uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

u Ottawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

William Klement

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

Ph.D. (Computer Science)

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Evaluating Machine Learning Methods: Scored Receiver Operating Characteristics (sROC) Curves

TITRE DE LA THÈSE / TITLE OF THESIS

Stan Matwin

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

Nathalie Japkowiczz

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

Christopher Drummond

John Oommen

Caral Brodley
Tufts University, Medford MA.

Herna Viktor

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

Evaluating Machine Learning Methods: scored Receiver Operating Characteristics (sROC) Curves

William Klement

A Thesis
Submitted to the School of Graduate Studies and Research
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Computer Science

The Ottawa-Carleton Institute for Computer Science
University of Ottawa
Ottawa, Ontario, Canada

November 21, 2010

Your file Votre référence
ISBN: 978-0-494-74234-1
Our file Notre référence
ISBN: 978-0-494-74234-1

Abstract

This thesis addresses evaluation methods used to measure the performance of machine learning algorithms. In supervised learning, algorithms are designed to perform common learning tasks including classification, ranking, scoring, and probability estimation. This work investigates how information, produced by these various learning tasks, can be utilized by the performance evaluation measure. In the literature, researchers recommend evaluating classification and ranking tasks using the Receiver Operating Characteristics (ROC) curve. In a scoring task, the learning model estimates scores, from the training data, and assigns them to the testing data. These scores are used to express class memberships. Sometimes, these scores represent probabilities in which case the Mean Squared Error (Brier Score) is used to measure their quality. However, if these scores are not probabilities, the task is reduced to a ranking or a classification task by ignoring them. The standard ROC curve also eliminates such scores from its analysis.

We claim that using non-probabilistic scores as probabilities is often incorrect, and doing it properly would mean imposing additional assumptions on the algorithm or on the data. Ignoring these scores fully, however, is also problematic since, in practice, although they may provide a poor estimate of probabilities, their magnitudes, nonetheless, provide information that can be valuable for performance analysis. The purpose of this dissertation is to propose a novel method that extends the ROC curve to include such scores. We, therefore, call it the scored ROC curve. In particular, we develop a method to construct a scored ROC curve, demonstrate how to reduce it to a standard ROC curve, and illustrate how it can be used to compare learning models.

Our experiments demonstrate that the scored ROC curve is capable of measuring similarities as well as differences in the performance of different learning models, and is more sensitive to them than the standard ROC curve. In addition, we illustrate our method's ability to detect changes in data distribution between training and testing.

Acknowledgment

In completing this dissertation, the quest for inspiration and imagination has lead me on a fascinating, and sometimes, cumbersome journey. I wish to acknowledge the vital support of many who contributed to my endurance and achievement. I thank my supervisors, Dr. S. Matwin and Dr. N. Japkowicz, for their commitment and support. I am grateful for their patience and wisdom that helped me grow able to carry out this research. I thank my examining committee for feedback that contributed to the success of my quest, and I thank Dr. C. Drummond for his insightful discussions that set the focus of this dissertation.

Part of this research was conducted during a visit to the Computer Science department, University of Bristol, UK. I thank my host, Dr. P. Flach for making the visit pleasant and productive. His inspiring supervision enabled me to develop the topic of this thesis. I wish to express my appreciation for the intellectual and financial support of Dr. W. Michaowski at Telfer School of Management, University of Ottawa. I also extend my gratitude to faculty members and colleagues at the school of Information Technology and Engineering who have been an ample source of encouragement and advice, particularly, Dr. S. Szpakowicz. Many thanks to my dear friends, Anna and Alex, for giving me strength and endurance. I am grateful to Dr. W. Dobosiewicz of Guelph University, and Dr. W. Armstrong of Dendronic Decisions Ltd., who encouraged me to pursue the path of scientific research.

Finally, I thank the Natural Sciences and Engineering Research Council of Canada (NSERC), the Government of Ontario, the University of Ottawa, and Ontario Centres of Excellence (OCE) for scholarships and financial support that made this research possible.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In his editorial to the Machine Learning journal, Langely (1988) argues that the main criteria to show learning relies on the demonstration of improvement in performance. His notion of performance-testing, thereafter, inspired the machine learning community to conduct experiments to assess the performance of learning methods to demonstrate their effectiveness. Nowadays, performance assessment has become essential to the development of applications and research in machine learning. In the literature, almost all machine learning contributions include an experimental performance testing component. Furthermore, extensive studies have been conducted to address this issue, and various workshops have been held to stimulate contributions in this area. Recently, the workshop series on evaluation methods for machine learning (Drummond et al., 2009, 2008, 2007, 2006) have focused primarily on this issue.

This research deals with evaluation methods that assess the performance of machine learning algorithms in supervised settings. We focus on how to measure and visualize similarities and differences between learning models based on the scores they assign to individual data points. Common machine learning tasks include classification, ranking, scoring, and probability estimation. This dissertation utilizes a comparative framework to organizes these learning tasks based on information relevant to their performance information. This organization, we argue, justifies the need for an evaluation method capable of measuring

the performance using the magnitudes of class membership scores. To address this problem, this thesis develops a novel evaluation method that extends the Receiver Operating Characteristics (ROC) Curve to include class membership scores. The resulting curve, we call it the scored ROC Curve, allows for the visualization of such scores while preserving performance information related to classification and ranking. Our experiments demonstrate the superiority of the proposed method over the standard ROC when measuring similarities and differences between learning models. Finally, our results show that our scored ROC method is sensitive to changes in the domain.

1.1 Motivation

Many popular machine learning methods produce models that assign class membership scores to individual instances. These scores represent the model's confidence in its predictions (Fawcett, 2003), and thus, they convey valuable information related to class memberships of data points. Respective classification decisions can be obtained by imposing a decision threshold on the scores. A ranking of the data points, from most to least positive, can be obtained by sorting them in the decreasing order of their scores (Fawcett, 2003). When measuring the performance of classification, most evaluation methods take into account the decision but ignore the confidence (the score value), and common methods that evaluate the ranking performance focus on the order of positives relative to the negatives. However, they also exclude the score magnitudes from their analysis.

This thesis proposes that class membership scores provide valuable information which should be considered by the performance measure. In a sense, the learning model produces decisions, as well as confidence in those decisions in the form of scores. Their values remain excluded from evaluation measures commonly used to assess classification or ranking performance. The following sections aim to motivate the use of class membership scores in the performance analysis by discussing their advantages and disadvantages, and by discussing applications that can potentially benefit from the proposed method.

1.1.1 The use of class membership scores

Advantages of using the class membership scores are best illustrated using a realistic example in an application domain. For instance, consider a learning system trained to decide whether a head injury patient should undergo a Computed Tomography (CT) scan in an emergency department of a hospital.

With the absence of scores, a classification model identifies two groups of patients; those who require CT scans and those who don't. The class membership scores can help establish a priority (an order) of patients to undergo CT scans. However, an order provides little distinction between patients based on their need for a CT scan. The potential harmful effects, especially among children, due to the exposure to radiations associated with CT imaging have been well documented (Brenner and Hall, 2007). Therefore, an emergency department physician may be concerned with assessing the need to perform these CT scans. Such an assessment can be performed using the magnitudes of class membership scores assigned to patients so that patients who have the highest score values may be recommended to undergo urgent CT scans, whereas patients with lower scores are referred to alternative measures. Patients with low score may be held for further observation or may be discharged. Clearly in this case, the physician relies more on the confidence in the decision rather than the decision itself. Perhaps this is one of the reasons why physicians prefer to deal with probabilities when making diagnoses.

Consequently, it is common for medical studies to use statistical methods to analyze probabilities of decisions rather than simple classifications. However, a great deal of analysis may be needed to verify assumptions they may impose. Modern machine learning methods may not provide the structure nor the mechanism needed to perform such verification. For example, a physician may wish to inspect the conditional probabilities of individual attributes, used by a Naive Bayes model for example, to verify their independence. In this case, she is able to verify the assumption of attribute independence because the learning model provides these probabilities. However, more complex learning models may need to be treated with the "black box" approach which makes this verification task more difficult.

Furthermore, common learning models offer little guarantees that the class membership scores they produce qualify for the consideration as probabilities. In the case of the Naive Bayes method, it has been shown that despite its effectiveness as a classifier (even when its assumptions are violated), it tends to produce poorly calibrated[1] scores (Bennett, 2000; Jiang and Zhang, 2006). Poorly calibrated scores make for poor probability estimates. Effectively, a potential lack of performance, coupled with the inability to verify assumptions, can lead to misleading results. Consequently, practitioners refrain from using these learning methods when there is little evidence of their expected performance.

A disadvantage of using class membership scores lies in the interpretation of their values. Referring back to our example of head injury patients, the decision of where to draw the boundaries on the score values to order a CT scan remains subjective at best, and it requires domain knowledge and expertise. If the scores are good probability estimates, then the physician will be able to interpret their values with the aid of her medical experience. However, when the scores are not probabilities, this interpretation becomes more difficult.

Despite the difficulty, if the physician is presented with a visualization of the scores to enable her to identify gaps between patients' scores, she may be alerted to potential similarities among groups of patient (those who are assigned similar score values). In this case, the physician may choose to focus on a group of patients for further assessment and to possibly draw more informative conclusions. The use of only classification decisions is limiting, and the application of ranking fails to show differences or gaps between patients. In order to focus on patients according to their needs for a CT scan, it is important that the magnitudes of their class membership scores be considered in the performance analysis.

Finally, including the scores into the evaluation process allows us to better evaluate learning models. With our method, learning models can be compared based on how well they distinguish between individual data points. For example, evaluation penalties for classification errors made by the learning model with low confidence (lower score values), should be less than those given for errors made with higher confidence (higher scores). In this case,

[1]Probabilities may be calibrated by comparing their predicted values to their observed values in the data

the model indicates weaker confidence in the former errors than it does in the latter. It is misleading to penalize the model for both errors equally. Therefore, this dissertation develops an evaluation method that extends the existing ROC method to include these scores. An advantage of this extension lies in preserving classification and ranking performance information while representing the magnitudes of the scores themselves.

1.1.2 Potential applications

Several applications and performance assessment strategies can benefit from the ability to visualize and compare class membership scores. We argue that the use of our evaluation method allows for making more informative performance-related observations based on score similarities and differences.

In addition to medical decision-making applications illustrated earlier, the problem of assessing user preferences represents another class of suitable application domains. This is a well-established problem, yet little is currently available to compare such preferences. For example, user preferences apply in domains where the task involves selecting movies to view, cars to buy, books to read, documents to retrieve (in information retrieval domains), etc. In these domains, classification decisions convey whether alternatives are preferred or not, and ranking the choices, from most preferred to least preferred, can be obtained by sorting them according to their assigned score values. However, distances between ranks (which can show gaps) cannot be obtained unless we consider the score values themselves.

Popular evaluation measures include accuracy, AUC, MSE score, and ROC curves. Accuracy depicts the frequency of correctly deciding whether a choice is preferred or not, but it fails to distinguish between choices in the same class. The AUC depicts the probability of a preferred choice being ranked higher than a non-preferred one but conveys little about how far apart these choices are. The MSE score portrays the overall error of the probability of a given choice being preferred by the user or not. These three metrics fail to indicate which choices, among several preferred choices, are more desirable than others. Arguably, these metrics are designed to produce a summary of performance, they are not meant to

provide such information. However, this leaves us with one evaluation method useful to assess the performance in such applications; the ROC curve visualizes the trade off between true positive rates and false positive rates for all classification threshold values. It can also be shown that each line segment along the ROC curve represents an individual instance, or in this case, a viable choice. However, the ROC curve depicts the performance based on the ranking order of choices. It ignores the magnitudes of scores, thus, it fails to show differences between choices within the same rank, i.e., it fails to show gaps or differences in score values. In line with our positioning, Cortes et al. (2007) argues that determining if one data point is preferred over another is insufficient, rather, one may desire an assessment of the magnitude of the preference between the two points.

For this reason, this research extends the ROC curve to include the magnitudes of class membership scores in a way that preserves ranking and classification performance information. The resulting curve, our scored ROC curve, depicts the combined performance of classification and ranking weighted by the magnitudes of the scores. Our evaluation method measures not only the correctness of classification and rankings, but also the preservation of preference magnitudes. Such problems have been introduced by Cortes et al. (2007) and are relevant to many applications including; search engine design, movie recommendation systems, and similar ranking systems.

1.1.3 A tool for research

Another advantage of our evaluation method lies in its empirical value as a research tool. This thesis shows that the scored ROC analysis is able to detect similarities and differences among learning models, and it can measure changes in the underlying data distribution. The experimental results, presented in this work, show that it does so better than the ROC analysis. This section argues that these abilities offer a useful tool for researchers.

In recent years, comparing the quality of class membership scores, produced by learning models, has captured the attention of many machine learning researchers. Their key interest lies in identifying which learning models produce better probability scores. In particular,

the Naive Bayes classifier has been widely criticized for producing poor probability estimates despite being an effective classifier (Bennett, 2000; Margineantu and Dietterich, 2002; Grossman and Domingos, 2004; Zhang and Su, 2004). Similarly, obtaining probability scores from probability estimating trees (PET) has become more common. The PET method represents a strong alternative to the Naive Bayes because the former is shown to produce better calibrated probabilities (Flach and Matsubara, 2007; Niculescu-Mizil and Caruana, 2005; Ferri et al., 2003; Provost and Domingos, 2003; Margineantu and Dietterich, 2002; Zadrozny and Elkan, 2002, 2001). To our knowledge, a method like ours remains unavailable. Consequently, researchers resort to either using metrics that provide a summary of the overall performance (Brier score or MSE), or they rely on plotting sigmoid functions that depict the smoothness of scores independent of their ranking performance (Niculescu-Mizil and Caruana, 2005). While the former provides a scalar summary insensitive to individual data points, the latter fails to depict the learning performance and measures the smoothness of scores only. Therefore, these studies can benefit from using our method because it allows them to draw more meaningful conclusions due to its ability to visualize score assignment characteristics while preserving the ranking and classification performance.

