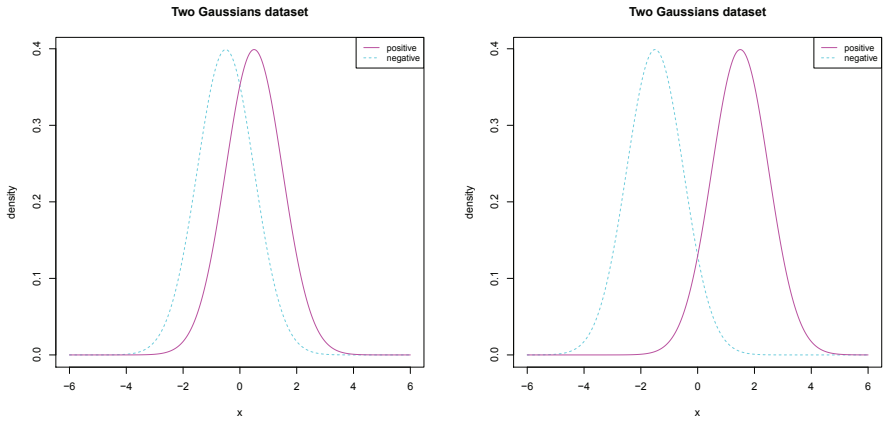
4.4 Results

We show through controlled experiments with artificial data that our method can automatically adapt to the inherent uncertainty in data. We further show that this approach can be useful in reducing the error rate in real data sets. The amount of reduction depends on how well SVM models the decision boundary and the dispersion of errors with respect to the decision boundary. We show that our automatic selection of uncertainty threshold successfully captures most of the errors while preserving the correct samples. These sets of experiments are described in the next two sub-sections.

Artificial data

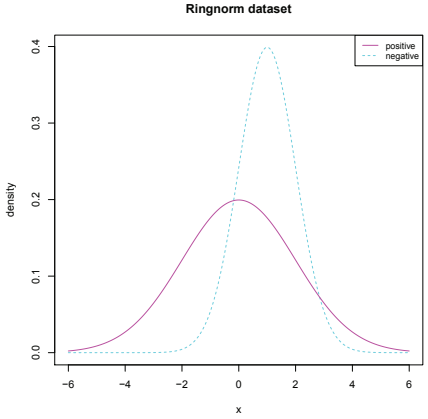
We used the artificial data to analyze the behavior of the uncertainty threshold under different degrees of overlap between the classes. We used two types of artificial data sets. The first type consists of two gaussians with the same standard deviation (σ) and various distances between their two means ranging from 0.5σ to 3σ. Two examples of feature generation are presented in Fig. 4.3(a) and Fig.4.3(b). This data set is called the *twonorm* data set. The second type of artificial data sets is also made up of two gaussians with one mode completely overlapping the other. This data set is called the *ringnorm* data set (Fig. 4.3(c)). For each artificial data set we generate 500 samples equally divided in the two classes and trained an L_2 regularized hinge loss SVM model with an RBF kernel. The best parameters (γ^*, C^*) were chosen using a grid search and five fold cross validation.

In addition to analyzing the performance, we also used the artificial data to show how the uncertainty threshold is obtained both based on the F_1 measure and Fisher's score, and to show the differences between computing the threshold on the geometric margin versus the posterior probability. One example of computing the threshold on the *twonorm* data



(a)

(b)



(c)

Figure 4.3: Class dependent distributions of each feature in the artificial data sets. Each feature is generated from two normal distributions with the same standard deviation (σ) ((a) and (b)). We considered eleven different distances between the means of the two distributions ranging from 0.5σ to 3σ. Two examples are presented in (a) and (b) for the distance equal to σ and 3σ respectively. In (c) a different type of artificial data set is considered, the ringnorm data set. In this case the standard deviations of the two normal distributions are no longer identical, they are 1 and 2 respectively, and the distance between the means is $1/\sqrt{d}$, where d is the dimensionality of the data set (in our case equal to 2).

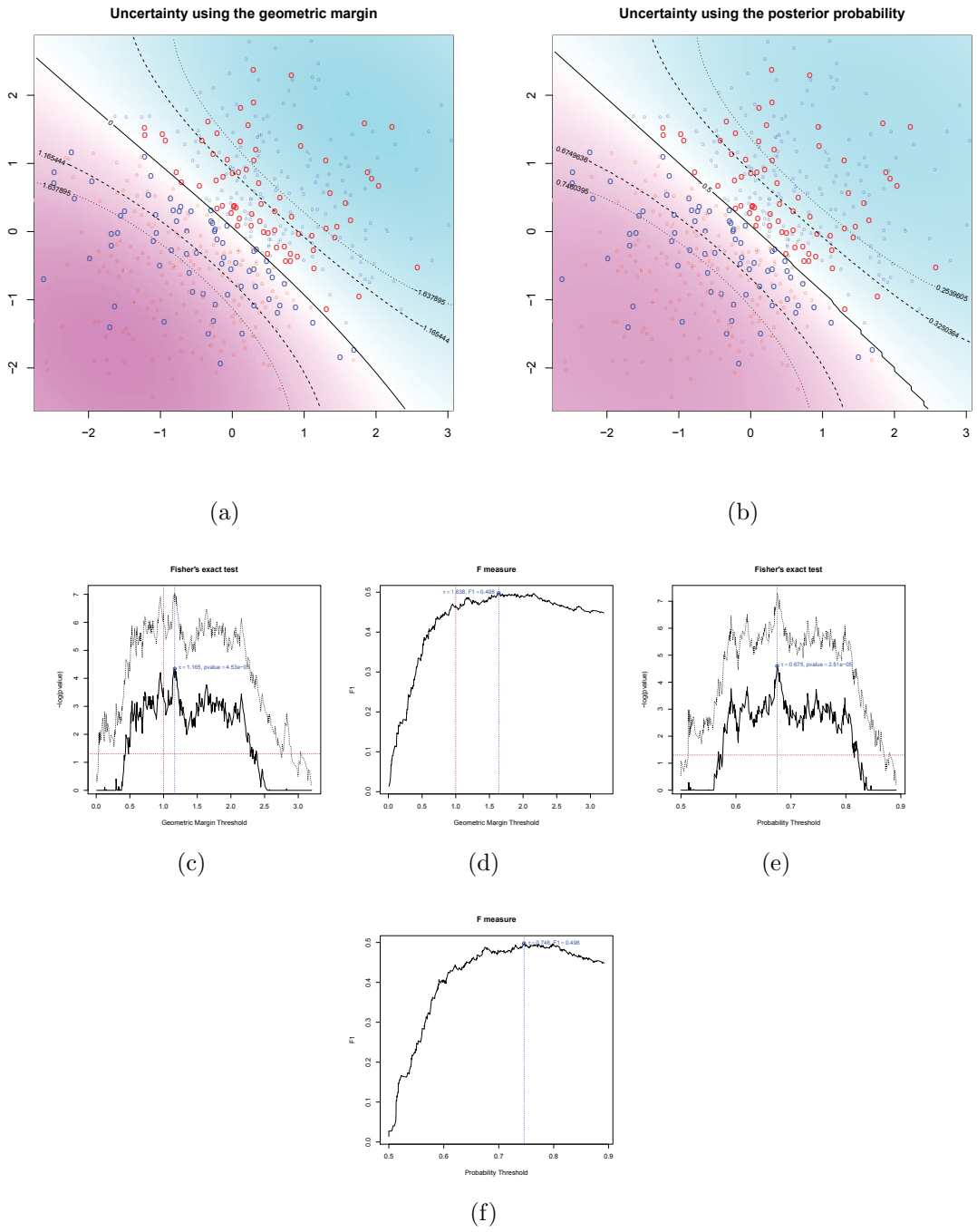


Figure 4.4: Side-by-side comparison of uncertainty region detection using geometric margin (a)(c)(d) and posterior probability (b)(e)(f) on the *twonorm* artificial data set with the distance between the means equal to 0.5σ, where σ is the standard deviation. The solid lines are the decision boundaries based on the geometric margin (a) and the posterior probability (b) respectively, while the shades represent the confidence of the discriminant function. The circles are the examples from the training set, with the bold circles representing the ones misclassified during training. The uncertainty thresholds are chosen as the maximum values in the trade-off graphs (c) and (e) for Fisher, and (d) and (f) for F_1. These optimal thresholds (τ) are marked with vertical blue lines in the trade-off graphs and with dashed (Fisher) and dotted (F_1) lines in (a) and (b). The trade-off graphs represent the range of all possible values of Fisher and F_1 respectively over the entire training set. By choosing the maximum on these graphs we optimize the uncertainty region to contain the most misclassified samples and the fewest correctly classified samples at the same time.

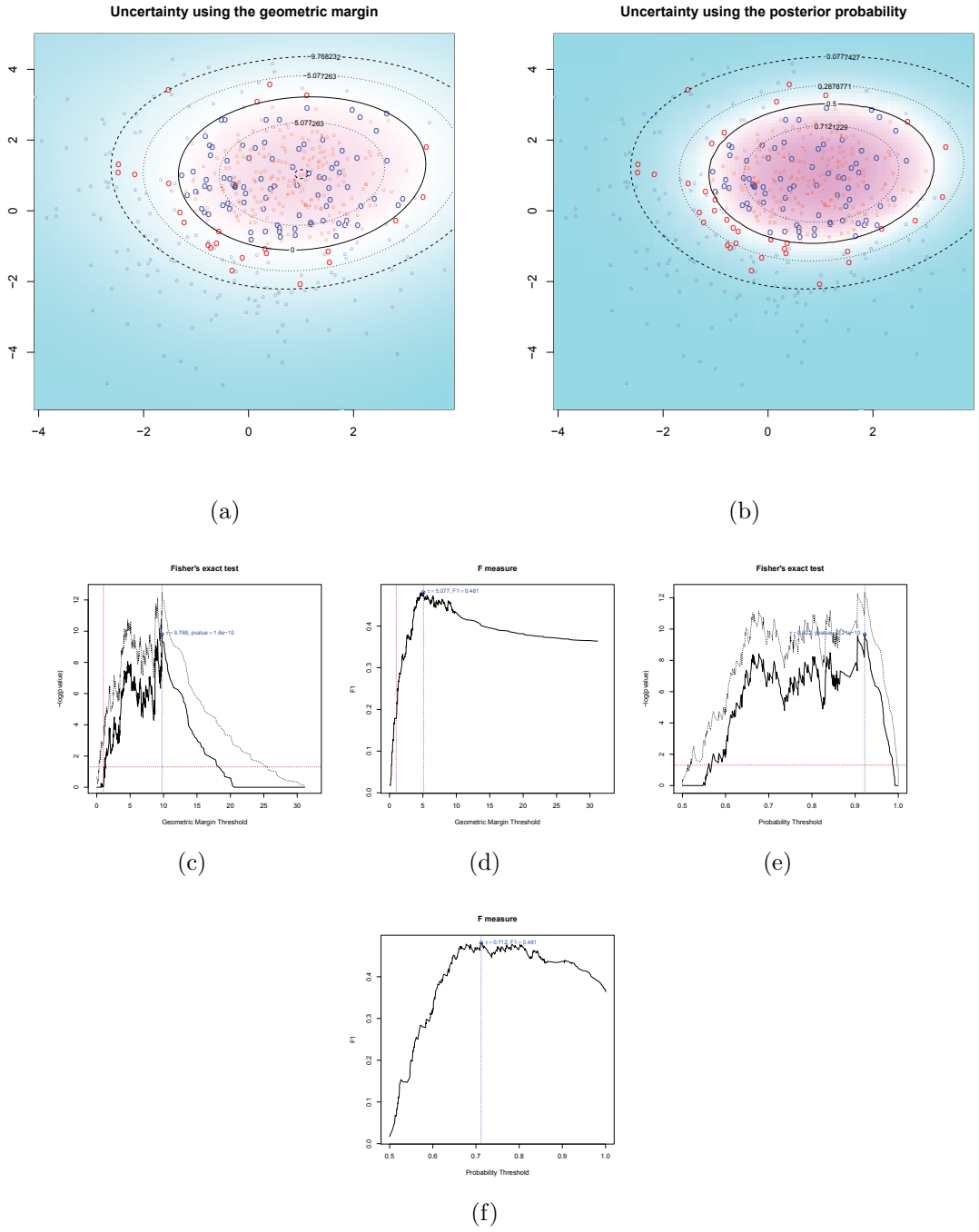
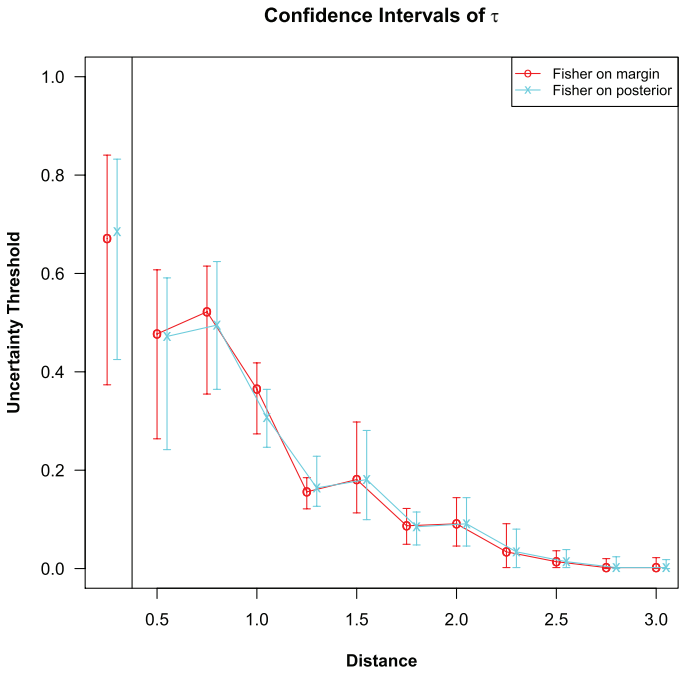