Machine learning algorithms assume that data is randomly drawn from the domain. The class distribution is assumed to remain unchanged between training and testing. Forman (2005) argues that, in practice, this assumption is often violated, which causes classifiers to struggle and probability estimates to deteriorate. Changes in the data distributions are not limited to class distribution only; learning in changing environments is a well-documented problem (Widmer and Kubat, 1996, 1998; Hand, 2006), and it is categorized by the type of change in the environment (Narasimhamurthy and Kuncheva, 2007). Changes include but are not limited to; population shifts (Kelly et al., 1999), concept drifts (Lane and Brodley, 1998), sample selection bias (Hand, 2006), and changing class definitions (Hand, 2006). Detecting similarities and differences between models constructed from multiple data samples allows for the assessment of how well data sets represent the domain. While training and testing models in static data distributions produce similar performance, models constructed

Table 1.1: Weather data from (Witten and Frank, 2005).

No.	outlook	temperature	humidity	windy	play
1	sunny	85	85	FALSE	no
2	sunny	80	90	TRUE	no
3	overcast	83	86	FALSE	yes
4	rainy	70	96	FALSE	yes
5	rainy	68	80	FALSE	yes
6	rainy	65	70	TRUE	no
7	overcast	64	65	TRUE	yes
8	sunny	72	95	FALSE	no
9	sunny	69	70	FALSE	yes
10	rainy	75	80	FALSE	yes
11	sunny	75	70	TRUE	yes
12	overcast	72	90	TRUE	yes
13	overcast	81	75	FALSE	yes
14	rainy	71	91	TRUE	no

and evaluated in changing environments may produce variations in performance.

Our second experimental component demonstrates the ability of the proposed method to detect changes in the data distribution. Bennett (2000) argues that such changes impact the quality of probability estimates learned in the domain. Section 4.3.2.1 of this dissertation presents how the proposed method determines, what we define as, the appropriateness of scores by comparing their values to their class labels. It can be shown that this process makes the method sensitive to the distribution of scores over data points. Thus, if the data distribution varies between training and testing, our method can be expected to reflect these changes in the class membership scores. Therefore, we test such property by conducting an experiment to detect performance differences in a changing environment.

1.2 Extending the ROC curve

To support of the motivations of this research, this section presents an example that compares information conveyed by both methods, the ROC and the scored ROC methods. It demonstrates information absent from the former but is depicted by the latter. This exam-

Table 1.2: Sample results for the *Weather* data.

No.	Label (play)	Classification (threshold = 0.5)	Rank	Score
7	yes	yes	1	1
3	yes	yes	1	1
13	yes	yes	1	1
12	yes	yes	1	1
9	yes	yes	2	0.96
5	yes	yes	3	0.93
10	yes	yes	4	0.89
6	no	yes*	5	0.66
4	yes	no*	6	0.49
1	no	no	7	0.43
8	no	no	8	0.30
11	yes	no*	9	0.29
14	no	no	10	0.04
2	no	no	11	0.01

* Incorrect classifications.

ple also demonstrates the extension of the standard ROC analysis to include the scores.

The example uses the *Weather* data used to decide whether to play a game based on observed weather conditions (Witten and Frank, 2005). Table 1.1 lists 14 sample days for which the weather conditions are recorded along with the decision label (*play*). A machine learning algorithm is applied to this data set to produce the sample class membership scores shown in the last column of Table 1.2. While sorting the instances by their score values produces the rankings shown in the second-last column, imposing a 0.5 threshold on the scores results in classification decisions listed in the middle column of the same Table.

First, consider the sample learning results shown in Table 1.2. If we restrict the focus to the classification results only, we are able to assess the accuracy of the decisions. However, we are unable to make any observations that relate instances to each other. Based on classifications only, we're unable to distinguish between instances seven and ten. If we expand the focus to include the order of results (without the rank values), we can observe that instance seven is ranked much higher than ten, because the learning model conveys

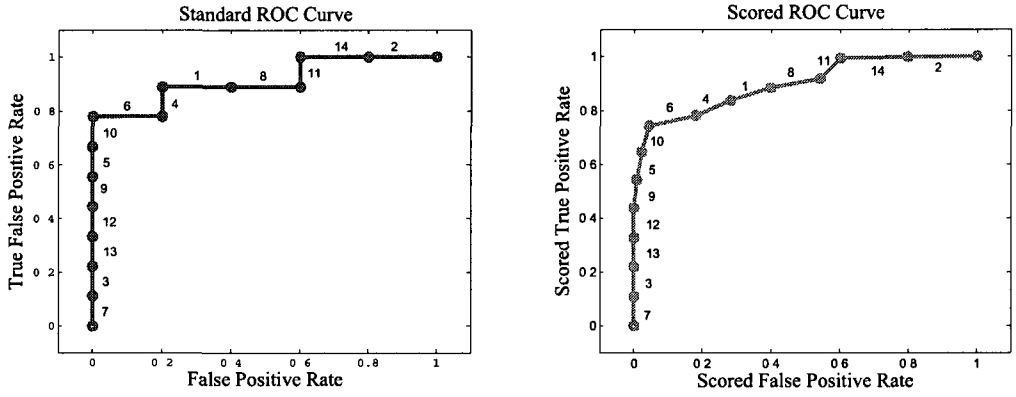

Figure 1.1: Visualizing the ROC curve (left) and the proposed scored ROC curve (right) for Table 1.2. Line segments correspond to indicated instances.

higher confidence in the former being more positive than the latter. The magnitude of this confidence becomes more evident if we consider the rank values which can be treated as scores. These rank scores allow us to determine distances between instances based on the confidence of the model is in its prediction. It is important to state that assessing the learning results by relating instances can be valuable because it allows for grouping the observation (the identification of similarly scored instances). For instance, if we consider the instances in the top rank (rank one), we may be able to recommend them for further investigation because the learning model puts them all in the same rank. However, the magnitude of this distinction cannot be determined by the rank values only. If we consider the class membership scores (last column of Table 1.2), we can see that the learning model clearly groups instances seven, three, thirteen, twelve, nine, and five in the top ten percent of the scoring scale $[0, 1]$. After instance ten, we can see a significant gap (0.89 - 0.66 = 23 percent) to the next score value. Clearly, the learning model provides strong confidence in this group of instances being positive.

However, not all scores are equally informative. For example, if we compare the rank values assigned to instances seven and ten, we can determine that they are three ranks-steps apart. However, such distance is dependent on observing instances nine and five. Had they been absent, instance ten would have been assigned rank two. Furthermore, scores

may be difficult to interpret for many reasons including, in the case of rank values, their lack of normalization. This suggests that the rank values are not necessarily good scores. Alternatively, if we compute the distance based on the class membership scores (the last column of Table 1.2), then, the distance between instances seven and ten is one minus 0.89 which equals to 0.11 regardless of other instances between them.

The above arguments demonstrate the ability of class membership scores to convey information relevant to the performance of the learning model with a higher resolution of discrimination between data points than that of classification and ranking. Chapter 3 positions this research based on a comparative framework that organizes common machine learning tasks using such information.

From a visualization perspective, the left plot of Figure 1.1 shows the corresponding ROC curve for results presented in Table 1.2. The ROC curve represents each instance by a line segment, and while those representing positive instances are drawn vertically, those that represent the negatives are plotted horizontally. The sequence of plotting these line segments follows the decreasing order of ranks starting at the origin (0,0) and ending at the point (1,1) in the space. The right plot of the same Figure shows the corresponding scored ROC curve for the same set of results. Plotting the scored ROC curves follows a similar strategy as the standard ROC with one difference: the slopes of individual line segments follow the magnitudes of class membership scores assigned to respective instances. The numbers beside each line segment, in both plots, show the corresponding instance numbers as listed in Table 1.2. Using the two curves, we proceed to illustrate information absent form the standard ROC curve but is added by the proposed scored ROC curve. The objective is to justify the extension of the ROC curve as a contribution of this thesis.

1.2.1 Information absent from the ROC analysis

The ROC curve is designed to depict the ranking performance. Visually, it displays an arrangement of line segments each of which represents an individual data point (see Figure 1.1). The ROC treats the positives differently than it does the negatives. However, it

fails to distinguish between individual positives, or individual negatives. This failure is evident in the left plot of Figure 1.1, where all the positives are drawn vertically and all the negatives are plotted horizontally. This lack of discrimination between individual points limits the ability of grouping similar points to one of two options: the correct class (positive or negative) or the correct ranking (positives before negatives).

Furthermore, inspecting the ranking values of the top seven rows of Table 1.2 reveals that the top four instances all have the same rank (one), and the subsequent three instances have three different ranks (two, three, and four). A quick glance at the ROC curve in the figure shows that their corresponding line segments form a vertical line. Within this line, the rearrangement of individual line segments does not impact the resulting ROC curve itself. Without considering their scores, any re-arrangement of these instances produces a vertical line. Effectively, this says that all the top seven instances have the same rank. We propose that this situation is misleading due to the failure of distinguishing between individual instances. A similar argument can be made for the last two line segments corresponding to instances fourteen and two (in the top right of the ROC curve). While their ranks are not equal, they are treated equally by the ROC curve because they are both plotted horizontally.

This inability to distinguish between instances extends to errors as well. For example, instances four and eleven are both misclassified positives which are treated equally by the ROC. However, the learning model provides less confidence in eleven being positive than it does in four. The former has a score of 0.29, almost half that assigned to instance four (0.49). The scores suggest that instance eleven is more likely a negative (it is closer to 0), whereas instance four is difficult to predict (its score is near 0.5). We propose that the error, made by the learning model in this case, is more evident for the former than the latter.

Ideal rankings, when all the positives are ranked higher than all the negatives, cause the ROC curve to make a single change in direction from climbing vertically to running horizontally. Concavities in the ROC curve indicate ranking errors. Every positive instance causes the curve to climb vertically by one step, and negative instances produce horizontal runs. When they are ranked incorrectly (negatives being ranked higher than positives), they

cause concavities (Flach and Wu, 2003, 2005). Such is the ROC curve segment starting with instance six and ending with instance eleven in Figure 1.1. There are two concavities, the first occurs between instances six and four, and the second appears between eight and eleven. These two concavities would have an identical dip in the curve if either instance one or eight was absent. This means that the difference between the two concavities relies on observing negative data points at higher ranks rather than their scores.

With a lack of sensitivity to the scores, it is difficult to identify gaps or measure differences between data points using the standard ROC curve. This difficulty is due to the omission of the scores from the ROC analysis. It can be argued that the ROC curve was designed to eliminate the scores from the process. In this section, however, we illustrate that the scores provide valuable information which the ROC is unable to depict. With this criticism, we do not advocate the replacement of the ROC method with a new one, but we justify its extension to include the scores to take advantage of its features. With such an extension, the proposed scored ROC curve adds prediction confidence information (the scores) to combine the evaluation of classification, ranking, and scoring by a single method.

1.2.2 Information added by scored ROC curves

The sample learning results listed in Table 1.2 are obtained by training and testing a learning model on data from Table 1.1. The corresponding ROC and scored ROC curves are presented in the left and in the right graphs of Figure 1.1 respectively.

Following arguments made in the previous section, we begin by considering the top seven instances of the results which correspond to line segments along both curves starting at the origin. In the scored ROC curve (in the right plot of the Figure), we can see that the first four instances produce vertical line segments because their scores are equal to one. Changing their order produces little effect on the curve. However, line segments corresponding to the subsequent three instances (number nine, five, and ten) have gradually increasing slopes and cause the curve to become more convex. This effect is a result of plotting the slopes proportional to the scores. Placing any of these three instances in between the previous

four (those with rank one) results in a concavity along the curve because their line segments are not vertical due to their scores being less than one. Effectively, the scored ROC curve makes a visual distinction between these instances whose scores differ. This conclusion is different than that previously drawn from the ROC curve, and it shows that the shape of the scored ROC curve does not depend on the ranking order only, but also on the magnitudes of the scores that express the confidence in predictions. Similarly, the scored ROC shows that instances fourteen and two have similar scores. Their corresponding line segments are almost horizontal. They actually have different slopes of very small values. This says that these two instances are negative with very little difference between them. Thus, the scored ROC curve depicts similarities and differences among instances by comparing the slopes of their corresponding line segments.

Following the same rationale as in the previous section, the two ranking errors (instances four and eleven) produce two corresponding concavities in the scored ROC curve as well (see the right plot of Figure 1.1). The first concavity appears between line segments six and four, and the second is between eight and eleven. However, this second concavity is more pronounced than the first one due to the pairwise difference in their corresponding scores (compare their corresponding scores in Table 1.2). The scored ROC curve weights each line segment by the score assigned to the data point it represents. This weighting process is described in details in Chapter 4. Visually and unlike the standard ROC, the scored ROC curve is able to distinguish between data points based on their score values despite their incorrect order.

The added ability to discriminate between any pair of instances based on their scores allows the scored ROC curve to identify and to measure score gaps and differences. For example, the curve in the right plot of Figure 1.1 makes a sudden and pronounced change in direction between line segments ten and six. This says that the scores assigned to their associated data points have dropped dramatically. In addition, another pronounced change in direction occurs between line segments eleven and fourteen also on the same curve. In Table 1.2, the two highest score gaps are found between instances ten and six, and between

eleven and fourteen. Hypothetically, these two score gaps can be used to partition the data set into three distinct groups of instances. The top seven instances appear to be similar when considering their scores. Their line segments along the scored ROC curve have similar slopes. The next group of points begins with instance six and ends with instance eleven, and the final group consisting of the remaining two instances whose scores are lowest.

In summary, this example illustrates that the standard ROC curve fails to distinguish between individual data points in one class, and that our extension to it enables the resulting scored ROC curve to be sensitive to the scores assigned to individual data points while preserving their ranking performance information. In addition, this example demonstrates the ability of the proposed method to identify gaps and measure differences between individual data points. This ability allows for the grouping of data points based on the confidence of the learning models in its predictions. This example shows that this information is absent from the ROC curve and is added by the scored ROC curve.

1.3 Existing performance measures

In this section, we look at other ways of evaluating the performance of learning models. We highlight their shortcomings and discuss how our method does not suffer in the same way.

The accuracy metric is well-established and commonly used to measure the frequency of making correct classification decisions. Accuracy is an overall rate of correct classification decisions for all classes. This makes it prone to reporting inaccurate performance when faced with highly imbalanced data (Provost et al., 1997). Accuracy has also been shown to be less consistent and less discriminant between classes than the area under the standard ROC curve (AUC) (Ling et al., 2003b). Other scalar metrics can also be used to analyze the performance of classification; these include precision and recall which can be combined to produce the F-measure. Such metrics report the performance for individual classes, however, the main disadvantage of using them, with respect to this research, is that they are designed to produce an overall scalar summary of performance. Therefore,

they are unable to distinguish between individual data points. Evaluation methods that produce a two-dimensional representation of performance may be able to address the above disadvantage. These include; lift curves (Flach, 2004), DET curves (Fawcett, 2003), relative superiority graphs and LC index (Fawcett, 2003), and cost curves (Drummond and Holte, 2006). However, these methods assess the performance based on the decisions but ignore the confidence of the model represented by class membership scores.

Recently, Wu et al. (2007) proposed a method to incorporate score margins into the AUC, called sAUC. Their idea is to detect the existence of fixed-size score gaps between positive and negatives data points. Effectively, this measures how quickly the AUC, under the standard ROC curve, deteriorates when the scores assigned to the positive instances are decreased. They argue that when evaluating the performance of binary classification, both scores and ranks should be combined. This leads to an important feature of the sAUC method in that it incorporates score-related information into the AUC metric. However, our method differs from theirs in many ways. While our method weights the standard ROC curve by the absolute values of the scores, theirs relies on score differences being greater than a fixed gap. In addition, they measure these score gaps only between positive and negative data points, which limits their assessment to the positives versus the negatives. In contrast, our method relies on the values of the scores themselves, which enables the visualization of score differences and gaps of any size between any pairs of instances regardless of which class they belong to, i.e., our method can distinguish between data points in one class, theirs cannot. This says that we report a different kind of performance information than they do. Our method depicts the performance of ranking combined with the magnitude of confidence in predictions conveyed by the class membership scores. Their sAUC method relates to the behavior of the AUC, under the standard ROC, if the positive scores are decreased. Furthermore, our method develops the actual scored ROC curve, and they only develop the sAUC metric which is a summary of a different kind of performance.

The above performance measures are designed to evaluate the performance of classification and ranking tasks only. Probability estimates tasks are evaluated by the mean squared

error (MSE), also known as, Brier score (Brier, 1950). However, the Brier score takes into account probability estimates but not rankings, and the AUC accounts for the ranking performance but not the scores (Flach and Matsubara, 2007). Our method measures the performance while taking into account all three tasks of classification, ranking, and scoring. Probability estimating tasks remain a separate issue when it comes to performance analysis.

Evaluating scores that are not probabilities is a subjective task by nature. Their interpretation impacts the selection of the performance measure. In the current state, there is little to no evaluation measures that can assess the quality of scores, which leaves a gap in the area of performance analysis. Our work aims to address this gap and proposes a visualization method that allows users to compare scores, so that they can draw informative conclusions. Our experimental results illustrate the ability of our proposed method, the scored ROC method, to detect similarities as well as differences between learning models based on the scores they produce. In addition, our experiments show that the proposed method can measure such similarities and differences better than the standard ROC.

1.4 Research Contributions

This research makes several novel contributions on two levels: methodological and practical. While the former lies in the curve construction, strategies, and analysis presented in this thesis, the latter relates to the use of our method to conduct scientific investigations. The specific contributions of this dissertation include:

- The use of a comparative framework of learning tasks in which:
  - a learning task conveys performance information in its outcome,
  - performance information is expressed with various granularity and can be abstracted or specialized based on the desired learning task, and
  - the performance analysis makes use of such information without eliminating information or imposing additional assumptions.

- The construction of an evaluation method that:
  - extends the classic ROC analysis to include class membership scores,
  - provides an evaluation and visualization method based on individual scores,
  - combines the performance of classification, ranking, and scoring,
  - measures similarities and differences between learning models,
  - is sensitive to data in changing environments, and
  - enhances the detection of performance similarities and differences among models constructed from:
    * benchmark data,
    * data sampled from static distributions, and
    * data drawn from a changing domain.

1.5 Organization

This thesis contains 6 chapters. Chapter 2 provides a comprehensive review of evaluation methods available in the machine learning literature. The objective is to present a technical background of our research area. Chapter 3 presents arguments that position this research among related existing work in the machine learning community. This positioning is centered on a detailed discussions of the comparative framework used to organize common machine learning tasks based on performance-related information. Chapter 4 presents our novel method that incorporates class membership scores into the ROC curve. It also illustrates how a scored ROC curve is constructed as an extension of an existing algorithm that plots the ROC curve. In addition, this chapter presents a proof that a scored ROC curve can be reduced to a standard ROC curve by eliminating the scores from the analysis.

Chapter 5 presents experimental design and results in two components. The first discusses empirical results of measuring performance similarities and differences between various learning models. These models are constructed from benchmark data set obtained

from the UCI repository (Asuncion and Newman, 2007). The second empirical component illustrates the abilities of the scored ROC method in changing domains. Such changes result in performance differences between learning models which, we show, are captured by our method. In both components, we compare conclusions made by the scored ROC curve against those made using the standard ROC curve. Finally, Chapter 6 concludes with a summary of results, analysis and achievements made by this research. The chapter also explores possible future research directions including issues related to the design of our method and to its potentials in assessing the performance of learning models.

Chapter 2

Background

This chapter provides a technical background needed for this research. It reviews principles and techniques related to the development of the proposed evaluation measure. Such principles include classification, ranking, and estimating class membership scores which may be probabilities. We review a collection of evaluation methods commonly used to evaluate the above learning tasks. Among them is the Receiver Operating Characteristics (ROC) curve analysis on which we focus a great deal. The reason for this focus is that our proposed curve analysis method extends the ROC curve to include class membership scores.

2.1 Classification, ranking and scoring

Following the description of Flach and Matsubara (2007), we let X be a set of n instances where the i^{th} instance is a vector x_i of values for attributes $A_1, \cdots A_n$.

In a classification task, a *classifier* is a mapping $\hat{c} : X \to C$ where C is a set of labels with binary classification being a special case where $C = \{+, -\}$. A *scoring classifier* is a mapping $\hat{s} : X \to \mathbb{R}$ which estimates and assigns scores $\hat{s}(x_i)$ for each $x_i \in X$. For simplicity, we assume that the score $\hat{s}(x_i)$ relates to the expectation of instance x_i being positive; therefore, instances with higher scores have higher expectation of being positive than those with lower scores. Consequently, ordering instances in X by a decreasing order

Table 2.1: The confusion matrix for testing a learning model.

		Predictions Y	Predictions N	
Labels	+	tp	fn	n^+
	−	fp	tn	n^-
		p^+	p^-	n

of their $\hat{s}$ scores produces a ranking of X. When a scoring classifier produces scores as probability estimates of instances being positive, then it is called a *probability estimator* which produces a mapping $\hat{p} : X \rightarrow [0, 1]$, where $\hat{p}(x)$ is taken as an estimate of the posterior $p(+|x)$.

Ranking is an order with potential ties represented by an equivalence relation over X (Flach and Matsubara, 2007). A ranker is a function represented as $\hat{r} : X \times X \rightarrow \{>, =, <\}$ that decides, for any pair of examples, whether the first is more likely ($>$), equally likely ($=$), or less likely ($<$) to be positive than the second. It is important to point out that a ranker may or may not produce scores for instances; thus, a *scoring ranker* produces scores, which may well be probabilities, then ranks the instances in their order of score values. A *non-scoring ranker* produces an order or ranks of instances without providing any scores.

Binary classifications can be obtained from scores, ranks, or probabilities, by using a fixed threshold to decide whether data points are above or below the decision threshold. Those that are above (or equal to) the threshold are classified as positives, otherwise, they are classified as negatives. This shows that a classification task ignores the magnitudes of ranks, scores, or probabilities and focuses on the class labels. In ranking tasks, the emphasis is on the order of data points, and in scoring, the objective is to estimate class membership scores which may represent probability estimates.

2.2 Performance Analysis

The process of evaluating the performance of learning involves training a learning model (a classifier, a ranker, or a probability estimator) on a set of training examples. Once trained, the resulting model is asked to make predictions on a set of test instances whose labels are known (supervised learning). Thereafter, the corresponding predictions and their labels are compared to determine if they match. This comparison produces the confusion matrix shown in Table 2.1. The rows of this matrix represent the number of test instances that are labeled positive or negative, denoted by n^+ and n^- respectively. Therefore, the size of the test data set n is equal to $n^+ + n^-$. The columns of Table 2.1 represent the number of instances predicted by the learning model as positives and as negatives, entries p^+ and p^- also respectively. This confusion matrix depicts the correct and incorrect proportions of the predictions.

When comparing predictions to class labels, there are four possible outcomes for each instance in the test set. A correctly classified instance is one whose label matches its corresponding prediction made by the learning model. Thus, this instance may represent a true positive or a true negative which are denoted by tp and tn entries in Table 2.1 respectively. An incorrectly classified instance can be a positive instance, classified as negative, or a negative instance classified as positive. These are known as false negatives or false positives respectively, and they are denoted by fn and fp entries in Table 2.1 also respectively. Therefore, the confusion matrix shows the proportions of true positives, false negatives, false positives, and true negatives instances resulting from testing the learning model on the particular test data.

It is clear that such a confusion matrix reports the correctness of classification decisions made by the learning model; however, a scoring classifier, a scoring ranker, or a probability estimator produce class membership scores, rankings, or probabilities for test instances respectively. In such cases, generating the confusion matrix may be less obvious. Such tasks need to be turned into a classifier to produce a confusion matrix, and therefore, a

fixed threshold is needed to make classification decisions. The process is intuitive, if the score of a particular instance is greater than, or equal to, the threshold value, it is classified as a positive instance, otherwise, it is classified as a negative instance, and entries to the confusion metric can be populated.

Any performance metric that relies on entries from this confusion matrix, shown in Table 2.1, is susceptible to ignoring the magnitudes of the scores, the class membership scores assigned to instances by the learning model, and it can only base its measure of performance on classification decisions. The reason for such susceptibility is that these scores are used for the sole purpose of comparison against the classification threshold, then the entries summarize the observed correct and incorrect classifications. The most commonly used evaluation methods in machine learning, other than Brier (1950) score or MSE and Information Gain (Kononenko and Bratko, 1991), begin with the population of such confusion matrices. Therefore, most fail to account for score magnitudes in their analysis.

Other useful rates can be computed from this confusion matrix and its entries. These are shown in Equations 2.1, 2.2, 2.3, 2.4, and 2.5.

$$\text{True Positive Rate } tpr = \frac{tp}{n^+} \tag{2.1}$$

$$\text{False Positive Rate } fpr = \frac{fp}{n^-} \tag{2.2}$$

$$\text{False Negative Rate } fnr = \frac{fn}{n^+} \tag{2.3}$$

$$\text{True Negative Rate } tnr = \frac{tn}{n^-} \tag{2.4}$$

$$\text{Class Skewness } c = \frac{n^-}{n^+} \tag{2.5}$$

Subsequent sections review commonly used evaluation metrics, used by machine learning researchers, to measure the performance of common learning tasks including classification, ranking, or probability estimation.

2.2.1 Classification performance

Perhaps the most commonly known performance metric, is the accuracy. It is widely used due to its natural definition that measures the ratio of correctly classified instances in a data set. This metric reports the proportion of data which is correctly classified as a percentage. Classification error can be computed by 1 minus the accuracy. For completeness, the accuracy and the error rate are computed using entries obtained from the confusion matrix shown in Table 2.1 and substituting them in Equations 2.6 and 2.7 respectively.

$$\text{Accuracy} = \frac{tp + tn}{n} \tag{2.6}$$

$$\text{Error Rate} = 1 - \text{Accuracy} = 1 - \frac{tp + tn}{n} = \frac{fn + fp}{n} \tag{2.7}$$

In principle, the accuracy metric measures classification performance. When the learning model is a classifier, the metric can be used after the confusion metric is populated; however, if the model produces class membership scores, which may be used for ranking or as probabilities, it needs to be converted into a classifier in order to use the accuracy metric. This conversion may defeat the purpose of constructing a ranker or a scoring model. The interest of constructing such models is in the rankings or in the probabilities it produces.

In recent years, the accuracy metric has come under heavy criticism by many studies (Provost et al., 1998; Lavrac et al., 1999; Cortes and Mohri, 2004; Fawcett, 2003; Flach, 2003; Ling et al., 2003b; Wu et al., 2007; Law, 2008). Ling et al. (2003a) shows that accuracy is inappropriate for measuring learning performance due to its inconsistency and its inability to discriminate between classes as per their definitions. In the general view of evaluation, accuracy has several weaknesses that has many researchers looking for an alternative evaluation metric, particularly, the AUC metric described later in this chapter. The first shortfall of accuracy lies in that it requires a fixed threshold on the values of the scores (ranks or probabilities). The fixed threshold results in the accuracy's inability to respond to changing class or cost distributions (Wu et al., 2007). Obtaining good accuracy for a given threshold value is hardly relevant to the accuracy obtained with another. Furthermore,

when the number of instances belonging to the majority class in the data set is significantly larger than those in the other class, it is easy to obtain good performance with accuracy regardless of the model's ability to distinguish between the classes. This shows that accuracy handles class distribution imbalance poorly and fails to assess the performance properly.