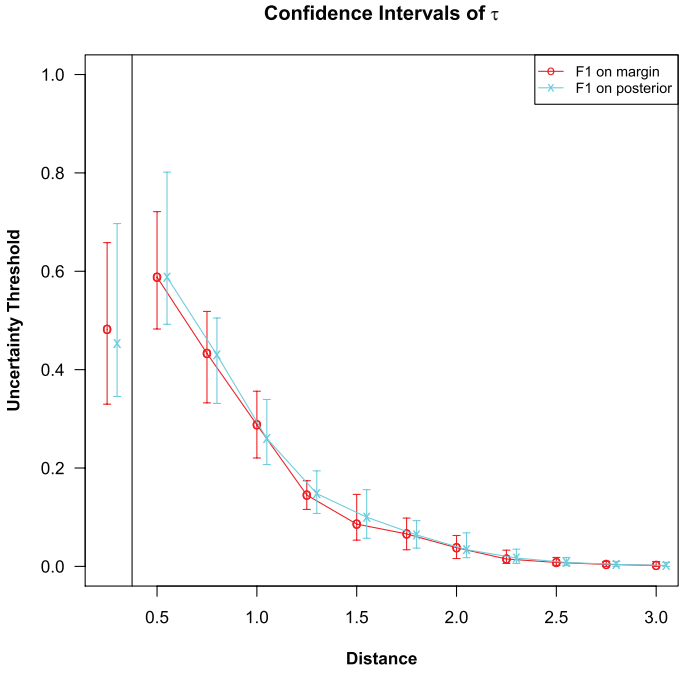
Figure 4.5: Side-by-side comparison of uncertainty region detection using geometric margin (a) and posterior probability (b) on the ringnorm artificial data set which contains two gaussians with one mode completely overlapping the other. The circles represent examples from the training set, while the bold circles represent the misclassified samples during training. The solid lines are the decision boundaries based on geometric margin (a) and posterior probability (b) respectively, and the shades represent the confidence of the discriminant function. The dashed (Fisher) and dotted (F_1) lines are the automatically chosen thresholds (τ) for the uncertainty region. The thresholds are chosen as the maximum values in the trade-off graphs (c) and (e) for Fisher, and in the graphs (d) and (f) for F_1. These maximum values are marked with vertical blue lines in the trade-off graphs. The trade-off graphs represent the range of all possible values of Fisher and F_1 respectively over the entire training set. By choosing the maximum on these graphs we optimize the uncertainty region to contain the most misclassified samples and the fewest correctly classified samples at the same time.

set is presented in Fig. 4.4 and for the *ringnorm* data set in Fig. 4.5. For each measure we compute all possible values across the entire training set, 4.4(c)-4.4(f) and 4.5(c)-4.5(f), and select the maximum value as the uncertainty threshold. For the Fisher exact test we used the negative logarithm of the p-value and hence the maximum is the most significant enrichment. These maximums are marked with vertical blue lines. Based on this example, there is no clear difference between using geometric margin (4.4(a) and 4.5(a)) and posterior probabilities (4.4(b) and 4.5(b)) to select the uncertainty threshold. However, in both cases the F_1 score (dotted lines) selects a more stringent threshold in comparison to the Fisher's exact test (dashed lines). Another difference between using the F_1 score and the Fisher's score for selecting the uncertainty threshold is the ability to set a significance threshold. This significance threshold (horizontal brown line in 4.5(c), 4.5(e), 4.4(c) and 4.4(f)) can be used to decide if uncertainty is needed or not. If the maximum Fisher's score is bellow this significance threshold using uncertainty is not warranted.

Using the *twonorm* data set we estimated the variance of the uncertainty threshold by generating 50 random data sets for each distance between the means. For each iteration we trained a model, computed the threshold and recorded the amount of data reported as uncertain during training. We used this data to determine the confidence intervals of the uncertainty threshold when using the *Fisher's score* in Fig. 4.6(a) and the F_1 score in Fig. 4.6(b). Both F_1 and *Fisher's score* capture the general monotonically decreasing uncertainty in the data sets with the increase in the distance between the means, with the F_1 measure being more conservative and rejecting fewer samples. In addition, there is no clear difference between selecting the uncertainty threshold on the geometric margin or the posterior probability. Since exact misclassification and rejection costs can rarely be defined explicitly, going through the extra step of calibrating posterior probabilities is not justified for selecting the uncertainty threshold. We applied the same procedure for the *ringnorm* data set (Fig. 4.6) and the same conclusions can be drawn: the F_1 score is more conservative and there is no difference in selecting uncertainty threshold between the geometric margin



(a)



(b)

Figure 4.6: Variance of the uncertainty threshold chosen based on Fisher's exact test (a) and F_1 test (b). The variance is assessed on the *ringnorm* data set (left graph of (a) and (b)) and on the *twonorm* data set (right side (a) and (b)) with distance between the means ranging from 0.5σ to 3σ. Both F_1 and Fisher's exact test capture the monotonically decreasing amount of uncertainty with the increase of the distance between the means. The F_1 measure is slight more conservative and rejects fewer samples. In addition, there is no difference in selecting the threshold on the geometric margin (red curve) or the posterior probability (blue curve).

and posterior probability.

The goal of the SVM classification is to predict the class label as accurately as possible. This implies the modeling of decision boundaries and not conditional probabilities. SVMs achieve this by bypassing the conditional probabilities using the *hinge loss*, thus focusing on predicting as accurately as possible near the decision boundary, while extreme points receive less attention. Although calibration procedures try to correct this problem, it is at the expense of further unwarranted computation. Since this region is where the most uncertainty lies, the goal of USVM to quantify the confidence is more in line with the goal of classification and the requirements of low complexity and sparsity than the goal of probabilistic prediction. Essentially predicting accurate probabilities is a much harder problem than classification itself, and sometimes it might not be worthwhile to predict the probabilities accurately in the entire feature space. Our experiments suggest that calibration of SVM outputs leads to comparable results near the decision boundary with further computational cost, which is undesirable for a rejection option when costs are unknown.

Real data

We compared the performance of USVM with SVM on the *MLL* gene expression data [6] for the first scenario and on three data sets from the UCI machine learning repository [7] for the second scenario. Armstrong et al. [6] show that Acute lymphoblastic leukemias carrying a chromosomal translocation involving the mixed-lineage leukemia gene (MLL, ALL1, HRX) have different expression profiles from conventional acute lymphoblastic (ALL) and acute myelogenous leukemias (AML). We use this data set to show that USVM can detect the novel MLL subtype from the conventional ALL and AML. The second data set is the Pima Indian diabetes data set (referred to as *pima*), which is a binary classification task with eight features and 768 samples. The third data set is the large scale high-dimensional *mnist* digit dataset [65]. It consists of 60,000 training and 10,000 testing samples of hand written digits. There are ten classes with approximately equal number of instances in each class. Each

instance is a 28×28 pixel image, thus 784 features in total. All images are centered, and their scale and rotation normalized. The last example is the UCI Cardiotocography dataset (CTG) which contains 2126 fetal cardiotocograms with 39 features. This dataset is specially relevant for detecting uncertain samples, since the fetal state, which is to be classified, is labeled as Normal, Pathologic and *Suspect*. Our premise is that by training only on the Normal against Pathologic samples our method is able to identify during the prediction the *Suspect* samples. For all data sets we use the RBF kernel and choose the bandwidth γ, penalty C and the uncertainty threshold τ using cross-validation on the training set. The model for *pima* is tested during cross-validation and the model for *mnist* tested on the test set for *mnist*. We tested the model for CTG both during cross-validation and on the test set that consists only of the *Suspect* samples.

The *MLL* data set contains gene expression profiles of three leukemia subtypes (ALL, AML and MLL), each class containing (24, 28 and 20) samples respectively. The data from all classes were normalized together using Robust Multiarray Average (RMA) [58] and probesets with the same gene symbol were averaged or dropped when no symbol was available for a total of 8655 genes. Subsequently, the MLL class was set aside for testing and USVM was trained on samples containing ALL and AML groups. The training was performed on 11 genes selected using the Nearest Shrunken Centroid [102] method only on the training data. From the 20 MLL test samples, 13 were rejected. To see where the 20 samples lie in this 11 dimensional feature space, we draw a scatter plot of the first two principal components in Figure 4.7. As can be seen in this figure, the ALL (red diamonds) and AML (blue circles) classes group together closely, while samples from the novel MLL subtype (orange, grey and cyan circles) are further apart from the training groups. The grey circles are the rejected samples which constitute 65% of the total novel subtype, while the orange and cyan circles are classified as ALL and AML respectively.

On the *pima* data set, the uncertainty threshold is chosen such that the region of uncertainty is highly error prone (40.37%) in comparison with the acceptance region (13.05%).

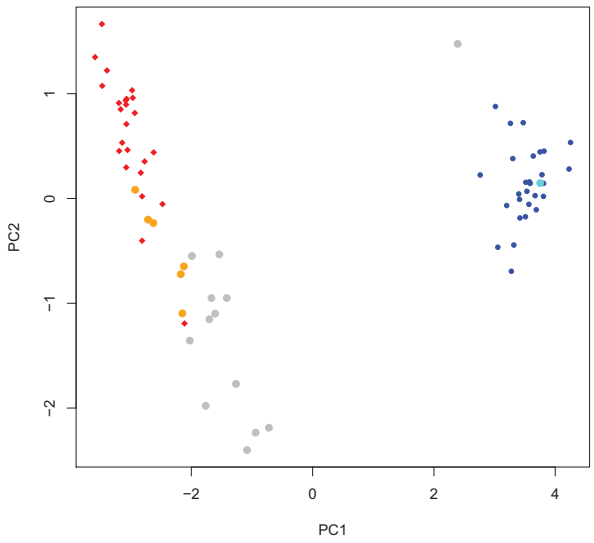


Figure 4.7: A scatter plot of the first two principal components of the MLL data set. The smaller symbols of red diamonds and blue circles are the ALL and AML training data respectively. The bigger circles are the MLL test data either predicted as ALL/AML (orange/cyan) or rejected (grey).