To deal with this problem, Flach (2003) measures the accuracy weighted by class skew and as defined in Equation 2.8. Class skewness is the ratio of class distribution $c = \frac{n^-}{n^+}$. When $c = 1$, the resulting accuracy metric is reduced to Equation 2.9 and is interpreted by the average classification accuracy on the positive and the negative classes.

$$\text{Class-Skew Sensitive Accuracy} = \frac{tpr + c(1 - fpr)}{1 + c} \tag{2.8}$$

$$\text{Class-Skew Insensitive Accuracy} = \frac{tpr + 1 - fpr}{2} = \frac{tpr + tnr}{2} \tag{2.9}$$

Also in dealing with class distribution imbalance, Lavrac et al. (1999) computes an alternate version of accuracy known as the Weighted Relative Accuracy (WRAcc). The weighted relative accuracy can be computed sensitive or insensitive to class skew as shown in Equations 2.10 and 2.11 respectively.

$$\text{Weighted Relative Accuracy Insensitive to Class Skew} = tpr - fpr \tag{2.10}$$

$$\text{Weighted Relative Accuracy Sensitive to Class Skew} = \frac{4c(tpr - fpr)}{(1 + c)^2} \tag{2.11}$$

In information retrieval research, other metrics are used to analyze the quality of information being retrieved. In such tasks, the learning model should retrieve an appropriate volume of information precisely. This measurement is known as recall and precision, and can be combined by the F-measure. These two metrics have been used to evaluate the performance of document classification as relevant or irrelevant (Lewis, 1990, 1991). The assumption, in this case, is that the set of documents remains static and does not change over time. Such an assumption may not hold in many applications. While recall measures the proportion of successfully retrieved relevant documents, precision depicts the proportion

of relevant documents among those retrieved. Using entries in the confusion metric of Table 2.1, precision and recall are defined by Equations 2.12 and 2.13 respectively.

$$Precision = \frac{tp}{p^+} \tag{2.12}$$

$$Recall = \frac{tp}{n^+} \tag{2.13}$$

Information retrieval tasks may be designed to produce weights, scores, or probabilities to indicate the relevance of documents to the query. to help determine a ranking of documents ranging from most relevant to least relevant. The result is a list of ranked documents that may be cut off at some point to eliminate the large volume of undesired documents because a query often retrieves a large collection of documents. In text classification, however, a collection of documents belonging to the desired class is readily available for training, whereas in information retrieval applications, relevant and irrelevant documents are defined by a query used to retrieve them. At the time of retrieval, some documents may not be available for training, or the query may be refined, after training, to better retrieve relevant documents. Therefore, the collection of training document can vary over time. This situation suggests that the class distribution may change, which makes the simultaneous optimization of precision and recall difficult (Fisher et al., 2004).

To understand such problems, researchers use a two-dimensional representation of the trade-off between precision and recall over all threshold values, known as precision-recall curves. The idea is to plot values of precision against values of recall for all threshold values. Fawcett (2003) shows that, like accuracy, precision and recall curves suffer from the same effect when it comes to class skewness, when one class is more prevalent than the other. Conclusions about classifier performance made by precision and recall metrics differ when there is a shift in the class distribution. Similar to the accuracy metric, precision can be weighted to become sensitive to class skewness (Flach, 2003) as shown in Equation 2.14.

$$\text{Class-Skew Sensitive Precision} = \frac{tpr}{tpr + c\dot{f}pr} \tag{2.14}$$

Table 2.2: Sample class membership scores.

Labels	Scores
+	0.80
+	0.70
-	0.60
+	0.40
-	0.30
-	0.20

The F-measure is a metric which combines both precision and recall into a single scalar metric by averaging fp and fn rates (Rijsbergen, 1979; Flach, 2003). It is defined by Equation 2.15, and like accuracy and precision, the F-measure can also be modified to become sensitive to class skewness as shown in Equation 2.16 (Flach, 2003).

$$\text{Class-Skew Insensitive F-measure} = \frac{2tpr}{tpr + fpr + 1} \tag{2.15}$$

$$\text{Class-Skew Sensitive F-measure} = \frac{2tpr}{tpr + c\dot{f}pr + 1} \tag{2.16}$$

Finally, class skewness is related to class priors that represent the probability of obtaining a positive instance from the domain. Kononenko and Bratko (1991) presents an evaluation method that computes the information gain based on incorporating class priors into the evaluation metric which uses the number of correct classifications. Their idea is based on calculating the entropy of information gain and assigning a relative score for a learning model applied in a given domain. Like Brier score (Brier, 1950), this information gain-based evaluation measure represents a summary of performance rather than assessing individual instances.

2.2.2 Ranking performance and the ROC analysis

The Receiver Operating Characteristics (ROC) curves have been used to visualize and organize classifiers based on their ranking performance (Fawcett, 2003). The ROC curve

originates in signal detection theory to represent the trade-off between hit rates and false alarm rates (Egan, 1975; Swets et al., 2000) and has been used to analyze the accuracy of diagnostic systems (Swets, 1988) and in medical studies (Zou, 2002).

The first introduction of ROC curves in machine learning is presented by Spackman (1989) who shows how to use their analysis to evaluate and to compare learning models. Due to their wealth of information and robustness to class skewness (unequal misclassification costs), they have become increasingly more popular (Fawcett, 2003; Flach, 2003, 2004).

The ROC curve produces a visual representation of the performance associated with a prediction model, whether it is a classifier (scoring or non-scoring), a ranker (scoring or non-scoring), or a probability estimator. The ROC curve is constructed from classification decisions using entries in a series of confusion matrices similar to that illustrated in Table 2.1 on page 21. When generating an ROC curve, a threshold on class membership scores, produced by the learning model subject to the evaluation, is imposed to obtain classification decisions for instances to populate the confusion matrix. Varying this threshold from 0 to 1 produces several classifiers each with its own confusion matrix, and for which the true positives rates are plotted against the rates of the false positives. Thus, each threshold value is plotted as a point. Joining the points builds the ROC curve based on classification decisions made by the model for all possible threshold values.

For example, Table 2.2 gives sample scores produced by a learning model on six sample instances. The corresponding ROC plot is shown in Figure 2.1. The ROC space possesses particular features. Each point in the ROC space represents a classification result or a confusion matrix. Point $(0, 0)$ is the trivial classification of only negative predictions, which results in the zero values for the true positive rate and also for the false positive rate. Similarly, point $(1, 1)$ represents the trivial classification of only positive predictions where all positive instances are classified correctly and all negative instances are incorrectly classified as positives. The increasing diagonal line represents the random performance where the classification of an instance as positive or as negative is based on random chance. The top left corner, point $(0, 1)$, is the point of optimal performance because it depicts a 100% true

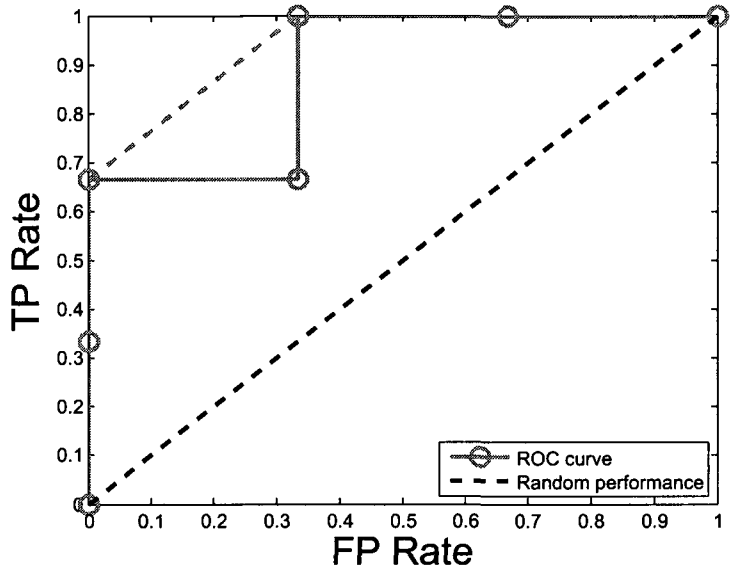

Figure 2.1: The ROC Curve for Table 2.2

positive rate and a 0% false positive rate. Finally, point $(1, 0)$ represents the performance of a learning model whose predictions are always inverted, i.e., one that always classifies positive instances as negatives and vice-versa.

The performance of a learning model is desired to be above the diagonal random performance line toward the north-west corner of the plot. In this case, it can be said that the performance is better than random. If the performance of the model is situated toward the bottom-right corner of the graph and below the random diagonal, it can be said that the performance is poor. However, a model whose performance is below random and is located toward the bottom right corner of the space, arguably, contains useful information that is being used incorrectly (Flach and Wu, 2003). In this case, inverting its predictions will invert the performance about the diagonal line.

A learning model whose performance is close to random represents the worst case because its performance is of little use. This significance question is addressed by Forman (2002). As an illustration, consider Figure 2.2 in which model D exhibits perfect classification performance, model E shows better classification performance on the negative class, models

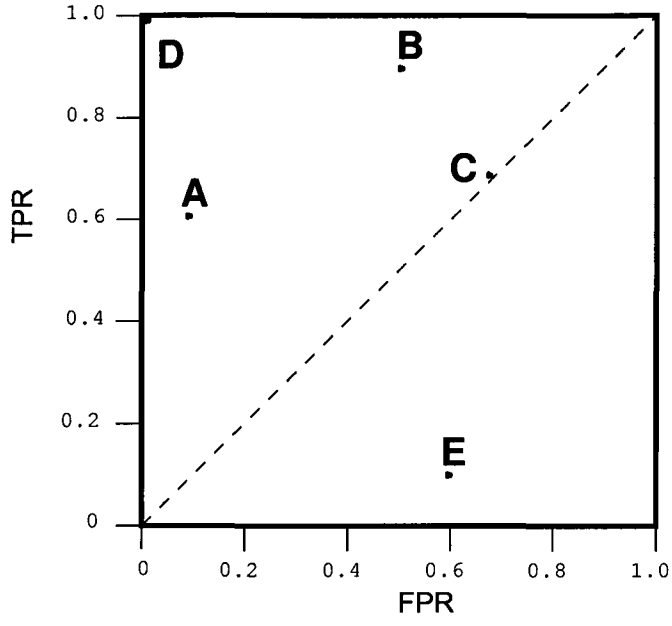

Figure 2.2: Comparing classification models in the ROC space (Fawcett, 2003).

A and B show the trade-off between tpr and fpr in their classification, and finally, model C shows classification performance very close to random. The latter is considered the worst classification performance.

We saw in Section 2.1 that a learning model can be designed to produce more than just class labels. Only few learning models, particularly rule-based models, produce class labels only. Fawcett (2003) calls such models "*discrete*" classifiers. Many learning algorithms are designed to produce class membership rankings, scores, or probabilities for instances, and they can be turned into classifiers. Imposing a varying threshold value on these scores results in a series of confusion matrices which correspond to points in the ROC space. Connecting such points of adjacent threshold values is valid and depicts a random performance between them due to the lack of knowledge of the performance between them (there is no other confusion matrix available in between the two points). This situation is illustrated in Figure 2.3 which shows the connection of two such points in the ROC space. While the

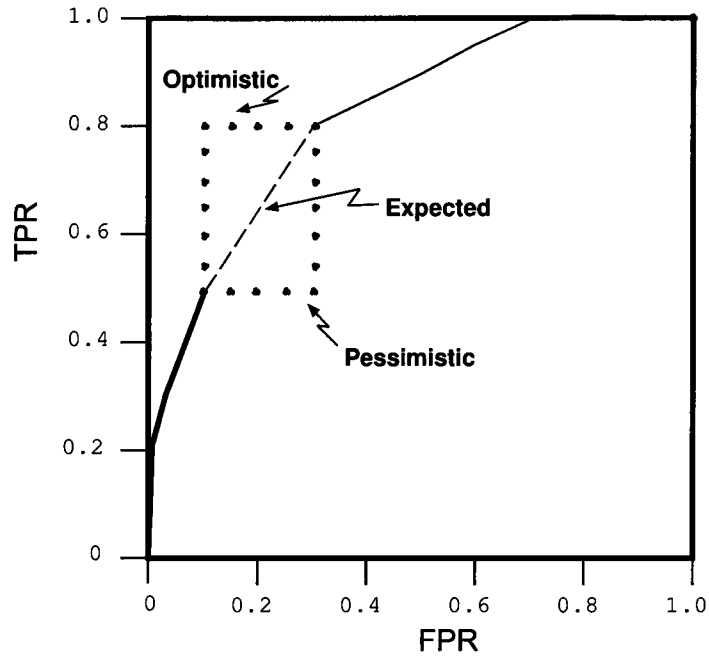

Figure 2.3: Connecting points to form an ROC curve (Fawcett, 2003).

upper part of the dotted box is considered an optimistic estimation of best classification performance, the lower part represents a pessimistic expectation. The dashed line shows the random performance. In this manner, the sequence of points in the ROC space forms the ROC curve, which represents the performance for all threshold values (Wu et al., 2007).

While the performance of a single model can be examined by inspecting its ROC curve, the performance of multiple models can be compared by comparing their respective ROC curves. For instance, consider the ROC curves shown in Figure 2.4. Model B, in the left plot of the figure, exhibits better performance than model A because its corresponding ROC curve dominates that of model A in most of the space except in the top right corner, there, model A shows better performance than B. However, model B shows better average performance than model A. In the right plot of Figure 2.4, the ROC curve associated with model B clearly dominates that of model A, because all points belonging to B are located

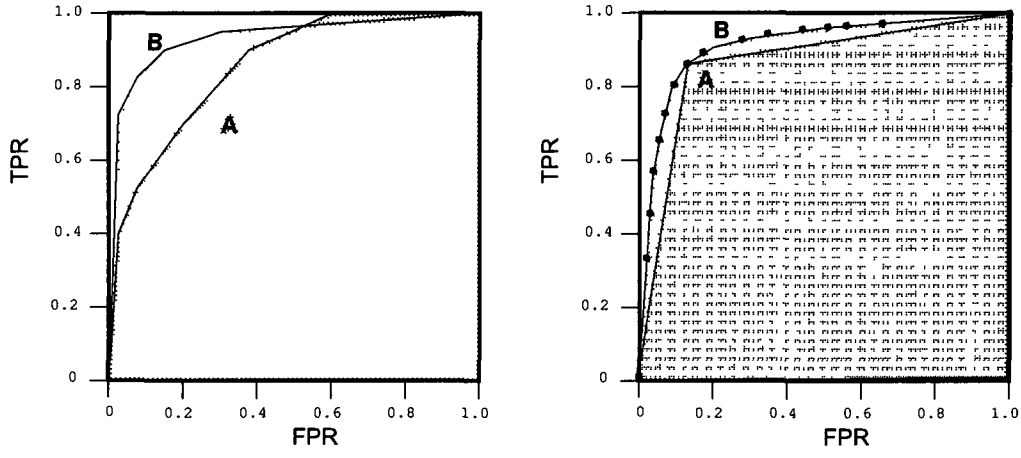

Figure 2.4: ROC curve dominance (Fawcett, 2003).

above (or the same as) those of A.