In terms of overall performance, USVM reduces the error rate by approximately 44% over the standard SVM. The cost of this performance increase is the percentage of samples that where declared uncertain - 35% of the test set. These results are summarized in Table 4.1. In addition, in Fig. 4.8, we present the evolution of the USVM accuracy (solid curve) and uncertainty percentage (dashed line) for all the possible thresholds on the geometric margin. Based on this figure, it can be noticed that the selection of the threshold is subjective and based on prior knowledge of the data set it can be chosen more or less stringent. However, USVM automatically selects the threshold that achieves the best odds ratio of errors in the uncertainty region.

The *CTG* dataset is particularly suitable for displaying the benefits of including uncertainty in the SVM prediction since it contains samples that are labeled as uncertain (*Suspect*) by expert physicians. We trained the classifier on Normal versus Pathological cases, and examine how it performs on the Suspects cases. Although during cross-validation both SVM and USVM achieve very good error rates (1.3% and 0.3% respectively), when tested against the *Suspect* cases USVM is able to identify 21.36% of them while SVM will always assign

Table 4.1: The error rate of the SVM prediction is reduced by approximately 44% when using the F_1 automatic method to compute the uncertainty threshold on the pima data set. This is achieved by discarding the points in the uncertain region which is characterized by a higher error rate.

	SVM	USVM	Uncertainty region	
			error rate	percentage discarded
D vs. H	22.66%	13.05%	40.37%	35.16%

Table 4.2: Both SVM and USVM perform very well on the CTG dataset. Even though only a small percentage of samples are declared uncertain, this margin of uncertainty identifies 21.36% of the *Suspect* cases while SVM is unable to identify any.

	SVM	USVM	Uncertainty region	
			error rate	percentage discarded
P vs. N	1.3%	0.3%	41.67%	1.91%

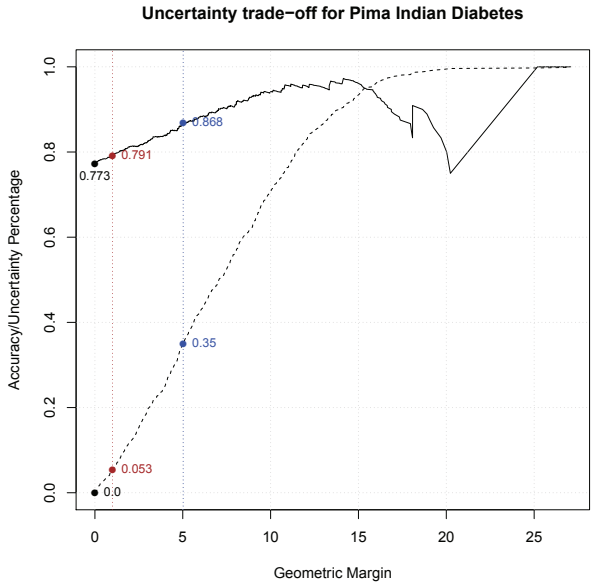


Figure 4.8: Evolution of the accuracy (solid curve) and uncertainty percentage (dashed curve) with the increase in geometric margin. The vertical lines represent the standard geometric margin γ (brown) and the optimal threshold chosen using the F_1 score (blue). The standard SVM accuracy and uncertainty are represented by the black points (77.30% and 0% respectively), while USVM accuracy and uncertainty by the blue points (86.8% and 35% respectively).

them to Normal or Pathological. These results are summarized in Table 4.2.

For the *mnist* data set, we decomposed it into binary classification problems. We performed the comparison for each digit against the rest. For each one of the ten classifiers, USVM reduces the error rate by approximately half by only declaring on average 2% of the

Figure 4.9: Sample of errors declared as uncertain by USVM on the mnist data set. Each row is the result of the discrimination between digits 0 through 9 against all the remaining digits. The left side represent false negatives (target digit samples that would have been missclassified without uncertainty) and the right side are false positives (other digit samples that would have been missclassified without uncertainty). The empty spaces are due to lack of more false negatives for classes 0 and 4.

test data to be uncertain (Table 4.3). This is achieved by choosing the uncertainty region such that it is highly error prone. For example, for the digit 8, 75% of the samples in the uncertainty region are errors. We present in Fig. 4.9 a set of samples that where declared uncertain by USVM. Each row represents one of the ten digit classifiers with the left side containing false negatives and the right side false positives.

4.5 Conclusion

The widely used SVM classifiers essentially make guesses for those points of the input space situated very close to the decision boundary. In spite of the fact that these guesses are well-informed and shown to minimize the error in lack of any additional information [111], in applications where refusing to classify an instance due to uncertainty would cost less than incorrect classification, it is important to identify regions of feature space that predictions are highly uncertain. The uncertain regions can also aid in predicting novel classes, such as new

Table 4.3: For all the class specific models on the mnist data set the error rates of the SVM predictions are approximately halved when using F_1 automatic method to compute the uncertainty threshold. Note the high percentage of errors discarded by using the uncertainty region.

	SVM	USVM	Uncertainty region	
			error rate	percentage discarded
0 vs. ALL	0.81%	0.33%	59.26%	0.82%
1 vs. ALL	0.45%	0.12%	35.48%	0.93%
2 vs. ALL	1.66%	0.84%	57.75%	1.44%
3 vs. ALL	1.93%	0.82%	47.01%	2.39%
4 vs. ALL	1.69%	1.01%	55.28%	1.25%
5 vs. ALL	2.01%	0.93%	34.82%	3.19%
6 vs. ALL	1.01%	0.56%	60.81%	0.75%
7 vs. ALL	1.81%	0.78%	47.47%	2.21%
8 vs. ALL	3.18%	1.71%	75.65%	1.99%
9 vs. ALL	3.11%	1.71%	44.51%	3.29%

cancer subtypes, unbeknownst to the classifier at training time. Due to the nature of this problem, there is an inherent tradeoff between rejection and incorrect classification which leads to a curve similar to the ROC curve. We propose two measures to automatically select a reasonable data dependent threshold for rejection. This can alleviate the user from ignoring uncertainties that commonly occur in practical applications with noisy data. The simulations and experiments show that selecting the uncertainty threshold based on the geometric margin is equally accurate compared to selection based on calibrated probability scores. Therefore the extra effort needed to compute calibrated probability scores is unwarranted.

CHAPTER 5 LEARNING A ROBUST DIAGNOSTIC SIGNATURE FROM LOW-SAMPLE HIGH-DIMENSIONAL DATA

5.1 Introduction

Every day we are capable of gathering and storing more and more data. This is especially true in bioinformatics, where high-throughput experiments produce a massive number of features for every sample. Traditional methods of regression and classification have not been able to keep up with this growth in number of features. As a result, there is great interest in finding feature selection/ranking approaches that identify and remove irrelevant features [51, 18, 98].

The ultimate goal of a diagnostic model is to predict well on unseen (test) data. However, due to the nature of gene expression data that has orders of magnitude more features than samples, applying classification methods directly, without feature selection, can result in poor prediction performance. As such, many feature selection methods have been proposed to remove or reduce the impact of irrelevant features [89]. Based on the interconnection of feature selection and classification methods, feature selection approaches have been historically divided into filter and wrapper methods. Filter approaches score each feature individually, while wrapper approaches score feature subsets according to their predictive performance through a classifier. Wrapper approaches become computationally prohibitive very quickly with increasing number of features, since the search space of the power set is exponential in the size of the feature set. Therefore, wrapper methods are approximated with a greedy search over heuristically ranked feature sets. A third category of approaches, known as embedded feature selection, has emerged recently, which sidesteps this problem by relaxing the combinatorial discrete problem to a continuous convex optimization, thereby performing feature selection and classification at the same time. In embedded approaches, the regularizer

component of the classifier, which controls the complexity of the model, is designed to favor sparse models.

Many advances in sparse learning and approximation, both in theory and methodology, have made it possible for embedded approaches to solve high-dimensional problems efficiently [101, 38, 128, 129, 114]. These approaches attempt to directly identify important features and classify the samples at the same time. This is achieved by using an l_1-norm component, when formulating the penalty. When $\mathbf{x} \in \mathbb{R}^n$ is a vector space, the l_1-norm of $\mathbf{x}$ is defined as $\|\mathbf{x}\|_1 = \sum_i |x_i|$. Unlike the smooth l_2-norm $\|\mathbf{x}\|_2^2 = \sum_i x_i^2$, which is used for example in ridge regression, the l_1-norm is non-smooth at zero. This enforces many of the small coefficients to become exactly zero, and therefore be removed from the model altogether.

Under certain regularity conditions, the l_1-penalty exactly recovers the correct feature set. These conditions typically limit the number of non-zero coefficients, and pairwise correlations between features [113]. When there is high pairwise correlation, the elastic net penalty groups correlated variables together, and uses the strength of the group to reduce the noise [129, 114]. However, the l_1 component of the elastic net penalty makes this method unstable. It has been shown that resampling methods can improve the stability of the l_1 penalized methods, providing consistent feature selection under weaker conditions [8, 73, 90].

In this paper, we combine a sparse classifier and a randomization procedure, to learn a robust diagnostic signature from gene expression data. The base classifier, which uses the elastic net penalty, is the so called Hybrid Huberized Support Vector Machine (HHSVM) [114]. The randomization procedure, is the stability selection method [73], following the sampling schedule of Shah and Samworth [90]. We refer to our combined method as *Robust Feature Selection* (RFS). We compare RFS with a number of popular and state of the art feature selection methods for low-sample high-dimensional datasets. These methods were chosen, as they were used in a recent comprehensive comparison [98] and an international competition [99].

We compare RFS with a family of dependence maximization approaches that include many popular univariate and multivariate feature selection methods as a special case. This family takes advantage of the Hilbert-Schmidt Independence Criterion to compute the dependence between the response and input [98]. It includes Pearson Correlation, mean difference and variations such as t and moderated t-statistic [97], the Centroid method [11], etc., as well as SVM RFE [52], along with the Nearest Shrunken Centroid method [102]. SVM RFE is a popular feature selection approach for high-dimensional data and the Centroid method is the best performing feature selection method, as reported in a recent comprehensive review [98] (see Table 3 and 4). We compare RFS to a variation of the Centroid method, which is the combination of moderated t-test [97] and fold change. This method was the overall winner of the Improver Diagnostics Signature Challenge [74]. Finally, we also compare RFS with the base HHSVM classifier.

We compare the above methods methods on 13 datasets collected from the Improver Challenge and other publicly available microarray datasets. We show through extensive comparisons that RFS significantly outperforms previously known methods in *predictive accuracy*, while improving the *stability* of selected features at the same time.

We start by describing the Improver Diagnostics Signature Challenge in Section 5.2. In Section 5.3, we describe the required preprocessing steps for gene expression data, namely sample selection, quality control and normalization. In Section 5.4, we describe the state of the art methods of feature selection and classification in gene expression data, which have been quite successful, as demonstrated by the winning team in the Improver Challenge. Then we elaborate on our Robust Feature Selection (RFS) method. In Section 5.6, we describe the scoring metrics used in the Improver Challenge and empirically evaluate RFS, and conclude in Section 5.7.

5.2 Improver Challenge

Industrial Methodology for PROcess VErification in Research (IMPROVER), is a joint project of IBM Research and Philip Morris International R&D [74, 76]. The aim of this project is to assess the extent and effectiveness of systems biology approaches in industrial settings. The hope is envisioning a future where these methods replace the empirical long-term but trustworthy clinical trials. The first phase of this project, named the diagnostics signature challenge, ended in Oct. 2012 [100]. The purpose was to validate the effectiveness of computational models that classify clinical samples based on transcriptomics data, by crowdsourcing the problem through an international competition [75, 99].