If we consider points on the individual ROC curves, we observe that model A, in the right plot of Figure 2.4, represents a discrete model which produces two scores. This is depicted by two line segments connecting a single ROC point to $(0,0)$ and to $(1,1)$ on either side. Furthermore, model B in the same plot has several line segments connecting many ROC points that form the ROC curve. This suggests that model B is a scoring classifier that produces several score values (in this case B is a probabilistic classifier). The analysis of ranks, scores, and probabilities in the ROC space are presented in more detail in Section 2.2.6 of this chapter.

2.2.3 Area Under the Curve (AUC)

To simplify the process of comparing models, the ROC curve can be reduced to a one-dimensional scalar metric by computing the area under the curve, the AUC (Hanley and McNeil, 1982; Bradley, 1997). This section describes the AUC scalar metric which aggregates the performance over all decision thresholds into a single scalar value (Fawcett, 2003; Wu et al., 2007). Traditionally, classification error has long been used to measure the per-

formance of classification methods (Mitchell, 1997). However, in recent years, arguments against the accuracy (Provost et al., 1998; Ling et al., 2003b) have concluded that the AUC is a better performance measure than accuracy (Ling et al., 2003b).

The AUC metric has gained popularity with the introduction of ROC curves to the machine learning community (Spackman, 1989; Fawcett, 2003). perhaps, the premise of its popularity is due to its relation to the Wilcoxon-Mann-Whitney statistic Hanley and McNeil (1982). Flach (2007) suggests four equivalent interpretations for the AUC as: *the expectation that a uniformly drawn random positive is ranked before a uniformly drawn random negative, the expected proportion of positives ranked before a uniformly drawn random negative, the expected true positive rate if the ranking is split just before a uniformly drawn random negative*, and *the expected proportion of negatives ranked after a uniformly drawn random positive.* Intuitively, the above defines the AUC as a metric to measure the ability of distinguishing between the positive and the negative class in a binary class setting. When it comes to classification problems, the AUC has been shown to be a better measure (Ling et al., 2003b; Provost et al., 1998) and a classifier can be developed by optimizing the AUC for the purpose of classification (Calders and Jaroszewicz, 2007). Balcan et al. (2007) establish a relationship between the AUC and accuracy based on reducing the ranking task to a classification task.

The estimation of the AUC can be done under parametric (Zhou et al., 2002), semi-parametric (Hsieh and Turnbull, 1996), and non-parametric (Hanley and McNeil, 1982) assumptions. The latter is mostly used in machine learning by a series of summation of the area of trapezoids formed by connecting point on the ROC curve (Wu et al., 2007; Fawcett, 2003). The AUC is equivalent to the Wilcoxon-Mann-Whitney U statistic test of ranks (Hanley and McNeil, 1982). The remainder of this section provides a detailed description of how the AUC can be calculated.

The AUC coincides with the Wilcoxon-Mann-Whitney statistic and the probability of correct pairwise rankings (Hanley and McNeil, 1982). For instance, if x_i^+ is a randomly chosen positive example and x_j^- is a randomly chosen negative, and if $r(x_i^+)$ and $r(x_j^-)$

Table 2.3: Sample data X with scores s_i and rankings r.

i	Label	s_i	r_i	i	Label	s_i	r_i
1	+	1.0	4	11	+	0.4	2
2	+	1.0	4	12	+	0.4	2
3	+	1.0	4	13	-	0.4	2
4	+	1.0	4	14	-	0.4	2
5	-	1.0	4	15	-	0.4	2
6	+	0.8	3	16	+	0.0	1
7	+	0.8	3	17	-	0.0	1
8	+	0.8	3	18	-	0.0	1
9	-	0.8	3	19	-	0.0	1
10	-	0.8	3	20	-	0.0	1

are their corresponding ranks respectively, then, the AUC is the probability of $r(x_i^+)$ being greater than the probability of $r(x_j^-)$, which is the probability of $s_i > s_j$ where s_i and s_j are the scores assigned to x_i and x_j respectively. This represents the probability of $(r(x^+) > r(x^-))$, and is calculated by Equation 2.17.

$$AUC = \frac{1}{n^+ n^-} \sum_{i=1}^{n^+} \sum_{j=1}^{n^-} I(x_i^+, x_j^-) \tag{2.17}$$

where

$$I(x_i^+, x_j^-) = \begin{cases} 1 & \text{if } r(x_i^+) > r(x_j^-) \\ \frac{1}{2} & \text{if } r(x_i^+) = r(x_j^-) \\ 0 & \text{if } r(x_i^+) < r(x_j^-) \end{cases} \tag{2.18}$$

To illustrate how the AUC is calculated, consider sample data set X shown in Table 2.3. Each instance x_i is ranked based on its associated score where equal scores have the same rank. For simplicity, we produce rankings r in a decreasing order to indicate that higher ranks are towards the top of the list which reflects higher likelihood of being positive. For instance, an instance x_i whose rank $r(x_i) = 4$ is expected to be positive more than an instance x_j whose rank $r(x_j) = 1$. For each rank r, we calculate entries of the columns

Table 2.4: AUC Calculations for Table 2.3. The columns represent the rank r, the classification threshold t, the instance score s, the number of positive instances n^+ ranked at r, the number of positive instances n^+ ranked above r, the number of negative instances n^- ranked at r, the number of negative instances n^- ranked below r, the true positive rate tpr for t, and the false positive rate fpr for t. The ROC curve is shown in Figure 2.5.

			A	B	C	D		
r	t	s	$n^+_{rank=r}$	$n^+_{rank>r}$	$n^-_{rank=r}$	$n^-_{rank<r}$	tpr	fpr
1	0.0	0.0	1	9	4	0	$\frac{10}{10}$	$\frac{10}{10}$
2	0.4	0.4	2	7	3	4	$\frac{9}{10}$	$\frac{6}{10}$
3	0.8	0.8	3	4	2	7	$\frac{7}{10}$	$\frac{3}{10}$
4	1.0	1.0	4	0	1	9	$\frac{4}{10}$	$\frac{1}{10}$

$$AUC = \frac{1}{n^+n^-}\sum_{i=1}^{r}(C_i \times B_i + \frac{1}{2}C_i \times A_i) = 0.75$$

presented in Table 2.4. These values are used to compute the AUC and to plot the ROC curve by plotting the true positive rate *tpr* against the false positive rate *fpr*.

At this point, classic ROC analysis can be employed to assess the classification performance for all threshold values of t. In such analysis, the decision of classification is based on comparing the score s_i against t to determine whether s_i falls above or below t indicating a positive or a negative classification. The magnitudes by which s_i scores exceed or fall short of t are excluded from the analysis. The corresponding ROC curve for this example is shown in Figure 2.5.

2.2.4 Confidence in the ROC analysis

In order to compare the performance of learning models in the ROC space, it is necessary to measure the variance of performance. The use of multiple tests sets produces a collection of ROC curves for a given learning model. These ROC curves need to be combined or averaged to make conclusive and significant remarks regarding the model's performance. To this extend, Fawcett (2003) presents several methods of how to average a collection of ROC curves. We briefly review them in this section.

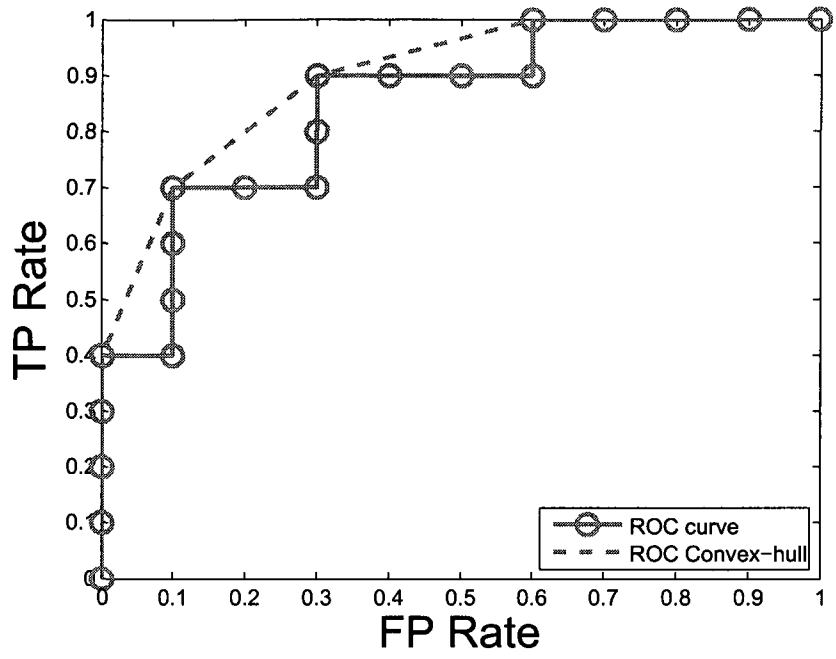

Figure 2.5: The ROC curve for Tables 2.3 and 2.4

The first is method combines results from testing the learning model by cross-validation or bootstrapping. It merges instances associated with each fold according to their assigned scores using the merge sort method. Fawcett (2003) argues that the downside of this method lies in its failure to provide a measure of the variance. Therefore, ROC curves, resulting from testing the model on various samples of the data, need to be averaged by projecting them onto a single dimension so that conventional methods of averaging can be used. This, however, raises questions concerning the appropriateness of the projection method itself (Fawcett, 2003). Projection methods follow the vertical averaging and the threshold averaging schemes presented in (Fawcett, 2003).

Projection-based methods have been used to average ROC curves in (Macskassy et al., 2005b; Macskassy and Provost, 2004) by constructing ROC confidence Bands (Macskassy et al., 2005a). The vertical averaging method examines consecutive fpr values and averages the tpr rates of several ROC curves for each fpr value (Provost et al., 1998). However, a

weakness of this approach lies in the lack of independent control over the model's fpr values (Fawcett, 2003; Macskassy et al., 2005b). Threshold averaging attempts to remedy this weakness by holding constant the threshold values of the learning model rather than its fpr values (Fawcett, 2003). This method of averaging selects a subset of uniformly distributed thresholds from the sorted set of all thresholds used by the ROC curves being averaged. For each such threshold values, the threshold averaging method identifies the set of ROC points that would be generated using that threshold on individual ROC curves. Then, it computes the mean and standard deviations generated for the fpr and tpr values of those points. This produces the mean ROC point as well as vertical and horizontal confidence intervals (Macskassy et al., 2005b; Macskassy and Provost, 2004).

For a more detailed description of averaging ROC curves and constructing ROC confidence bands, we refer to (Fawcett, 2003; Macskassy et al., 2005a,b; Macskassy and Provost, 2004). These methods are not directly related to this research, rather, they relate to the ROC analysis in general.

2.2.5 Other graphs related to the ROC

This section briefly describes alternate performance graphs and plotting spaces related to the ROC space, particularly, Cost Curves (Drummond and Holte, 2006), Coverage Space (Frünkranz and Flach, 2005), DET Graphs (Martin et al., 1997), and Relative Superiority Graphs (Adams and Hand, 1999).

Cost curves are designed to measure the expected cost of classifier performance for the full range of possible class distributions and misclassification costs (Drummond and Holte, 2006). They depict classifier performance given specific misclassification costs and class probabilities. Essentially, cost curves are able visualize classification performance rather than ranking performance. The cost of performance, in this case, is represented in terms of misclassification cost, namely by $C(-|+)$ and $C(+|-)$. If we assume that the cost of misclassifying a positive instance $C(-|+)$ is equal to the cost of misclassifying a negative instance $C(+|-)$, then, the expected cost becomes the error rate of misclassification. An

operating point is defined by $p(+)$, the probability of an instance belonging to the positive class. In this setting, cost curves plot the error rate as a function of $p(+)$. The error rate is shown on the y-axis and $p(+)$ is plotted on the x-axis.

When $x = p(+) = 0$, this means that all instances are classified as negatives and the error rate becomes the error rate on the negative examples. When $x = p(+)$, the same situation occurs for all instances being classified as positives and the error rate, in this case, represents the error rate on the positive instances. If we join these two points with a line, we plot the performance of the classifier as a function of $p(+)$. In general, the y-axis represents the normalized expected cost of misclassification (performance) and the x-axis represents the probability cost function $PCF(+)$ calculated as shown in Equation 2.19.

$$PCF(+) = \frac{p(+)C(-|+)}{p(+)C(-|+) + p(-)C(+|-)} \tag{2.19}$$

where $C(-|+)$ is the cost of misclassifying a positive (the cost of a false negative), $C(+|-)$ is the cost of misclassifying a negative (the cost of a false positive), $p(+)$ is the probability of a positive, and $p(-) = 1 - p(+)$ is the probability of a negative.

Cost curves and ROC curves are closely related by a line/point duality relationship. This means that a point in the ROC space is represented by a line in the cost space and vice-versa (Drummond and Holte, 2006). A point (fpr,tpr) in the ROC space is represented by the line shown by Equation 2.20, and a line in the ROC space whose slope is S with a y-intercept tpr_0 is plotted as a point in the cost space which has the coordinates shown in Equations 2.21 and 2.22.

$$Y = (fnr - fpr) \times p(+) + fpr \tag{2.20}$$

$$X = p(+) = \frac{1}{1+S} \tag{2.21}$$

$$Y = (1 - tpr_0) \times p(+) \tag{2.22}$$

For illustration, consider the sample data listed in Table 2.3 on page 34. The corresponding ROC curve is shown in Figure 2.5 on page 36, and the resulting cost curve is shown in

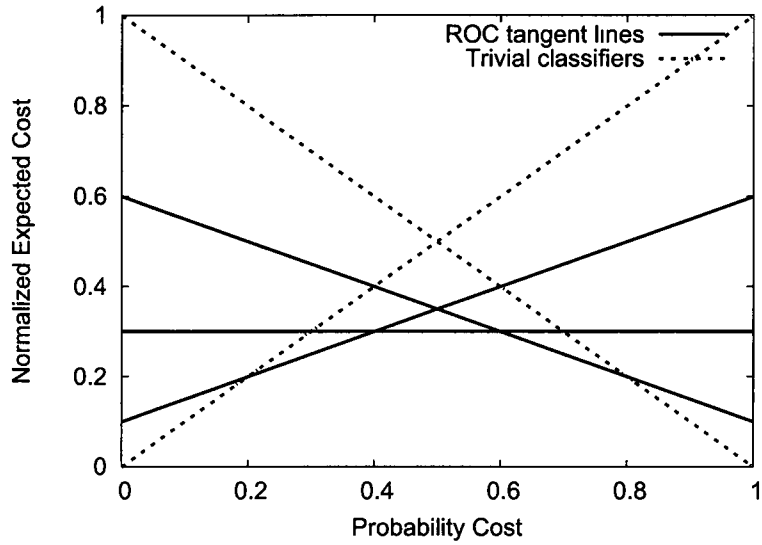

Figure 2.6: The Cost Curve for for Table 2.3

Figure 2.6. Each ROC point in Figure 2.5 produces a line in Figure 2.6. The lower envelope produced by the intersecting lines in Figure 2.6 represent the operating performance of this classifier. Additional details can be found in (Drummond and Holte, 2006).

Coverage Space is a close relative of the ROC space and is described in (Fränkranz and Flach, 2005). They visualize the performance of covering rule learning algorithms in a space variant of the ROC space The idea of coverage space is similar to the ROC space but without normalization to produce the rates. For instance, a point in the ROC space is located by the coordinates fpr and tpr for the X and Y axis respectively. In coverage space, the same point would be located at fp and tp for the X and Y axis respectively. Simply, coverage space is identical to an ROC space but plots absolute numbers rather than normalized rates. The Y-axis shows the number of the positive instances and the x-axis shows the number of the negative instances. When the data has an unequal number of positives to negatives, the coverage space is a rectangular space rather than squared, as it is the case for the ROC space Coverage space can be converted into an ROC space by simply normalizing the coordinates along the x-axis by the number of negatives, and along

the y-axis by the number of the positives.

Another relative graph to the ROC plot is the *DET graph* proposed in (Martin et al., 1997). It represents a complimentary method to the ROC. DET graphs plot the false negative rate instead of the true positive on the y-axis. They represent a plot of both errors against each other, the fnr against the fpr (Fawcett, 2003). In addition, DET graphs are log scaled on both axis such that the area on the bottom left of the graph is expanded and corresponds to the top left corner of the ROC plot. The idea is based on that good performing models tend to be located in that area of the plot. Therefore, a DET graph sets the focus on the area of interest in the context of performance.

The final graph in this review related to the ROC graph is the *Relative Superiority Graph* and the *LC Index* (Adams and Hand, 1999). The LC (Loss Comparison) index is a measure of confidence of superiority. It is used as a transformation of the ROC curve so that the comparison of learning models is based on the cost of learning. Their argument follows that the exact cost of information is rare, but some information of the learning cost is always available (Adams and Hand, 1999). They argue that an expert may be unable to specify the cost of misclassification but may define the cost of one error relative to the other. This method maps the ratio of costs onto the interval $[0, 1]$, then, transforms a set of ROC curves into parallel lines showing which line, representing a learning model, dominates in which segment of the interval. Relative Superiority Graphs can be viewed as a binary version of cost curves (Fawcett, 2003).

2.2.6 Scores and rankings in the ROC analysis

Perhaps the most important aspect of the ROC analysis related to this research is the role of class membership scores, probabilities and rankings in the ROC space. Fawcett (2003) states that the most important feature of the ROC analysis is that it measures the model's ability to assign *good* relative scores to data points. These scores are used to order instances according to the expectation of being positive. These act like rankings, and therefore, this can be paraphrased to say that the ROC analysis measures the model's ability to produce

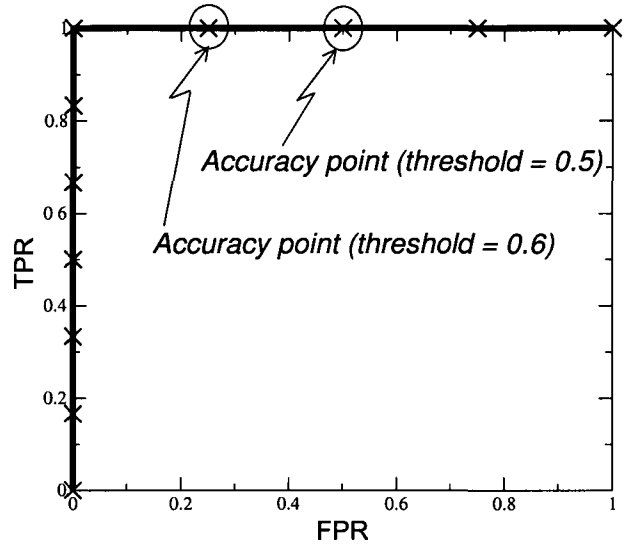

Figure 2.7: Classification in the ROC space (Fawcett, 2003).

a good ranking of the instances. Fawcett (2003) also points out that the scores are not required to be accurate, calibrated probabilities. This is clear from that the ROC analysis eliminates their use once classification decisions are made. The reason for this exclusion is based on that the ROC curve measures the ability to discriminate between the positive and the negative classes.

To illustrate the role of class membership scores in the ROC space, Fawcett (2003) presents Figure 2.7. The ROC curve in this figure shows that for some value of classification threshold (higher than 0.6), this model produces perfect classification. If the class membership scores, it produces, are calibrated probabilities, then, a threshold value of 0.5 makes sense. However, Figure 2.7 shows that threshold value of 0.5 produces higher error than 0.6 which is also less than optimal. By inspection, we can see that 0.5 or 0.6 as the threshold value produces less than optimal accuracy in this case. The optimal performance occurs for a threshold value higher than 0.6. This indicates that if these scores are probability estimates, then they are not properly calibrated. Despite this problem, the ROC curve is able to show an optimal performance due to its ability to measure the distinction

between the two classes, because it aggregates the performance for all threshold values (Wu et al., 2007). An ROC curve contains information about classification decisions and ranking performance; however, the quality of the scores remains absent from the analysis. Despite their absence, when these scores are probability estimates, they may be calibrated using the ROC convex-hull curve or by using other probability calibration methods. These are explained in detail later in this chapter.

Making a discrete (a non-scoring) classifier produce class membership scores (turn it into a scoring classifier) results in various quality of scores. The easiest method to do so is to use multiple classifiers and combine their predictions (by voting) to produce scores. The *MetaCost* system presented in (Domingos, 1999) bags an ensemble of discrete classifiers and combines their votes to produce class membership scores. Alternatively, the confidence of a rule, produced by a rule learner, can be used as a scoring method (Fawcett, 2001). Finally, scores can also be produced by exploiting specific features of the learning algorithm. Fawcett (2003) calls for "*looking inside*" the classifier and using instance statistics as scores. Decision trees are a popular example of this approach where class proportions in the leaves can serve as scores (Provost and Fawcett, 2001). Several learning algorithms have been studied extensively and are found to produces probability estimates. Such are: Probability Estimating Trees (Zadrozny and Elkan, 2001; Margineantu and Dietterich, 2002; Provost and Domingos, 2003; Ferri et al., 2003; Zhang and Su, 2006), Naive Bayesian Classifiers (Zadrozny and Elkan, 2001; Jiang and Zhang, 2006), and lexicographic rankers (Flach and Matsubara, 2007).

2.2.7 Probability estimates and the ROC convex-hull

The method of generating the ROC curve can be made to exploit the monotonicity of classification thresholds, where every instance, classified as positive with respect to a given threshold, remains classified as positive for all thresholds of lower values (Fawcett and Niculescu-Mizil, 2007). This algorithm is described in detail in Section 4.2 of Chapter 4. For each instance, this algorithm incrementally makes a single-step move in the vertical

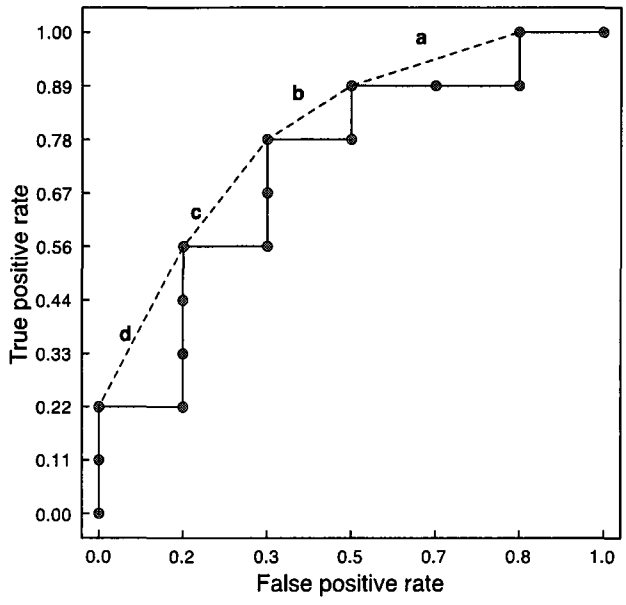

Figure 2.8: ROC convex-hull of a single model (Fawcett and Niculescu-Mizil, 2007).

or in the horizontal directions of the ROC space based on whether the instance is labeled positive or negative, respectively. The step-size of these moves are normalized by $\frac{1}{n^+}$ in the vertical direction and by $\frac{1}{n^-}$ in the horizontal direction to form line segments which make up the ROC curve.

The ROC convex-hull of an ROC curve is introduced in (Provost and Fawcett, 2001) which shows that a classifier is optimal, if and only if, it lies on the convex-hull connecting the ROC points. The ROC convex-hull is a piecewise-linear curve which is always concave-down, and it represents connecting the optimal models in the ROC space. An optimal learning model in the ROC space is the one located closest to the north-west corner of the graph. The slopes of line segments along the hull are monotonically non-increasing with $k-1$ discrete values, where k is the number of component classifiers that make up the hull including the end points of the space. An illustration is shown in Figure 2.8. The dashed curve represents the ROC convex-hull and is made up by joining line segments connecting the optimal ROC points. The solid curve is the ROC curve.

A line segment on the hull represents a tie among instances it covers. For a single classifier, the convex hull can be interpreted as an isotonic transformation of the original model designed to maximize the area under the ROC curve by repairing concavities in the ROC curve (Flach and Wu, 2003, 2005). The slopes of individual line segments are computed by Equation 2.23 where *pos* and *neg* are the numbers of the positives and negatives respectively in the group of instances of equal scores, and $skew = \frac{n^+}{n^-}$, the ratio of positives to negatives (Fawcett and Niculescu-Mizil, 2007).

$$slope = \frac{pos/n^+}{neg/n^-} = \frac{1}{skew} \times \frac{pos}{neg} \tag{2.23}$$

Fawcett and Niculescu-Mizil (2007) shows how the above slopes of individual line segments can be mapped into probability estimates once the class distribution, or skew, is known. Independently, Flach and Matsubara (2007) show that the calibration of probability estimates is linked to the ROC convex hull. Therefore, the slope of line segments of the convex-hull represent calibrated probability estimates for instances they cover. To calibrate the probability scores, produced by the learning model, they recommend constructing the ROC curve with the associated convex-hull, then, using the slopes of line segments as a probability estimates for instances covered by them. Such process is also illustrated in (Fawcett and Niculescu-Mizil, 2007) and is shown to be equivalent to isotonic regression (Zadrozny and Elkan, 2001). Methods of constructing the ROC convex-hull are omitted in this review because of their lack of relevance to this research; however, we provide a brief listing of them for completeness. Initially, Provost and Fawcett (2001) brought forward a method that generates the ROC convex-hull, but recently, Fawcett and Niculescu-Mizil (2007) presented an alternative, more efficient method. They call it the PAV algorithm designed to calibrate probability estimates. The PAV method uses the Pool Adjacent Violators algorithm and is based on an isotonic regression calibration method. They show that the PAV algorithm is equivalent to the method used by Provost and Fawcett (2001). Independent and similar results are presented by (Flach and Wu, 2005), where they improve

the performance of a learning model by rearranging the class membership scores to repair concavities on the ROC curve. They claim; such technique can be used to improve the probability scores due to the isomorphism between the ROC curve and the PAV algorithm (Fawcett and Niculescu-Mizil, 2007).

2.2.7.1 Calibrating probability scores

In decision-making tasks, the value of obtaining calibrated probability estimates relies on making reliable estimates of the true probability that each instance belongs to the class of interest (Cohen and Goldszmidt, 2004; Bennett, 2000). However, some learning models are prone to producing poor probability estimates (Fawcett and Niculescu-Mizil, 2007; Niculescu-Mizil and Caruana, 2005). Extensive studies have been conducted to improve the quality of probability estimation produced by decision trees and by Naive Bayes methods (Zhang and Su, 2006, 2004; Ferri et al., 2003; Margineantu and Dietterich, 2002; Provost and Fawcett, 2001; Zadrozny and Elkan, 2001), and studies in (Fawcett and Niculescu-Mizil, 2007; Flach and Matsubara, 2007; Zadrozny and Elkan, 2002; Provost and Fawcett, 2001; Niculescu-Mizil and Caruana, 2005; Caruana and Niculescu-Mizil, 2006) are focused on calibrating probability estimates.

The assessment of probability scores calibration is based on measuring their Brier score value which is also known as the mean squared error MSE (Brier, 1950). However, this metric provides a scalar summary of the over all assessment. In forecasting models, probabilities are assessed based on calibration and refinement (DeGroot and Fienberg, 1983). In machine learning, the former has become more popular than the latter. In this section we focus on probability calibration and briefly discuss its refinement.

The notion of calibration compares the estimated value of a parameter to its previously known value to determine whether the estimate is an over or under estimate and adjusts accordingly. For instance, if a statement says; *an event occurs with probability of 80%*, then, we can calibrate this estimate by conducting trials while recording observations of how many times the event occurs. If the event occurs only 65% of the time (less than 80%), we can

say that statement is an over estimate, and it can be calibrated down to 65%. However, if the event occurs 90% of the time (more than 80%), then, the statement is an underestimate and can be calibrated up to 90%. The idea is to adjust the probability estimate to better reflect the true probability by measuring its observed value.