Challenge participants were asked to provide predictive diagnostic signatures on four disease areas: Psoriasis, Multiple Sclerosis (MS), Chronic Obstructive Pulmonary Disease (COPD) and Lung Cancer. Fifty four teams participated from around the globe, most of which submitted results to all sub-challenges.

The IBM scoring team suggested several scoring and aggregation criteria to an outside scoring panel and after several iterations the outside panel selected the final performance metrics. However, only general guidelines where given to the participants and the final metrics where not divulged. All team names where anonymized by the intermediary IBM team executing the scoring process from the outside scoring review panel. The details of the scoring and aggregation metrics are discussed in Section 5.6.

The unlabeled test samples were provided by the organizers, however, training samples were to be found from public databases such as the Genomic Expression Omnibus (GEO) [32] or any private/proprietary databases by the participants. Due to this, data selection and preprocessing became an important factor in establishing a wining strategy.

To reduce different sources of heterogeneity and noise, and focus only on feature selection and classification algorithms, we use the same datasets and preprocessing methodology of the overall winning team. The preprocessed and normalized data from the Improver Challenge

were kindly provided to us by the winners of the diagnostics signature challenge [100]. The overall main theme that made the winning strategy was to keep the data selection, methods and models as simple as possible in every step. For example, two of the models had only two genes. In the rest of this paper we describe the methods used by the winning team and compare it to our RFS method for feature selection and classification. We further evaluate this method on several external datasets.

5.3 Preprocessing Microarray Data

The first step in any microarray experiment after hybridization is preprocessing. Although often ignored and treated as a black box, due to many sources of heterogeneity and noise, careful preprocessing can make a huge difference in the end results. Since training data was not provided by the organizers, there is an extra pre-step of selecting the training data from public genomic expression databases, such as GEO and Array Express. The suggested training datasets were from different labs, microarray platforms and batches. To reduce variability that is not due to biological effects, only datasets that were from the same platform as test data were chosen when possible. All test data were from single channel Affymetrix arrays.

Preprocessing of microarray data consists of the four steps of image quantification, background correction, normalization and summarization. Although the process is divided into these four steps and are performed in that order, they are interconnected in that any error from one step propagates to the next. Image quantification is usually performed by proprietary software from the chip makers, however, there are numerous methods for the other steps proposed in the literature that improve upon the proprietary software. A comparison of these methods [59] revealed that background correction has the most influence on the end results, and that methods that consider the interconnection of these steps are superior to the ones that consider them in isolation.

One of the widely used methods that incorporates all these steps is the Robust Multi-array Average (RMA) developed by [58]. Background correction is usually performed by correcting for background noise and processing effects, adjusting for cross hybridization and correcting the range of the expression values. In RMA, mismatch probes (MM) that are intended to measure the cross hybridization effect are ignored and deemed unreliable [79], and the observed target is assumed to be the sum of an exponential signal and a Gaussian background noise.

The purpose of normalization, the next step in the pipeline, is to reduce variability due to non-biological effects, such as experiments performed in different labs, conditions and batches. A comparison of normalization methods for high density oligonucleotide arrays can be found in [13]. RMA uses quantile normalization [13] for this step of the process. In summarization, the final step, multiple probes of each probeset are combined to produce an expression value. RMA uses the median polish algorithm [36] to robustly fit the expression values based on a multi-array model.

When the samples are from different platforms, the standard RMA can no longer be used. In this situation, all datasets that are from the same platform are normalized separately using RMA, followed by mapping probesets to genes (Enterez IDs), averaging over many to one mappings and dropping probesets with no Enterez ID mappings while keeping only genes present in both platforms. Finally all sample are normalized together using quantile normalization [13] to remove batch effects. In all datasets from the challenge, after normalizing with RMA, MAS5.0 [5] detection calls were used to drop probesets that were not reliable in a reasonable (25%) number of samples.

5.4 Feature Selection and Classification

Similar to the *No Free Lunch* argument about the lack of existence of an optimal classifier for all conceivable datasets, the same argument is true for an optimal feature selection

method. However, we are interested in methods that typically perform better on average for certain high-dimensional datasets, which can be assessed empirically. A generalization of many feature selection methods that subsumes filter and wrapper approaches such as Pearson Correlation, the Centroid [11], moderated t-test [97], SVM RFE [52] and Nearest Shrunken Centroid [102], along with a comprehensive comparison of the popular approaches can be found in [98]. In this framework, feature selection which is a combinatorial optimization task, is converted to a greedy procedure, optimizing the Hilbert-Schmidt Independence Criterion (HSIC). The basic premise here is that good features should be highly dependent on the labels. Based on this study which includes 28 microarray datasets, the Centroid method, also known as absolute fold change when expression values are log-transformed, has the overall best performance in both binary and multiclass problems.

Let $\mathbf{x} \in \mathbb{R}^{n,p}$ be a matrix of n samples and p features, and $\mathbf{y} \in \{+1, -1\}^n$ a vector of labels. The Centroid method [11] uses $\nu_j = \bar{x}_{j^+} - \bar{x}_{j^-}$ as the score for feature j, where j^+ and j^- are index sets over the positive $j^+ = \{(j,i)|y_i = +1\}$, and negative $j^- = \{(j,i)|y_i = -1\}$ samples, and features are selected according to the absolute value $|\nu_j|$. When x is the log expression value, this quantity is also known as absolute fold change.

The moderated t-test [97] uses an empirical Bayes approach to attenuate the normalization of each feature. Compared to the t-test, moderated-t estimates the distribution of variance, by pooling the variances across all features. This distribution is subsequently used to normalize the variance of all features. If $\bar{s}_j = (\frac{s^2_{j^+}}{m_+} + \frac{s^2_{j^-}}{m_-})^{\frac{1}{2}}$ is the t-statistic, where m and s are the empirical mean and standard deviation, the moderated t-test is

$$\tilde{s}_j = \frac{m\bar{s}_j^2 + m_0\bar{s}_0^2}{m + m_0}, \tag{5.1}$$

where $\bar{s}_j$ is the t-statistic and $\bar{s}_0$ and m_0 are the hyperparameters for the prior distribution

on $\bar{s}_j$, which are estimated using information from all dimensions

$$\begin{aligned} m_0 &= 2\Gamma'^{-1}\left(\frac{1}{d}\sum_{j=1}^{d}(z_j-\bar{z})^2 - \Gamma'\left(\frac{m}{2}\right)\right), \\ \bar{s}_0^2 &= \exp\left(\bar{z} - \Gamma\left(\frac{m}{2}\right) + \Gamma\left(\frac{m_0}{2}\right) - \ln\left(\frac{m_0}{2}\right)\right) \end{aligned} \tag{5.2}$$

where $\Gamma(\cdot)$ is the gamma function, $'$ denotes derivative, $z_j = \ln\left(\bar{s}_j^2\right)$ and $\bar{z} = \frac{1}{d}\sum_{j=1}^{d} z_j$.

The winners of the Improver Challenge used a combination of moderated t-test and Centroid methods for feature selection, along with Linear Discriminant Analysis [46] as the classification approach. They first reduced the feature set using the moderated t-test to all features having a p-value less than 0.005, and subsequently sorted them by decreasing absolute fold change value. Starting from top 2 features up to 10~20 depending on the dataset, they went down the list one feature at a time and chose the set with maximum $tAUROC$ of the classifier, where $tAUROC = \text{mean}(AUROC)/\text{sd}(AUROC)$ computed over five-folds.

5.5 Robust Feature Selection

As opposed to the described filter methods we employ an embedded approach. The Hybrid Huberized SVM [114] is a sparse classifier specifically designed for high-dimensional low-sample datasets. To understand the differences that makes HHSVM a suitable classifier, we start with the familiar SVM [14] optimization criterion. When $\mathbf{x}$ is a column vector sample and $y \in \{+1, -1\}$ is the label, SVM minimizes the risk function,

$$\min_{\beta_0,\boldsymbol{\beta}} \sum_{i=1}^{n} \left[1 - y_i\left(\beta_0 + \mathbf{x}_i^T\boldsymbol{\beta}\right)\right]_+ + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2, \tag{5.3}$$

where the loss function $(1-\cdot)_+ = \max(1-\cdot, 0)$ is called the *hinge loss* and the regularizer is the l_2-norm of the coefficients $\boldsymbol{\beta}$. The HHSVM changes the regularizer to a mixture of l_1

and l_2-norms known as the *elastic net penalty* [129],

$$\lambda_1||\boldsymbol{\beta}||_1 + \lambda_2||\boldsymbol{\beta}||_2^2, \tag{5.4}$$

and the loss function to *huberized hinge loss* [87]

$$\phi(yf) = \begin{cases} 0, & \text{for } yf > 1, \\ (1-yf)^2/2\delta, & \text{for } 1-\delta < yf \leq 1, \\ 1-yf-\delta/2, & \text{for } yf \leq 1-\delta, \end{cases} \tag{5.5}$$

where $\delta > 0$ is a prespecified constant. These changes result in the following HHSVM optimization criterion,

$$\min_{\beta_0,\boldsymbol{\beta}} \sum_{i=1}^{n} \phi\left(y_i\left(\beta_0 + \mathbf{x}_i^T\boldsymbol{\beta}\right)\right) + \lambda_1||\boldsymbol{\beta}||_1 + \lambda_2||\boldsymbol{\beta}||_2^2. \tag{5.6}$$

The elastic net regularizer encourages a combination of sparsity through the l_1-norm (λ_1) and grouping of correlated features through the l_2-norm (λ_2), while the huberized hinge loss enables faster computation by smoothing the hinge loss to make it always differentiable. Furthermore, a pathwise [33] method is used to solve (5.6) for every possible value of the regularization parameter λ_1. As such, the entire solution path of λ_1 is found for the same cost of solving this model for a single λ_1.

Although sparse classifiers with lasso and elastic net penalties are asymptotically consistent, they are sensitive to the regularization parameters in finite sample settings [8]. The stability selection method [73] is a general randomization procedure via subsampling to stabilize approximation of discrete structures, such as feature selection and inverse covariance estimation. Stability selection, provides a stronger guarantee for consistent feature selection under weaker conditions, than what is provided by the original sparse penalties, with finite sample error control. We compute an individual selection probability for each one of the fea-

tures using repeated subsampling of 50 percent without replacement according to [90]. The probability is computed based on the number of times the respective feature was selected in the trained model. The final classifier is trained on top k ranking features, where the rank is based on the selection probability.

Note that one should refrain from using linear kernel SVM on the reduced set of features, since SVM approximates the risk using only the support vectors, and discards most of the already small amount of available samples. Although we use Linear Discriminant Analysis (LDA) for the final classifier, using logistic regression would yield similar results.

5.6 Results

We compare the prediction performance and stability of the diagnostic signature found using RFS with HHSVM, Pearson Correlation (PC), moderated t-test and absolute fold change combination (mtfc), SVM RFE and Nearest Shrunken Centroid (NSC) on 13 microarray datasets. All datasets were chosen to be binary classification tasks, so that the final results would be comparable. For every dataset, the top k ranking features are computed for all methods, where $k \leq 20$. An LDA classifier is trained on the same folds where feature selection is performed, and tested on the hold-out set using 10 fold cross-validation, for every k-feature model ($k \leq 20$). Subsequently, various performance measures were computed on the hold-out set.

In this section, we first describe the performance metrics used in the Improver Challenge, and then present comparison of methods and aggregated results. In order to have a more accurate comparison, instead of the suggested train/test split of the Improver Challenge datasets, we combine all the data and perform cross-validation. Since no single metric can capture all the subtleties of a prediction task, three non-redundant metrics were chosen by the Improver Challenge scoring panel. The three performance criteria are Belief Confusion Metric (BCM) or otherwise known as Balanced Accuracy (BAL), Correct Class Enrichment

Metric (CCEM) and Area Under the Precision Recall Curve (AUPRC).