Following notation presented in Section 2.1, we define probability calibration as presented in (Fawcett and Niculescu-Mizil, 2007). A scoring classifier is defined as a mapping $\hat{s} : X \rightarrow \mathbb{R}$ which assigns scores $\hat{s}(x_i) \in \mathbb{R}$ instances $x_i \in X$. Calibrating these $\hat{s}(x_i)$ scores produces a mapping m such that $m(s(x_i)) \rightarrow [0, 1]$ is an estimate of the posterior probability of instance x_i being positive (Fawcett and Niculescu-Mizil, 2007). Finding this mapping m is known as calibration. Instead of using a scoring classifier, one can use a probability estimator which, as described in Section 2.1 of this chapter, is a mapping $\hat{p} : X \rightarrow [0, 1]$. However, these probabilities may not be calibrated. To calibrate them, a mapping m is needed to produce $m(\hat{p}(x_i)) \rightarrow [0, 1]$ as calibrated posterior probabilities of belonging to the positive class. Methods of calibrating probabilities can be based on parametric estimation (Platt, 1999), binning (Zadrozny and Elkan, 2001), or non-parametric isotonic regression (Zadrozny and Elkan, 2002). The PAV algorithm (Fawcett and Niculescu-Mizil, 2007) is also among such methods and was discussed in the previous section.

A second property of probability estimates, which can be used to compare them, is refinement. DeGroot and Fienberg (1983) compare probability estimates of two classification models in terms of calibration and refinement. Refinement measures how close the probability estimates are to zero and one. Similar assessment can also be carried out in the context of positive and negative examples. For instance, a scoring classifier (or a probability estimator) is desired to assign scores close to one for the positives and scores close to zero for the negatives. In such a case, the model produces refined scores that clearly distinguish between the positives and the negatives. It can be argued that the AUC under the ROC curve is maximized when scores are refined with a low classification error. It is preferred to construct a more refined classifier; however, refined scores may not produce good probability estimates, and therefore, calibration is emphasized as the primary concern when it

comes to probability estimation. A Naive Bayes model is an example of a refined but poor probability estimator (Zhang, 2004).

2.3 Chapter summary

This chapter presents a review of methods used to evaluating classification, ranking, and class membership score estimation in binary class problems. The performance of classification is measured, naturally, by the accuracy metric (or 1 - error rate), but has been criticized in (Provost et al., 1998; Ling et al., 2003b). The performance of ranking tasks is measured by the ROC curve (Fawcett, 2003; Provost et al., 1997; Provost and Fawcett, 2001) and associated analysis. An ROC curve is a two dimensional representation of performance for all threshold values, and its ability to respond to changing class or cost distributions makes it better suited for the task. Consequently, the area under the ROC curve becomes more appropriate than accuracy when summarizing classification performance. Measuring the quality of class membership scores as probability estimates is usually done by computing the mean squared error, or Brier score (Brier, 1950). The calibration of these probabilities is related to the ROC convex-hull curve (Fawcett and Niculescu-Mizil, 2007; Provost and Fawcett, 2001; Flach and Matsubara, 2007), and several methods can be used to calibrate scores and probability estimates. Finally, many well-established learning methods are able to provide calibrated, good estimates of class membership probabilities (Zadrozny and Elkan, 2001, 2002; Caruana and Niculescu-Mizil, 2006; Niculescu-Mizil and Caruana, 2005; Wu et al., 2007).

Reviews presented in this chapter provide a technical background used to help position the contributions of this dissertation among related evaluation methods. This positioning and related work are discussed in more detail in Chapter 3.

Chapter 3

Positioning

In supervised learning, a problem may be represented by various learning tasks that convey diverse information about class memberships of data points. Classification, ranking, scoring and probability estimation are such tasks commonly used to construct learning models. This chapter presents a comparative framework that organizes these tasks based on information relevant to the performance assessment process. Within this framework, we identify a gap in the performance analysis literature, and we argue that the contribution of this research fills this gap by extending a well-established evaluation method. In this process, the chapter presents a positioning of this research among closely related evaluation methods available in the machine learning literature.

In this research, a learning task refers to the construction of a learning model that solves a corresponding problem in a domain. The distinction between tasks relates to how their associated models express class memberships for individual data points. This may be done by classifications, ranks, scores, or probability estimates. For instance, the example discussed in Chapter 1 deals with the task of deciding whether to play an outdoor game based on weather conditions data. The learning task, corresponding to this problem, may involve classification that produces decisions directly, or it can be based on computing ranks, scores, or probabilities form which decisions can be obtained. Such is the distinction used to categorize learning tasks throughout this chapter.

3.1 Information content of common learning tasks

A classification task relies on a classification model to make decisions that categorize individual data points into appropriate classes. A binary classification task is a special case that makes decisions on two classes usually referred to as the positive and the negative class. In the multi-class settings, there are more classes which are often nominal. However, the classification task remains the same. An ordinal classification task extends the multi-class settings to include an order among classes. Effectively, this order has implications on class memberships of individual points. For instance, consider the classification of asthma patients who are experiencing exacerbations in the emergency department of a hospital. They may be classified in *mild, moderate,* and *severe* classes whose order relates to the severity of the asthma exacerbation. In this case, the order may imply an exclusive patient membership in one class, because concluding a *mild* severity of exacerbations, for example, can hardly suggest the suffering of a severe episode. Alternatively, the classification problem can be presented as: patients who are classified in a severity class higher than *mild* are admitted for observation or treatment, otherwise, they are discharged. Such classification allows for the interpretation of the AUC metric to be meaningful in the context of one class (*mild*) versus the others. However, the outcome of this learning task remains, nonetheless, a classification decision despite the class order.

A ranking task produces an order of the data points from the most positives, placed at high ranks, to the least positives, positioned at low ranks. Intuitively, a good ranker ranks the positives towards the top and places the negatives towards the bottom of the ordered list. Ranking can be depicted by a simple order or by assigning rank values to individual instances. Sorting them by their rank values produces the desired order. Scoring is a mechanism that enables the learning model to express confidence in the class memberships of individual instances. Therefore, they are called class membership scores. When these scores are probabilities, the task becomes a probability estimation task, which is an informed case of a scoring task and involves the modeling of posterior probability distribution of class

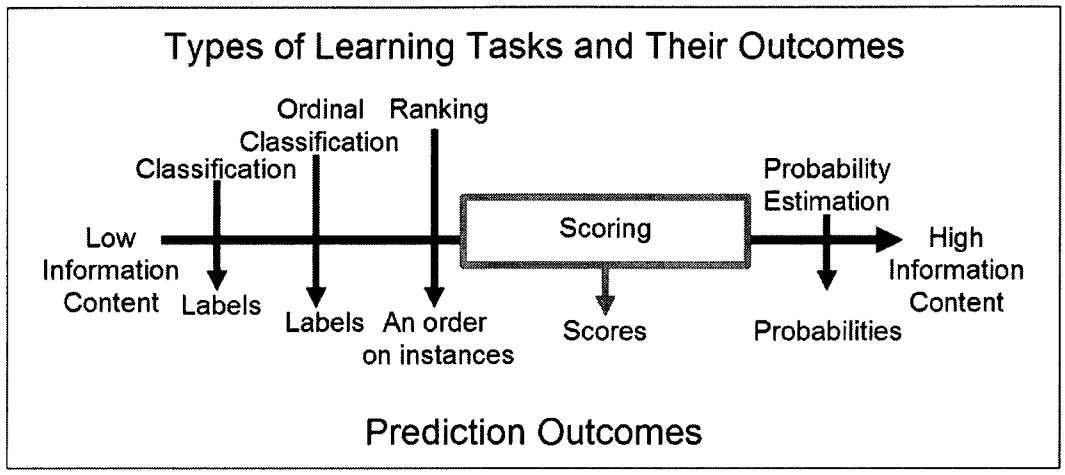

Figure 3.1: Information content of common machine learning tasks.

memberships given the training data. In this case, the scores reflect the probabilities of individual data points belonging to the positive class.

The premise of this thesis relies on utilizing a comparative framework, shown in Figure 3.1, to organize these learning tasks based on their outcomes. The outcome of a learning task conveys information that can be used by the performance evaluation method. The figure shows that the outcome of a classification task is the labels it assigns to instances, and for a ranking task, the outcome is their order. While a scoring task produces class membership scores, a probability estimation task is a more specific kind of a scoring task. It estimates class membership probabilities and assigns them to individual instances.

In this dissertation, we claim that these learning tasks can be organized by the increasing order of their information content as illustrated in Figure 3.1. For instance, the outcome of a classification task is a *yes* or a *no* decision. It indicates whether, or not, an instance belongs to the positive class. With decisions only, distinguishing between instances predicted in the same class is difficult. In a ranking task, the outcome is an order so that more, or less, positive instances can be identified relative to a given instance. Therefore in this case, we are able to compare pairs of instances but fail to detect gaps or compute distances between

them. Alternatively, the learning model may be made to produce ranking values to help determine ranking gaps and compute ranking distances between them. These ranking values can be considered as scores, however, the example presented in Section 1.2.1 of Chapter 1 illustrates that using the ranks to compute distances or to detect gaps between data points can be misleading. The reason for this problem relates to that the ranks show the order of points and ignore the magnitudes of their scores.

Based on the confidence of the learning model in its predictions, the scores describe an order and magnitudes of class memberships in the positive class. On the one hand, a considerable amount of information is needed to interpret the scores in the domain, because the interpretation depends on the method used to estimate them. This is a considerably difficult task more often than not, and to deal with this issue, some learning algorithms estimate probabilities of class memberships, because they can be easily interpreted in the domain using principles of statistics. However, estimating scores on the other hand, is less cumbersome than estimating probabilities. The latter estimate requires substantial knowledge of the domain and may impose requirements or make assumptions in the domain.

For instance, the Naive Bayes method requires that individual attributes be independent, and to be useful, individual probability values of attributes need to be non-zeros (to prevent the majority of class probabilities from being zero). This illustrates that it is necessary to understand that the Naive Bayes algorithm relies on Bayes rule of conditional probabilities. This understanding can indeed be beneficial, because it enables the explanation of predictions based on Baye's formula being used. However, such understanding may not be attainable for other, more complex methods. In addition, such an understanding of the estimation method is necessary for the purpose of performance assessment. This requirement may prevent the black-box treatment of the learning model. Finally, probability estimation methods may impose assumptions in the underlying distribution from which data is drawn. For example, the class distribution may be assumed to remains constant, and the data distribution may be assumed to be static (does not change over time). The violation of such assumptions have been argued in (Widmer and Kubat, 1996, 1998; Forman,

2005; Hand, 2006).

The ability to distinguish between instances depends on the granularity of predictions made by the learning task. Ties, or small differences, between predictions represent similarities between data points as expressed by the learning model. However, large gaps in their scores represent large differences between them and can be of value (Cortes et al., 2007). Using the CT scan patients example presented in Section 2.2.6 of Chapter 1, if the learning model produces similar scores and assigns them to a group of such patients, which are significantly different from scores assigned to other patients. This suggests that this group of patients have similar needs to undergo a CT scan. Such a distinction between patients, when correct, can alert the physician to differences in patients needs for treatment. Ignoring the scores by considering decisions or ranks without the magnitudes of the scores makes the assessment process insensitive to such similarities or differences.

Ling et al. (2003b) describes this property as the "discriminancy" of a measure. While a classification task distinguishes between positive and negative instances, a ranking task separates them based on how high (or how low) they are placed in an ordered list. The gap between them is determined by the availability of instances rather than by the strength of their class memberships. The latter is usually conveyed by their scores. In other words, ranking is an order of data points, thus, the ranking gap between two points is determined by observing other data points ranked and placed between them. If we compare the same two points in isolation, the ranking order can only convey which of the two ranks higher than the other. However, if we consider the magnitudes of their respective scores, it is possible to determine a gap between them (by subtraction) irrespective of the presence of other data. The same argument applies when the scores are probabilities. Probability scores represent a particular type of class membership scores, but they also provide such information. However, estimating probabilities is a more difficult task.

It is obvious that a prediction conveyed by a multi-valued score is more likely to distinguished between instances more often than a binary classification decision (a two-values score). On the one extreme, classification is considered to be the least informative task. It

represents an abstract task that produces yes or no decisions, with the least granularity. On the other extreme, probability estimation represents the most informative task capable of describing the probability distribution, learned from the domain. This additional information content is a result of the ability, of probabilistic methods, to exploit statistical theories and principles while potentially making assumptions or imposing restrictions. Probability estimation methods, we claim, are restrictive and inflexible due to being parametric in nature. They provide powerful tests and are most specialized with a high degree of granularity. One can argue that nonparametric methods can also be used. They require fewer assumptions, but they are also offer reduced statistical power (Motulsky, 1995).

In addition, the selection of parametric or nonparametric test is an informative decision that requires the assessment of data. Motulsky (1995) presents guidelines of how to choose from parametric or nonparametric tests. He points out the hazards of applying a parametric test to data obtained from a non-Gaussian distribution, and argues that applying a nonparametric test to data from a Gaussian distribution results in a loss of statistical power. Furthermore, the decision of which test to select relies on two main assumptions: one, a large amount of data is drawn form the domain, and two, this data drawn randomly. We argue that data collected in real-life domains is more likely to be collected by "convenience" than randomly, thus, the randomness assumption may often be violated. With respect to the size of data drawn form the domain, Motulsky (1995) points out that the size is a difficult issue to determine. Therefore, we argue that the use of statistical tests should not be practiced without the consideration of a magnitude of issues to ensure the applicability of statistical theories. Of course, using statistics offers substantial testing power, however, one needs to assess the data and select an appropriate test to their desired trade off between imposing assumptions and gains in testing power. We claim that such issues present difficulties in real-life domains.

In our framework, probability estimating tasks have the highest information content. As posterior probabilities contain sufficient information for all types of performance measure, if they can be readily estimated, they should be. Our claim, however, is that often this is not

the case. Either sufficient information is not available or, too often, probability estimates are made but are poor as the underlying assumptions are not met. The aim therefore is to use the most informative measure possible given the least assumptions.

3.2 Converting learning tasks

Conversions between tasks, in either direction, can be achieved by abstraction or by specialization, but these operations are not meant to be the inverse of one another. A specialized, more informative task can be abstracted by ignoring some information. In other words, a specialized task can be made to convey the same information as that of a more abstract task because the latter conveys less information. For instance, ignoring assumptions, required for probability estimation, allows for the treatment of probability scores as though they are just scores. Sorting instances in a descending order of their probability estimates, or their scores while ignoring their magnitudes, produces a ranking order (Flach and Matsubara, 2007). Finally, comparing class membership probabilities, scores, or rank values to a threshold results in classifications while ignoring their magnitudes and their order (Fawcett, 2003).