Consider $\mathbf{S}$ to be the classifier subject belief matrix and s_{ij} the classifier belief that subject i is in class j. If $\nu_{kj} = \sum_i s_{ij}$ where i is over all subjects in class k, then the normalized BCM is the normalized difference between the ideal confusion matrix (Target) and subject belief matrix, which for two classes is defined as,

$$\text{BCM} = 1 - \sum_{i=1}^{2} (1 - \nu_{ii}/N_i)/2, \tag{5.7}$$

where N_i is the number of subjects in class i. Correct Class Enrichment Metric (CCEM) estimates the enrichment of correctly assigned classes and is defined as

$$\text{CCEM} = \sum_{i \in CC} s_{ia(i)} - \sum_{i \in IC} s_{ia(i)}, \tag{5.8}$$

where CC and IC are correctly and incorrectly classified samples, and $a(i)$ is the class assigned to subject i according to the maximum belief for that subject. CCEM is bounded by $-N \leq \text{CCEM} \leq N$ and thus normalized using $\text{CCEM}_{\text{nor}} = (\text{CCEM}/N + 1)/2$ where N is the number of subjects. AUPRC and its close relative, Area Under the Receiver Operating Curve (AUC), estimate the probability that a positive sample is ranked above a negative sample. The main difference between AUPRC and the other two measures is that for a correct ranking, AUPRC does not care how far apart the pair of positive and negative samples are, whereas in BCM and CCEM, this distance is considered in the metric.

RFS is compared with five competing feature selection methods on 13 datasets as described in Table 5.1. The number of samples and features reported reflect the processed data, and not the original datasets. In addition to the three aforementioned metrics, we compute the AUC, Matthews Correlation Coefficient (MCC) and Accuracy using ten fold cross-validation.

For a close visual inspection, we include the pairwise comparison of RFS to HHSVM, which is the base classifier, and to moderated-t/fold-change, which is the winner of the

Table 5.1: Detailed description of datasets.

item	ref.	type	pos.	neg.	total	features
1	GSE15471	Pancreatic Cancer	35	35	70	19944
2	GSE18842	Non-Small Cell Lung Cancer	44	44	88	19944
3	GSE19188	Non-Small Cell Lung Cancer	62	91	153	19944
4	GSE8671	Colorectal Cancer	32	32	64	19944
5	GSE9348	Colorectal Cancer	12	70	82	19944
6	GSE9476	Acute Myeloid Leukemia	37	26	63	12495
7	[67]	Head and Neck Cancer	100	100	200	1148
8	[110]	Breast Cancer	62	53	115	24481
9	[108]	Breast Cancer	194	100	294	24453
10	[116]	Breast Cancer	193	93	286	12495
11	[12]	Lung Cancer	10	86	96	5299
12	[75]	Chronic Obstructive Pulmonary Disease	82	46	128	12655
13	[75]	Psoriasis	120	118	238	13261

Improver Challenge, on four performance measures in Figures 5.1(a) and 5.1(b) respectively. In Figure 5.1, each color and symbol combination represents a dataset, and each point is the average performance scatter of a k-feature model for two methods based on 10 fold cross validation. Every point above the diagonal line shows better performance on the metric in favor of the RFS feature selection. We can see from this figure that most points lie above the line in favor of RFS compared to both methods on all measures. SVM RFE exhibits high variance in performance, which is expected based on previous comparisons [98]. To summarize the prediction performances for all the datasets and methods, we follow [98] and present the empirical error rate with one difference. Instead of fixing the number of features to 10, we choose the best k-feature model ($k \leq 20$) for which the average of AUPRC, BCM and CCEM is maximized.

In Table 5.2, we report the empirical error of each model, and the feature size of the model that maximizes the average score. When evaluating the performance of a feature selection method, it is important that the selected feature list performs consistently well overall, rather than be the best in some cases and perform horribly on others. As such, the ℓ_2 distance is used, in order to penalize methods with higher variance, i.e. performing well on few datasets

and poorly on others. The final row of this table is the ℓ_2 distance of each method from the overall minimum achieved error, among all methods per dataset, denoted in the last column. This table shows that RFS has a superior overall performance with the ℓ_2 distance of 7.5, while the error rate of NSC, which is the next best method, is 12.0, based on the ℓ_2 distance of empirical error rates.

Table 5.2: Empirical error rate computed using 10 fold cross-validation for each feature selection method (lower values indicate better performance and values in boldface represent the minimum for each data set/row). "mtfc" is moderated-t/fold-change, PC is Pearson Correlation, NSC is Nearest Shrunken Centroid, SVM RFE is SVM Recursive Feature Elimination, HHSVM is Hybrid Huberized SVM and RFS is the proposed Robust Feature Selection method. The last two rows are the ℓ_1 and ℓ_2 distances of the average cross-validation error of each method from the overall minimum achieved error rate of all methods per dataset, which is shown in the last column. The proposed method (RFS) achieves the lowest overall error across all datasets.

Dataset	mtfc	PC	NSC	SVM RFE	HHSVM	RFS	Min
1	12.86	10.00	10.00	**8.57**	15.71	**8.57**	8.57
2	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.00
3	**2.58**	**2.58**	3.21	3.25	3.92	3.25	2.58
4	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.00
5	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.00
6	7.86	3.33	1.67	**0.00**	1.43	3.10	0.00
7	41.00	31.00	28.00	30.50	27.50	**22.50**	22.50
8	23.41	22.65	**14.85**	26.97	17.35	17.58	14.85
9	36.46	35.77	34.39	37.10	**32.03**	34.78	32.03
10	45.18	**40.27**	43.78	47.25	43.78	40.59	40.27
11	5.00	**0.00**	**0.00**	1.00	1.00	**0.00**	0.00
12	26.03	22.76	25.77	22.63	25.71	**18.85**	18.85
13	2.52	**0.42**	0.43	**0.42**	**0.42**	**0.42**	0.42
ℓ_1	5.19	2.57	2.05	3.25	2.57	**1.09**	-
ℓ_2	26.06	14.82	12.01	18.98	13.16	**7.50**	-

We also analyze the stability of each feature selection approach by computing the selection frequency of each feature among all the folds. Since 10-fold cross-validation is used, there are 10 sets of ranked lists for any given model size k ($k \leq 20$). In Table 5.3, we report the optimal chosen model size m (Size) based on the combined average of the three performance measures, the number of features in common in at least five folds s (Common), and the stability of these sets of features s/m (Stability). The last row, which is the average of each

statistic over all datasets, is a summary for each method. Table 5.3 shows that RFS is much more stable than the base HHSVM classifier, and close to NSC (the most stable method), while SVM RFE is the least stable among the considered methods. Note that if a method chose the first 10 features regardless of the data, it will be extremely stable but terrible at predictive accuracy. As such, although PC and NSC choose a more stable feature set, it does not necessarily suggest better performance in terms of predictive accuracy. In essence, RFS provides a principled approach to improve the stability and predictive accuracy at the same time.

5.7 Conclusion

In this work, we present Robust Feature Selection (RFS), as a method to learn a stable and accurate diagnostic signature from high-dimensional low-sample datasets. We compare RFS with a number of popular and state of the art feature selection methods, which have been previously successful, as reported in the literature [98, 52, 102], and an international competition [99]. We show through extensive experiments that RFS significantly outperforms all considered methods, in reducing the empirical error, while choosing sparse models at the same time. The goal of personalized medicine is to discover refined groupings of heterogeneous diseases such as cancer. Leveraging transcriptomics data has tremendously influenced and accelerated this process. However, in order to make personalized medicine possible, a diagnostic/prognostic signature differentiating various groups, should not only be accurate, but also stable enough to be reproducible in different settings [34]. Development of a stable and accurate diagnostic signature is the first step towards this goal. The Robust Feature Selection method tries to fill this gap by providing an improved diagnostic signature.

Table 5.3: Stability of each feature selection method as shown by the average number of features in common in at least five folds of the 10-fold cross-validation (higher stability values show improved stability). "Size" is the optimal chosen model size, m, based on the average of AUPRC, BCM and CCEM. "Common" is the number of features common in at least five folds, s. The stability column corresponds to "Common" divided by "Size", s/m (higher is better). The last row is the average of each statistic over all datasets. This table shows that the proposed RFS method is much more stable than the base HHSVM classifier, mtfc, SVM RFE, and its stability is close to NSC's and PC's, which are the most stable method. Although PC and NSC are slightly more stable, the chosen feature sets are not as predictive as RFS.

	mtfc			PC			NSC		
Dataset	Size	Common	Stability	Size	Common	Stability	Size	Common	Stability
1	6	5	0.83	3	3	1.00	2	2	1.00
2	11	11	1.00	12	11	0.92	13	11	0.85
3	19	20	1.05	14	13	0.93	14	13	0.93
4	10	9	0.90	2	2	1.00	3	3	1.00
5	18	17	0.94	2	2	1.00	2	2	1.00
6	20	14	0.70	17	16	0.94	4	3	0.75
7	15	1	0.07	6	6	1.00	6	5	0.83
8	16	12	0.75	18	16	0.89	20	18	0.90
9	19	16	0.84	13	9	0.69	15	16	1.07
10	2	1	0.50	18	15	0.83	17	15	0.88
11	11	9	0.82	1	1	1.00	3	3	1.00
12	11	8	0.73	8	8	1.00	20	21	1.05
13	16	16	1.00	6	6	1.00	20	20	1.00
Mean	13.38	10.69	**0.78**	9.23	8.31	**0.94**	10.69	10.15	**0.94**

	SVM RFE			HHSVM			RFS		
Dataset	Size	Common	Stability	Size	Common	Stability	Size	Common	Stability
1	14	4	0.29	5	1	0.20	9	7	0.78
2	17	13	0.76	9	6	0.67	11	9	0.82
3	18	5	0.28	20	12	0.60	19	18	0.95
4	8	4	0.50	3	2	0.67	3	3	1.00
5	8	4	0.50	5	4	0.80	3	2	0.67
6	14	13	0.93	17	11	0.65	20	20	1.00
7	18	8	0.44	16	10	0.63	11	10	0.91
8	20	5	0.25	9	7	0.78	13	11	0.85
9	19	7	0.37	17	9	0.53	17	13	0.76
10	8	0	0.00	5	3	0.60	20	18	0.90
11	5	4	0.80	7	5	0.71	5	5	1.00
12	11	3	0.27	12	8	0.67	9	8	0.89
13	20	18	0.90	17	14	0.82	6	6	1.00
Mean	13.85	6.77	**0.48**	10.92	7.08	**0.64**	11.23	10.00	**0.89**

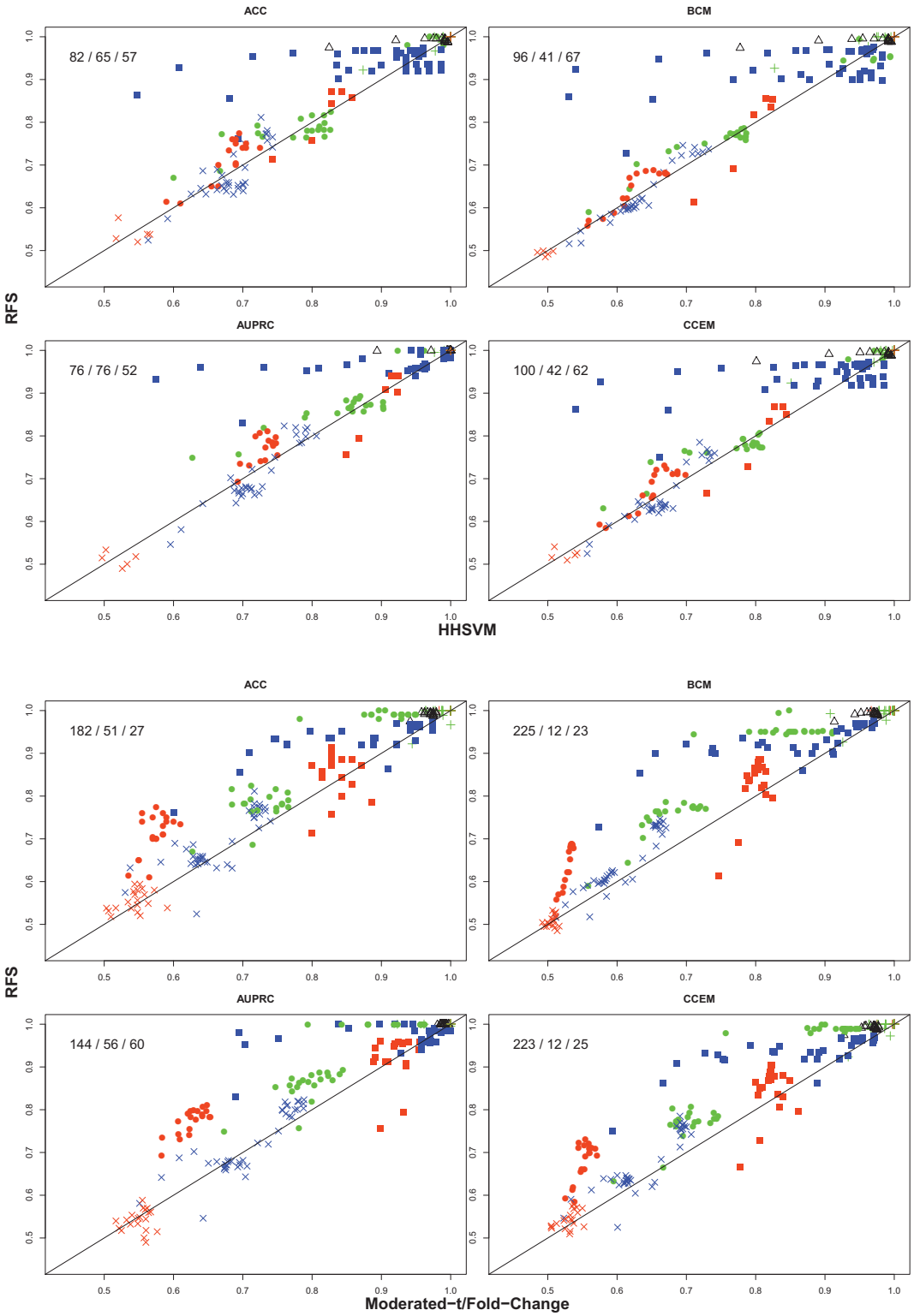


Figure 5.1: Pairwise performance scatter plot of RFS compared to HHSVM (top) and moderated-t/fold-change (bottom). Each color/symbol represents a dataset and each point is the average performance of a k-feature model ($k \leq 20$), based on 10 fold cross validation. ACC is accuracy, AUPRC is area under the precision recall curve, BCM is belief confusion metric and CCEM is correct class enrichment metric. Every point above the diagonal line shows better performance in favor of RFS with respect to the measure. The 3 numbers, $a/b/c$, on the top left corner of each plot, show the number of points "above/on/below" the diagonal line.

CHAPTER 6 CONCLUSIONS AND FUTURE WORK

Machine learning approaches can be divided into standard tasks such as supervised, unsupervised and semi-supervised learning. Due to its high impact and utility, the same concepts have been employed in a wide variety of applications. However, applying standard machine learning tasks in different domains and datasets comes with certain difficulties and requires great effort in order to succeed. This work aims to provide solutions for some of the recurring problems of machine learning in biology and medicine.

One of the major problems in classification of patient population is attributing costs to different classes and imbalanced prior class distribution. In Chapter 2, we explore various ways of developing cost sensitive classification. We emphasize the two distinct regimes in classification methods, i.e., the ones that estimate posterior probability and the ones that learn a discriminant function directly. This fundamental difference is the most important aspect to consider when developing cost sensitive classifiers.

Next, we apply the best practices found in cost sensitive learning to ontology mapping. Ontologies have been extensively used in biology and medicine. The Open Biological and Biomedical Ontologies (OBO) [96] for example is a collaborative effort to developing and maintaining useful ontologies. Since the number of mismatches grows in $\mathbf{O}(n^2)$ when the number of matches or concepts in a pair of ontologies is $\mathbf{O}(n)$, the training data is highly skewed. Furthermore, due to the human intervention requirement to fix the errors, the cost of false negatives is much lower than false positives. In Chapter 3, to make data integration easier, we develop an automatic ontology mapping framework that learns the mapper while taking these factors into account.

The ultimate goal of personalized medicine is the ability to identify specific disease mechanisms that may be at play in each patient, and provide a targeted individualized treatment. A first step towards this was the introduction of treatments targeted at disease subtypes such

as HER2+ breast tumors. Even though breast cancer subtypes are well understood, 20% of the HER2 targeted treatments are ineffective, and many of the remaining 80% will eventually become resistant. These failures may be avoided by refining our current understanding of disease subtypes. Therefore, identifying homogeneous disease subtypes and patient subgroups is an intermediate step towards the ultimate goal of personalized medicine.

Chapter 4, deals with classifying patient subtypes from heterogeneous diseases. Support Vector Machines are one of the most successful classifiers in use today. However, they are unable to predict whether the samples are from completely new classes/distributions or too close to the decision boundary to be considered reliable. We propose a solution based on SVM to reject samples that do not belong to any class from the training data or are too unreliable to be classified. We show through artificial data that our method captures most of underlying uncertainty in classification. Furthermore, we validate the method by successfully applying it to Diabetes, Cardiotocography, MNIST and Leukemia datasets. For the Diabetes, Cardiotocography and MNIST datasets the proposed method effectively reduces false positives by discarding unreliable predictions, and for the Leukemia dataset, the method rejects samples from the new class which were not present during training.

In Chapter 5, the last part of this thesis, we develop novel methods that handle data typically found in biological experiments. In such experiments, data generated from high-throughput technology are very high-dimensional, usually in the order of tens of thousands. In addition, the number of samples that we can obtain are relatively low and in the order of tens to hundreds. In this setting, standard and classical methods of hypothesis testing fails due to very low power. Furthermore, many machine learning methods such as classification and regression would be unreliable, and thus knowledge discovery from such experiments will be very difficult.

We develop the Robust Feature Selection (RFS) as a novel method for classification in high-dimensional low-sample datasets. RFS is a sparse classification method that utilizes a randomization procedure for stability. RFS can learn an accurate and stable diagnostic

signature from high-throughput experiments. We compare RFS with a number of popular and state of the art feature selection methods and show through extensive experiments significant improvement in both accuracy and stability of the selected features in comparison to all considered methods.

Although the machine learning methods developed here alleviate some of the recurring problems in bioinformatics, the challenges of the discipline are far from over. The techniques described here finds subgroup of samples that are similar or subsets of genes that differentiate the samples. As our understanding of the biology in life sciences further advances and this knowledge is utilized for personalized medicine, so would the intricacy and abundance of information to be considered when moving forward. For example, there are more information available about the interaction among the genes and pathways of interest that regulate biological processes. Sparse structured methods could be employed to combine this information with high-throughput experiments to refine the groupings and enhance the quality of our predictions. Furthermore, sparse prediction methods such as the graphical lasso can be used to find previously unknown interaction networks.

BIBLIOGRAPHY

[1] Inria ontology. `http://fr.inrialpes.exmo.rdf.bib.owl`.

[2] Karlsruhe ontology. `http://www.aifb.unikarlsruhe.de/ontology`.

[3] OWL web ontology language. `http://www.w3.org/TR/owl-features/`, 2004.

[4] N. Abe, B. Zadrozny, and J. Langford. An iterative method for multi-class cost-sensitive learning. In *Proceedings of the 10th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'04)*, 3–11, New York, NY, USA, 2004. ACM.

[5] Affymetrix. Statistical algorithms description document. Technical report, Affymetrix, Santa Clara, CA, 2002.

[6] S. A. Armstrong, J. E. Staunton, L. B. Silverman, R. Pieters, M. L. den Boer, M. D. Minden, S. E. Sallan, E. S. Lander, T. R. Golub, and S. J. Korsmeyer. MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia. *Nature Genetics*, 30(1):41–47, January 2002.

[7] A. Asuncion and D. Newman. UCI machine learning repository, 2007.

[8] F. R. Bach. Bolasso: model consistent lasso estimation through the bootstrap. In *Proceedings of the 25th International Conference on Machine learning*, 33–40. ACM, 2008.

[9] P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

[10] P. L. Bartlett and M. H. Wegkamp. Classification with a reject option using a hinge loss. *Journal of Machine Learning Research*, 9:1823–1840, June 2008.

[11] J. Bedo, C. S, and A. Kowalczyk. An efficient alternative to svm based recursive feature elimination with applications in natural language processing and bioinformatics. In *In Artificial Intelligence*, 170–180, 2006.

[12] D. G. Beer, S. L. Kardia, C.-C. Huang, T. J. Giordano, A. M. Levin, D. E. Misek, L. Lin, G. Chen, T. G. Gharib, D. G. Thomas, M. L. Lizyness, R. Kuick, S. Hayasaka, J. M. Taylor, M. D. Iannettoni, M. B. Orringer, and S. Hanash. Gene-expression profiles predict survival of patients with lung adenocarcinoma. *Nature Medicine*, 8(8):816–824, Jul 2002.

[13] B. M. Bolstad, R. A. Irizarry, M. Åstrand, and T. P. Speed. A Comparison of Normalization Methods for High Density Oligonucleotide Array Data Based on variance and bias. *Bioinformatics*, 19(2):185–193, January 2003.

[14] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the 5th Annual Workshop on Computational Learning Theory*, 144–152. ACM, 1992.

[15] J. P. Bradford, C. Kunz, R. Kohavi, C. Brunk, and C. E. Brodley. Pruning decision trees with misclassification costs. In *Proceedings of the 10th European Conference on Machine Learning (ECML'98)*, 131–136, London, UK, 1998. Springer-Verlag.

[16] A. P. Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30:1145–1159, 1997.

[17] L. Breiman. Bias, variance, and arcing classifiers. Technical Report 460, Statistics Department, University of California, Berkeley, CA, USA, 1996.

[18] G. Brown, A. Pocock, M.-J. Zhao, and M. Luján. Conditional likelihood maximisation: a unifying framework for information theoretic feature selection. *The Journal of Machine Learning Research*, 13:27–66, 2012.

[19] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[20] N. V. Chawla, K. W. Bowyer, and W. P. Kegelmeyer. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

[21] C. Chow. On optimum recognition error and reject tradeoff. *IEEE Transactions on*

Information Theory, 16(1):41–46, 1970.

[22] W. W. Cohen, P. D. Ravikumar, and S. E. Fienberg. A comparison of string distance metrics for name-matching tasks. In *IIWeb*, 73–78, 2003.

[23] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[24] M. H. DeGroot and S. E. Fienberg. The comparison and evaluation of forecasters. *The statistician*, 12:12–22, 1983.

[25] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, December 2006.

[26] A. Doan, J. Madhavan, R. Dhamankar, P. Domingos, and A. Halevy. Learning to match ontologies on the semantic web. *The International Journal on Very Large Data Bases (VLDB)*, 12(4):303–319, 2003.

[27] P. Domingos. Metacost: A general method for making classifiers cost-sensitive. In *In Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining*, 155–164. ACM Press, 1999.

[28] P. Domingos. A unified bias-variance decomposition and its applications. In *In Proceedings 17th International Conference on Machine Learning*, 231–238. Morgan Kaufmann, 2000.

[29] P. Domingos and M. Pazzani. Beyond independence: Conditions for the optimality of the simple bayesian classifier. In *In Proceedings of the 13th International Conference on Machine Learning*, 105–112, 1996.

[30] C. Drummond and R. C. Holte. C4.5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling. In *In Proceedings of the International Conference on Machine Learning (ICML 2003) Workshop on Learning from Imbalanced Data Sets II*, 1–8, Washington, DC, USA, 2003.

[31] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification*. John Wiley & Sons, 2012.

[32] R. Edgar, M. Domrachev, and A. E. Lash. Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Research*, 30(1):207–210, 2002.

[33] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *The Annals of Statistics*, 32(2):407–499, 2004.

[34] L. Ein-Dor, O. Zuk, and E. Domany. Thousands of samples are needed to generate a robust gene list for predicting outcome in cancer. *In Proceedings of the National Academy of Sciences*, 103(15):5923–5928, 2006.

[35] C. Elkan. The foundations of Cost-Sensitive learning. In B. Nebel, editor, *In Proceedings of the 17th International Conference on Artificial Intelligence (IJCAI'01)*, 973–978, San Francisco, CA, August 2001. Morgan Kaufmann Publishers, Inc.

[36] J. D. Emerson and D. C. Hoaglin. Analysis of two-way tables by medians. *Understanding Robust and Exploratory Data Analysis*, 165–210, 1983.

[37] F. Esposito, D. Malerba, and G. Semeraro. A comparative analysis of methods for pruning decision trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):476–491, 1997.

[38] J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001.

[39] U. M. Fayyad and K. B. Irani. On the handling of continuous-valued attributes in decision tree generation. *Machine Learning*, 8:87–102, 1992.

[40] R. A. Fisher. On the interpretation of χ^2 from contingency tables, and the calculation of p. *Journal of the Royal Statistical Society*, 85(1):87–94, 1922.

[41] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *In Proceedings of the International Conference on Machine Learning*, 148–156, 1996.

[42] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2):337–407, 2000.

[43] M. Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32(200):pp. 675–701, 1937.

[44] M. Friedman. A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, 11(1):pp. 86–92, 1940.

[45] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2):131–163, 1997.

[46] K. Fukunaga. *Introduction to statistical pattern recognition*. Access Online via Elsevier, 1990.

[47] G. Fumera and F. Roli. Cost-sensitive learning in support vector machines. In *In VIII Convegno Associazione Italiana per LŠIntelligenza Artificiale*, 2002.

[48] G. Fumera and F. Roli. Support vector machines with embedded reject option. In *Pattern Recognition with Support Vector Machines*, 68–82. Springer, 2002.

[49] P. Geibel and F. Wysotzki. Perceptron based learning with example dependent and noisy costs. In T. Fawcett and N. Mishra, editors, *In Proceedings of the International Conference on Machine Learning*, 218–225. AAAI Press, 2003.

[50] Y. Grandvalet, A. Rakotomamonjy, J. Keshet, and S. Canu. Support vector machines with a reject option. In *Neural Information Processing Systems (NIPS)*, 537–544, Vancouver, Canada, December 2008.

[51] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

[52] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1-3):389–422, 2002.

[53] H. Han, W. Wang, and B.-H. Mao. Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In D.-S. Huang, X.-P. Zhang, and G.-B. Huang, editors, *ICIC (1)*, volume 3644 of *Lecture Notes in Computer Science*, 878–887. Springer, 2005.

[54] J. A. Hanley and B. J. McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, April 1982.

[55] R. Herbei and M. H. Wegkamp. Classification with reject option. *Canadian Journal of Statistics*, 34(4):709–721, 2006.

[56] J. Huang and C. X. Ling. Using AUC and accuracy in evaluating learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 17(3):299–310, 2005.

[57] R. L. Iman and J. M. Davenport. Approximations of the critical region of the friedman statistic. *Communications in Statistics-Theory and Methods*, 9(6):571–595, 1980.

[58] R. A. Irizarry, B. Hobbs, F. Collin, Y. D. Beazer-Barclay, K. J. Antonellis, U. Scherf, and T. P. Speed. Exploration, Normalization, and Summaries of High Density Oligonucleotide Array Probe Level Data. *Biostatistics*, 4(2):249–264, 2003.

[59] R. A. Irizarry, Z. Wu, and H. A. Jaffee. Comparison of affymetrix genechip expression measures. *Bioinformatics*, 22(7):789–794, 2006.

[60] M. A. Jaro. Probabilistic linkage of large public health data files. *Statistics in medicine*, 14(5-7):491–498, 1995.

[61] L. Jiang, D. Wang, Z. Cai, and X. Yan. Survey of improving naive bayes for classification. In R. Alhajj, H. Gao, X. Li, J. Li, and O. R. Zaïane, editors, *Advanced Data Mining and Applications*, volume 4632 of *Lecture Notes in Computer Science*, 134–145. Springer, 2007.

[62] A. Kalyanpur, B. Parsia, E. Sirin, B. Cuenca-Grau, and J. Hendler. Swoop: A web ontology editing browser. *Journal of Web Semantics*, 4(2), 2005.

[63] E. J. Keogh and M. J. Pazzani. Learning augmented bayesian classifiers: A comparison of distribution-based and classification-based approaches. In *In Proceedings of the 7th International Workshop on Artificial Intelligence and Statistics*, 225–230. Citeseer, 1999.

[64] U. Knoll, G. Nakhaeizadeh, and B. Tausend. Cost-sensitive pruning of decision trees. In F. Bergadano and L. de Raedt, editors, *In Proceedings of the European Conference*

on Machine Learning, volume 784 of *LNAI*, 383–386, Berlin, Apr. 1994. Springer.

[65] Y. LeCun and C. Cortes. Mnist character recognition data set.

[66] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. In *Soviet physics doklady*, volume 10, 707, 1966.

[67] H.-S. Lin, H. S. Talwar, A. L. Tarca, A. Ionan, M. Chatterjee, B. Ye, J. Wojciechowski, S. Mohapatra, M. D. Basson, G. H. Yoo, B. Peshek, F. Lonardo, C.-J. G. Pan, A. J. Folbe, S. Draghici, J. Abrams, and M. A. Tainsky. Autoantibody approach for serum-based detection of head and neck cancer. *Cancer Epidemiol Biomarkers Prevention*, 16(11):2396–405, 2007.

[68] H.-T. Lin, C.-J. Lin, and R. C. Weng. A note on Platt's probabilistic outputs for support vector machines. *Machine Learning*, 68(3):267–276, 2007.

[69] C. X. Ling, Q. Yang, J. Wang, and S. Zhang. Decision trees with minimal costs. In C. E. Brodley, editor, *In Proceedings of the International Conference on Machine Learning*, volume 69 of *ACM International Conference Proceeding Series*. ACM, 2004.

[70] X.-Y. Liu and Z.-H. Zhou. The influence of class imbalance on cost-sensitive learning: An empirical study. In *In Proceedings of the International Conference on Data Mining*, 970–974. IEEE Computer Society, 2006.

[71] D. D. Margineantu. *Methods for Cost Sensitive Learning*. PhD thesis, Oregon State University, 2001.

[72] K. McCarthy, B. Zabar, and G. Weiss. Does cost-sensitive learning beat sampling for classifying rare classes? In *In Proceedings of the 1st International workshop on Utility-based Data Mining (UBDM'05)*, 69–77, New York, NY, USA, 2005. ACM.

[73] N. Meinshausen and P. Bühlmann. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473, 2010.

[74] P. Meyer, L. G. Alexopoulos, T. Bonk, A. Califano, C. R. Cho, A. de la Fuente, D. de Graaf, A. J. Hartemink, J. Hoeng, and N. V. Ivanov. Verification of systems biology research in the age of collaborative competition. *Nature Biotechnology*, 29(9):811,

2011.

[75] P. Meyer, J. Hoeng, R. Norel, J. Sprengel, K. Stolle, T. Bonk, S. Corthesy, A. Royyuru, M. C. Peitsch, J. J. Rice, and G. Stolovitzky. IMPROVER challenge: Diagnostics signature. `http://www.sbvimprover.com`, 2012.

[76] P. Meyer, J. Hoeng, R. Norel, J. Sprengel, K. Stolle, T. Bonk, S. Corthesy, A. Royyuru, M. C. Peitsch, J. J. Rice, and G. Stolovitzky. Industrial methodology for process verification in research (IMPROVER): Towards systems biology verification. *Bioinformatics*, 28(9):1193–1201, 2012.

[77] A. E. Monge and C. Elkan. The field matching problem: Algorithms and applications. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 267–270, 1996.

[78] S. Mukherjee, P. Tamayo, D. Slonim, A. Verri, T. Golub, J. Mesirov, and T. Poggio. Support vector machine classification of microarray data. *CBCL Paper 182/AI Memo 1677*, 182, 1999.

[79] F. Naef, D. A. Lim, N. Patil, and M. O. Magnasco. From features to expression: High-density oligonucleotide array analysis revisited. *arXiv preprint physics/0102010*, 2001.

[80] P. B. Nemenyi. *Distribution-free multiple comparisons*. PhD thesis, Princeton University, 1963.

[81] A. Y. Ng and M. I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *NIPS*, 841–848, 2001.

[82] A. Niculescu-Mizil and R. Caruana. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd International Conference on Machine learning*, 625–632, Bonn, Germany, August 2005. ACM.

[83] J. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[84] F. Provost and P. Domingos. Tree induction for probability-based ranking. *Machine*

Learning, 52(3):199–215, 2003.

[85] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan-Kaufmann, 1993.

[86] T. Robertson, F. Wright, R. L. Dykstra, and T. Robertson. *Order restricted statistical inference*. Wiley, London, 1988.

[87] S. Rosset and J. Zhu. Piecewise linear regularized solution paths. *The Annals of Statistics*, 35(3):1012–1030, 2007.

[88] C. Rudin. The p-norm push: A simple convex ranking algorithm that concentrates at the top of the list. *Journal of Machine Learning Research*, 9:2233–2271, 2008.

[89] Y. Saeys, I. Inza, and P. Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, 2007.

[90] R. D. Shah and R. J. Samworth. Variable selection with error control: another look at stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 75(1):55–80, 2013.

[91] S. H. Shahri and H. M. Jamil. An extendable meta-learning algorithm for ontology mapping. In *Flexible Query Answering Systems, 8th International Conference, FQAS 2009, Roskilde, Denmark, October 26-28, 2009. Proceedings*, 418–430, 2009.

[92] V. S. Sheng and C. X. Ling. Thresholding for making classifiers cost-sensitive. In *Proceedings of the National Conference on Artificial Intelligence*, 2006.

[93] D. J. Sheskin. *Handbook of Parametric and Nonparametric Statistical Procedures*. Chapman & Hall/CRC, 4 edition, 2007.

[94] P. Shvaiko and J. Euzenat. A survey of schema-based matching approaches. In *Journal on Data Semantics IV*, 146–171. Springer, 2005.

[95] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical owl-dl reasoner. *Journal of Web Semantics*, 2007.

[96] B. Smith, M. Ashburner, C. Rosse, J. Bard, W. Bug, W. Ceusters, L. J. Goldberg, K. Eilbeck, A. Ireland, C. J. Mungall, et al. The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration. *Nature biotechnology*,

25(11):1251–1255, 2007.

[97] G. K. Smyth. Linear models and empirical Bayes methods for assessing differential expression in microarray experiments. *Statistical applications in genetics and molecular biology*, 3(1):Article3, 2004.

[98] L. Song, A. Smola, A. Gretton, J. Bedo, and K. Borgwardt. Feature selection via dependence maximization. *The Journal of Machine Learning Research*, 13(1):1393–1434, May 2012.

[99] A. L. Tarca, M. Lauria, M. Unger, E. Bilal, S. Boue, K. K. Dey, J. Hoeng, H. Koeppl, F. Martin, P. Meyer, et al. Strengths and limitations of microarray-based phenotype prediction: Lessons learned from the improver diagnostic signature challenge. *Bioinformatics*, 29(22):2892–2899, 2013.

[100] U. G. Thomas. Philip Morris International, IBM Announce IMPROVER Systems Biology Verification Challenge Winners. *GenomeWeb BioInform*, October 2012.

[101] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 267–288, 1996.

[102] R. Tibshirani, T. Hastie, B. Narasimhan, and G. Chu. Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proceedings of the National Academy of Sciences*, 99(10):6567–6572, 2002.

[103] K. M. Ting. An instance-weighting method to induce cost-sensitive trees. *IEEE Transactions on Knowledge and Data Engineering*, 14(3):659–665, 2002.

[104] F. Tortorella. Reducing the classification cost of support vector classifiers through an ROC-based reject rule. *Pattern Analysis and Applications*, 7(2):128–143, 2004.

[105] P. D. Turney. Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *Journal of Artificial Intelligence Research*, 2:369–409, 1995.

[106] P. D. Turney. Types of cost in inductive concept learning. In *Workshop on Cost-Sensitive Learning at the 17th International Conference on Machine Learning*, volume

cs.LG/0212034, 15–21, 2000.

[107] O. Udrea, L. Getoor, and R. J. Miller. Leveraging data and structure in ontology integration. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 449–460. ACM, 2007.

[108] M. J. Van De Vijver, Y. D. He, L. J. van't Veer, H. Dai, A. A. Hart, D. W. Voskuil, G. J. Schreiber, J. L. Peterse, C. Roberts, M. J. Marton, et al. A gene-expression signature as a predictor of survival in breast cancer. *New England Journal of Medicine*, 347(25):1999–2009, 2002.

[109] C. J. van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 2nd edition, 1979.

[110] L. J. van 't Veer, H. Dai, M. J. van de Vijver, Y. D. He, A. Hart, M. Mao, H. L. Peterse, K. van der Kooy, M. J. Marton, A. T. Witteveenothers, G. J. Schreiber, R. M. Kerkhoven, C. Roberts, P. S. Linsley, R. Bernards, and S. H. Friend. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415(6871):530–536, January 2002.

[111] V. N. Vapnik. Principles of risk minimization for learning theory. In *Advances in Neural Information Processing (NIPS'4)*, 831–838, San Mateo, CA, 1992. Morgan Kaufmann.

[112] C. Voichita, P. Khatri, and S. Draghici. Identifying uncertainty regions in Support Vector Machines using geometric margin and convex hulls. In *Proceedings of the International Joint Conference on Neural Networks, JCNN 2008. IEEE World Congress on Computational Intelligence*, Hong Kong, 1-8 June 2008. Calin's paper.

[113] M. J. Wainwright. Sharp thresholds for noisy and high-dimensional recovery of sparsity using l1-constrained quadratic programming (lasso). *IEEE Transactions on Information Theory*, 55(5):2183–2202, 2009.

[114] L. Wang, J. Zhu, and H. Zou. Hybrid huberized support vector machines for microarray classification and gene selection. *Bioinformatics*, 24(3):412–419, 2008.

[115] W. Wang, J. Lu, R. Lee, Z. Gu, and R. Clarke. Iterative Normalization of cDNA Mi-

croarray Data. *IEEE Transactions on Information Technology in Biomedicine*, 6(1):29–37, March 2002.

[116] W. Q. Wang, Y. H. Zhou, and R. Bi. Correlating genes and functions to human disease by systematic differential analysis of expression profiles. In *Advances in Intelligent Computing, Pt 2, Proceedings*, volume 3645 of *Lecture Notes in Computer Science*, 11–20. 2005.

[117] G. I. Webb, J. R. Boughton, and Z. Wang. Not so naive bayes: Aggregating one-dependence estimators. *Machine Learning*, 58(1):5–24, 2005.

[118] M. Wegkamp and M. Yuan. Support vector machines with a reject option. *Bernoulli*, 17(4):1368–1385, 2011.

[119] M. Yuan and M. Wegkamp. Classification methods with reject option based on convex risk minimization. *The Journal of Machine Learning Research*, 11:111–130, 2010.

[120] B. Zadrozny. Learning and evaluating classifiers under sample selection bias. In *Proceedings of the 21st International Conference on Machine learning (ICML'04)*, 114, New York, NY, USA, 2004. ACM.

[121] B. Zadrozny and C. Elkan. Learning and making decisions when costs and probabilities are both unknown. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'01)*, 204–213, New York, NY, USA, 2001. ACM.

[122] B. Zadrozny and C. Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02)*, 694–699, New York, NY, USA, 2002. ACM.

[123] B. Zadrozny, J. Langford, and N. Abe. Cost-sensitive learning by cost-proportionate example weighting. In *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM'03)*, 435, Washington, DC, USA, 2003. IEEE Computer Society.

[124] J. H. Zar. *Biostatistical Analysis (5th Edition)*. Prentice-Hall, Inc., Upper Saddle

River, NJ, USA, 2007.

[125] F. Zheng and G. I. Webb. Efficient lazy elimination for averaged one-dependence estimators. In *Proceedings of the 23rd International Conference on Machine Learning (ICML'06)*, 1113–1120, New York, NY, USA, 2006. ACM.

[126] Z. Zheng and G. I. Webb. Lazy learning of bayesian rules. *Machine Learning*, 41(1):53–84, 2000.

[127] Z. Zheng, G. I. Webb, and K. M. Ting. Lazy bayesian rules: A lazy semi-naive bayesian learning technique competitive to boosting decision trees. In *Proceedings of the 16th International Conference on Machine Learning*. Citeseer, 1999.

[128] H. Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101(476):1418–1429, 2006.

[129] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

ABSTRACT

REGULARIZED RISK MINIMIZATION IN BIOLOGY AND MEDICINE

by

SAIED HAIDARIAN SHAHRI

August 2015

Advisor: Dr. Sorin Draghici

Major: Computer Science (Bioinformatics)

Degree: Doctor of Philosophy

Machine learning as a field is defined to be the set of computational algorithms that improve their performance by assimilating data. As such, the field as a whole has found applications in many diverse disciplines from robotics and communication in engineering to economics and finance, and also biology and medicine. It should not come as a surprise that many popular methods in use today have completely different origins. Despite this heterogeneity, different methods can be divided into standard tasks, such as supervised, unsupervised, semi-supervised and reinforcement learning.

Although machine learning as a field can be formalized as methods trying to solve certain standard tasks, applying these tasks on datasets from different fields comes with certain caveats, and sometimes is fraught with challenges. In this thesis, we develop general procedures and novel solutions, dealing with practical problems that arise when modeling biological and medical data.

Cost sensitive learning is an important area of research in machine learning which addresses the widespread and practical problem of dealing with different costs during the learning and deployment of classification algorithms. In many applications such as credit fraud detection, network intrusion and specifically medical diagnosis domains, prior class distributions are highly skewed, which makes the training examples very much unbalanced.

Combining this with uneven misclassification costs renders standard machine learning approaches useless in learning an acceptable decision function. We experimentally show the benefits and shortcomings of various methods that convert cost blind learning algorithms to cost sensitive ones. Using the results and best practices found for cost sensitive learning, we design and develop a machine learning approach to ontology mapping.

Next, we present a novel approach to deal with uncertainty in classification when costs are unknown or otherwise hard to assign. Support Vector Machines (SVM) are considered to be among the most successful approaches for classification. However prediction of instances near the decision boundary depends more on the specific parameter selection or noise in data, rather than a clear difference in features. In many applications such as medical diagnosis, these regions should be labeled as uncertain rather than assigned to any particular class. Furthermore, instances may belong to novel disease subtypes that are not from any previously known class. In such applications, declining to make a prediction could be beneficial when more powerful but expensive tests are available. We develop a novel approach for optimal selection of the threshold and show its successful application on three biological and medical datasets.

The last part of this thesis provides novel solutions for handling high dimensional data. Although high-dimensional data is ubiquitously found in many disciplines, current life science research almost always involves high-dimensional genomics/proteomics data. The "omics" data provide a wealth of information and have changed the research landscape in biology and medicine. However, these data are plagued with noise, redundancy and collinearity, which makes the discovery process very difficult and costly. Any method that can accurately detect irrelevant and noisy variables in omics data would be highly valuable. We present Robust Feature Selection (RFS), a randomized feature selection approach dedicated to low-sample high-dimensional data. RFS combines an embedded feature selection method with a randomization procedure for stability.

Recent advances in sparse recovery and estimation methods have provided efficient and

asymptotically consistent feature selection algorithms. However, these methods lack finite sample error control due to instability. Furthermore, the chances of correct recovery diminish with more collinearity among features. To overcome these difficulties, RFS uses a randomization procedure to provide an accurate and stable feature selection method. We thoroughly evaluate RFS by comparing it to a number of popular univariate and multivariate feature selection methods and show marked prediction accuracy improvement of a diagnostic signature, while preserving a good stability.

AUTOBIOGRAPHICAL STATEMENT

SAIED HAIDARIAN SHAHRI

Education

- Ph.D. Computer Science, Wayne State University, Detroit MI, USA, July 2015.
- M.S. Artificial Intelligence & Robotics, University of Tehran, Tehran, Iran, May 2007.
- B.S. Computer Science, Ferdowsi University of Mashhad, Mashhad, Iran, May 2004.

Peer review publications

1. S. Haidarian, C. Voichita and S. Draghici. *Learning a Robust Diagnostic Signature from Low-Sample High-Dimensional Data.* (under review).
2. C. Voichita, Z. Xu, S. Haidarian, R. Romero and S. Draghici. *Identifying Patient Subgroups Using Classification Uncertainty.* (under review).
3. N. G. Than, R. Romero, A. L. Tarca, S. Haidarian and D. E. Wildman. *A primate subfamily of galectins expressed at the maternal-fetal interface that promote immune cell death.* In Proceedings of National Academy of Science, 106(24): 9731–9736 (PNAS'09). USA June 16, 2009.
4. H. Haidarian Shahri and S. Haidarian Shahri. *Eliminating Duplicates in Information Integration: An Adaptive, Extensible Framework.* IEEE Intelligent Systems, Vol. 21, No. 5, pp. 63–71, Sept/Oct, 2006.
5. J. Faiz, S. Lotfi-fard and S. Haidarian Shahri *Prony-Based Optimal Bayes Fault Classification of Over-current Protection.* IEEE Transactions on Power Delivery, vol. 22, no. 3, July 2007.
6. S. Haidarian Shahri and H. M. Jamil. *An Extendable Meta-learning Algorithm for Ontology Mapping.* Proceedings of the 8th International Conference on Flexible Query Answering Systems (FQAS'09), Roskilde, Denmark, October 26-28, 2009.
7. S. Haidarian Shahri and M. Nili Ahmadabadi. *Learning to Attend to Concepts: An Incremental Hidden Variable Networks Approach.* Proceedings of the 4th International Workshop on Attention and Performance in Computational Vision at the International Joint Conference on Artificial Intelligence (IJCAI'07), Hyderabad, India, 2007.
8. S. Haidarian Shahri and M. Nili Ahmadabadi. *Incremental Learning of Cognitive Concepts: A Hidden Variable Networks Approach.* Proceedings of the International Symposium on Practical Cognitive Agents and Robots (PCAR'06), Perth, Australia, 2006.
9. S. Haidarian Shahri and A. R. Masoumzadeh. *Prioritizing Pass Options by Reinforcement of Simulated Soccer Agents.* Proceedings of the 25th IASTED International Conference on Artificial Intelligence and Applications (AIA'07), Innsbruck, Austria, 2007.
10. K. Razi, S. Haidarian Shahri and A. Rahimi Kian *Finding Nash Equilibrium Point of Nonlinear Non-cooperative Games using Coevolutionary Algorithms.* Proceedings of the 7th International Conference on Intelligent Systems Design and Applications, Brazil, 2007.
11. S. Haidarian Shahri, F. Rastegar and M. Nili Ahmadabadi *Bayesian Approach to Learning Temporally Extended Concepts.* Proceedings of the 12th International CSI Computer Conference (CSICC'07), Tehran, Iran, 2006.