Converting a task in the other direction is rather cumbersome. Specializing more abstract, less informative tasks requires a considerable amount of information usually unavailable. Fawcett (2003) suggests "looking inside" the classification model itself to determine recorded instance statistics. Decision trees, for example, can compute probability score estimates (Provost and Domingos, 2003), and the confidence of a rule can also be used to produce scores (Fawcett, 2001). Finally, an ensemble of discrete classifiers can be combined to produce the frequency of votes as scores (Domingos, 1999). Such methods involve accessing information internal to the learning algorithm, which requires inspecting the structure of the learning model, and potentially, further processing the data.

Intuitively, if sufficient information is available to describe how the scores are estimated, we may be able to consider the scores as probabilities. Nonetheless, it is necessary to

gain an understanding of the learning algorithm structure to examine and, possibly, to verify assumptions it makes. With a lack of such information, existing evaluation methods eliminate these scores and reduce the learning task to a ranking task, to a classification task, or to a combination there of (Fawcett, 2003).

3.3 Positioning within existing literature

The position of this research relies on scoring tasks as part of the comparative framework illustrated in Figure 3.1. We claim that there is a gap in the machine learning literature when it comes to the evaluation of scoring tasks. Our arguments follow two rationales: the first relies on questioning the quality of probabilities (e.g. poor estimates) which can necessitate their assessment as scores, and the second, argues that existing performance metrics including accuracy, AUC and Brier score are designed to summarize the performance, which makes them unable to provide the necessary distinction between individual data points when measuring performance similarities or differences (e.g. gaps in scores or the lack there of). In this case, the ROC performance curve is a useful, well-established evaluation method designed to visualize the ranking performance. Its ability to represent individual data points has been demonstrated by Fawcett and Niculescu-Mizil (2007). However, despite its wealth of information (Flach, 2004), the ROC analysis ignores the magnitudes of the scores (Wu et al., 2007), but it still can provide a useful base for a new valuation method. The scored ROC curve extends the ROC curve to include the score magnitudes, and therefore, it is capable of assessing the performance of a scoring task without reducing it to a ranking.

3.3.1 The quality of probability estimates

In practice, probability estimates can fail to represent probabilities, a case in which most researchers convert the probability estimating task into a ranking to enable the performance assessment by the ROC curve. In Section 3.2, we argue that this conversion results in a reduction in information made available for the process of performance assessment. This

reduction is a consequence of a shortage in current evaluation methods, which are unable to take the score magnitudes into account. A similar argument can be made for the practical reduction of a scoring task to a ranking in general. When evaluating the performance of a scoring task, we advocate that scores be examined for the consideration as probability estimates e.g., by computing their Brier (1950) score. Probability estimates, as suggested by the comparative framework presented in Section 3.1, are the most informative results with respect to performance assessment and their performance can be assessed by the well-established methods of probability theory. If for any reason – discussed in details later in this section – they fail for such a consideration, then instead of reducing the task into a ranking, we propose they be assessed by the method proposed in this thesis. Current evaluation methods can only measure the performance of classification or ranking, hence, most researchers reduce the scoring task to a ranking by ignoring the magnitudes of scores all together.

Issues concerning scores and probability estimation have been documented by several studies in the machine learning literature (Bennett, 2000; Margineantu and Dietterich, 2002; Niculescu-Mizil and Caruana, 2005; Zadrozny and Elkan, 2001, 2002; Provost and Domingos, 2003; Ferri et al., 2003; Grossman and Domingos, 2004; Zhang and Su, 2004; Flach and Matsubara, 2007). Fawcett and Niculescu-Mizil (2007) state that scoring classifiers are commonly used to provide not only class labels, but also scores associated with data points. These scores usually reflect the confidence of the classifier in its predictions, because the higher the score, the higher the probability of the associated instance being positive. In some applications, the scores are used to estimate posterior class membership probabilities (Fawcett and Niculescu-Mizil, 2007). To motivate their calibration, it can be argued that the raw scores do not always provide good estimates of true probabilities, because some models are prone to producing poor probability estimates. This observation is also made in (Niculescu-Mizil and Caruana, 2005).

Probabilities are valuable because they contain sufficient information useful for all types of performance assessments. If they can be readily estimated, they should be. However,

we claim that in practice this is often not the case due to several factors. Either sufficient information needed to estimate probabilities is unavailable, or more often, probability estimates are made but their quality is poor as the underlying assumptions are not met. The quality of probability estimates relies on elements including assumptions made, estimation methods, statistical tests used to measure the performance, and changes in the domain. For example, most statistical methods, parametric and non-parametric, assume that sufficient data is randomly drawn from the domain. Effectively, one needs to examine two issues, or possibly make assumptions: how much data is sufficient, and the collection is random. In practice, data is collected by convenience and potentially in a manner that limits the cost of collection. This can be amplified in medical domains. Medical researchers put great emphasis on data collection which leads to higher quality data of mostly limited size due to the cost of collection (Cios and Moore, 2002). In most domains, the quality and size of data may be compromised by the trade off between convenience and cost (Motulsky, 1995).

The ability to verify assumptions relies not only on the domain but also on the understanding of the method of estimation. For instance, the Naive Bayes method assumes independence between individual attributes, therefore, it is not recommended for text mining (Rennie et al., 2003). Despite its tolerance to violations of this assumption in classification (Domingos and Pazzani, 1997; Zhang, 2004), Naive Bayes has been shown to produce poorly calibrated probability estimates (Bennett, 2000). Also, decision trees were not designed to produce probability estimates, but with the use of Laplace correction, they can (Margineantu and Dietterich, 2002; Provost and Domingos, 2003; Ferri et al., 2003). This shows that when estimating probabilities, one cannot not use a learning method in a black-box approach because an understanding of assumptions is necessary to ensure the probability estimates are reliable. This can be restrictive in some domains. The inability to verify assumptions may result in poor probability estimates at best, and in the worst case, violating assumptions may invalidate the consideration of these estimates as probabilities. Using a statistical test, in such a case, becomes questionable and assessing the quality of scores becomes a desirable solution which existing evaluation measures cannot offer. In

addition to failing probability estimates and in other application domains, it may be unnecessary to treat the scores as calibrated posterior probabilities. Instead, they are used to generate the ROC curve to estimate the expected performance (Fawcett and Niculescu-Mizil, 2007). Given the organization of learning tasks presented in Figure 3.1, we argue that this reduction omits information relevant to the performance analysis by eliminating the scores. We propose that they be evaluated using our scored ROC curve, which can also be used to assess the performance of probabilities as though they are scores.

Another issue that affects the quality of probability estimates deals with changes in the underlying distribution of data in the domain. The idea of constructing a learning model from the training data to make predictions on future, unseen data relies on making the assumption that the data distribution varies little over time. For instance, most machine learning methods assume that the class distribution remains constant over time. However, Forman (2005) argues that this assumption is often violated in real-life domains. Along with Bennett (2000), they argue that changes in the domain cause probability estimates to deteriorate. If changes in the domain are undetected, such a deterioration in the quality of probability estimates can lead to unreliable results when using a statistical test. In the current state, evaluation methods are unable to assess the performance in such cases, and they eliminate valuable information by discarding the scores. Thus, given that probability estimates are affected by variations in the data distribution, we would expect the scores to also be impacted by such variations. Therefore, an evaluation measure sensitive to changes in the score values can be expected to detect such deterioration. This is why we expect our method, the scored ROC method, to be sensitive to changes in the underlying distribution of data in the domain. This problem of learning in changing environments is a well documented problem since Widmer and Kubat (1996, 1998) brought it to the attention of the community.

3.3.2 Existing evaluation measures

According to the comparative framework presented in Section 3.1, the evaluation measure relies on information made available by the learning model. For instance, the accuracy metric measures how often classification decisions are made correctly, and the AUC metric assesses the overall probability of ranking positive data points higher than the negatives. In either case, the performance measure is limited to binary decisions and excludes score magnitudes. For scores, the mean squared error (MSE) – also known as Brier (1950) score – measures the average squared error of estimation. When they are probabilities, Brier score (Brier, 1950) is used to measure the average squared deviation of the predicted probabilities. This measure, like many other scalar evaluation metrics, including accuracy, precision, recall, etc., is designed to produce a summary of performance insensitive to the distribution of class memberships scores on individual data points. Such sensitivity is important when visualizing scores for desired characteristics while identifying associated points. This sensitivity allows for the identification of data points that have common scores values (their scores show little variations) versus those points that differ in score values assigned to them. In Chapter 1, we referred to such score differences as gaps in score values. We also illustrated that they can be valuable in determining performance similarities and differences among data points. The Receiver Operating Characteristic (ROC) method addresses this issue of visualization of individual points (Fawcett, 2003; Fawcett and Niculescu-Mizil, 2007) but excludes the magnitudes of their scores as shown in Section 1.2 of Chapter 1. Therefore, this work starts with an algorithm that incrementally constructs the ROC curve Fawcett and Niculescu-Mizil (2007) and extends it to include the scores so that each line segment, associated with a given data point, is weighted proportional to the score magnitude.

3.3.2.1 The margin-based AUC (or sAUC)

Recently, researchers recognized the need for an evaluation method able to include information conveyed by the scores in the performance analysis. Wu et al. (2007) propose an evaluation metric based on the area under the ROC curve (they call it the sAUC), which is

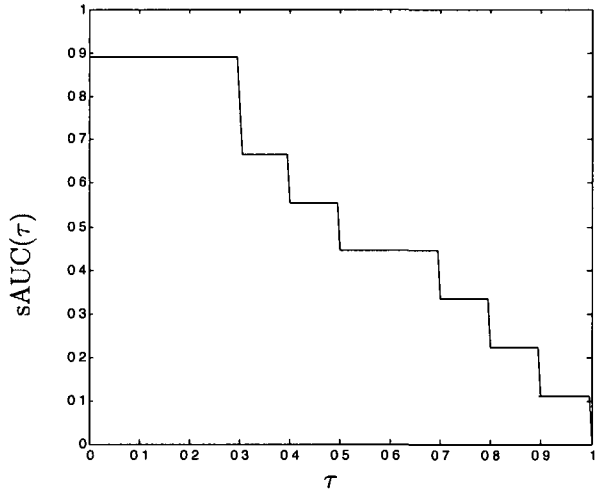

Figure 3.2: Sample AUC deterioration for an increasing τ (Wu et al., 2007).

sensitive to differences in class membership scores. They argue that excluding such differences, by eliminating the scores form the analysis, may lead to over-fitting the validation set when selecting learning models with high AUC values. Wu et al. (2007) demonstrate that the sAUC can be computed without having to construct the associated ROC curve itself, and they interpret this sAUC metric by how quickly the standard AUC deteriorates when the score assigned to the positive data points are decreased.

In more specific terms, Wu et al. (2007) develops a modified version of the AUC which incorporates differences in scores into the calculation of the AUC. The idea is based on modifying the AUC to indicate whether the difference of the pairwise scores is greater than a predetermined margin τ. Their idea is to record the existence of fixed-size score gaps between positive and negatives data points. This margin-based AUC (or sAUC) can be calculated as shown in in Equation 3.1, which accumulates the pairwise difference in scores between pairs of correctly ranked positives and negatives, but only when they (the score differences) are greater than a fixed gap τ (see Equation 3.2). The result is the area under the ROC curve for a given gap value τ. Therefore, plotting $sAUC(\tau)$ against values of τ shows how quickly the AUC values deteriorate as τ increases.

$$sAUC(\tau) = \frac{1}{n^+ n^-} \sum_{i=1}^{n^+} \sum_{j=1}^{n^-} (s(x_i^+) - s(x_j^-))\Psi(\tau) \tag{3.1}$$

where

$$\Psi(\tau) = \begin{cases} 1 & \text{if } (s(x_i^+) - s(x_j^-)) > \tau \\ 0 & otherwise \end{cases} \tag{3.2}$$

Figure 3.2 illustrates that when τ increases the sAUC value decreases. This is so because increasing τ leads to increasing gaps in scores, and thus, fewer such pairs will have their score differences greater than τ. Therefore, the sAUC values will gradually deteriorate (decrease) as τ increases.

Clearly, the sAUC involves information related to the class membership scores that describe the behavior (deterioration) of the standard AUC metric under certain conditions (when the positive scores are decreased). This makes the sAUC relevant but significantly different from our method. This work develops a scored ROC curve weighted by the magnitudes of the scores. Wu et al. (2007) derive a performance metric that summarizes the performance and relies on score differences (or gaps) rather than score values. To properly compare the two methods, the sAUC should only be compared to the area under the scored ROC curve. The former method is presented by Wu et al. (2007) and utilizes fixed-size gaps between all pairs of positive and negative data points. They measure these score gaps only between positive and negative data points, which limits their assessment to the positives versus the negatives. This is depicted by Equations 2.17 and 3.2. In their case, the score difference $(s(x_i^+) - s(x_j^-))$ is accumulated for all pairs of negatives and positives but only when it is greater than τ (see the indicator function $I()$ shown in Equation 3.2.

In contrast, our method enables the visualization of score differences and gaps of any size between any pairs of instances regardless of which class they belong to, i.e., our method can distinguish between data points in one class. The ability of our scored ROC curve to distinguish between any two pairs of instances lies in understanding that the scored ROC curve plots the positive proportion against the negative proportion of the score, as depicted

by comparing the score with the label assigned to the instance (as discussed in Section 4.3.2.1 of Chapter 4). This information, depicted by our scored ROC curve, can be viewed in two ways: the first can be a parallel to the AUC under the standard ROC curve in that it represents the weighted separation between the positives and the negatives, weighted by the magnitudes of the scores, and the second is that this information relates to the separation between the positive and the negative fractions (s_i and $1 - s_i$) of the scores. In each view, the area under the scored ROC curve becomes related to the separation between the cumulative positive scores and the cumulative negative scores. In other words, when we consider the normalization of score magnitudes presented in Section 4.3.2.3 of Chapter 4, we can argue that the scores ROC curve represents the separation between positive and negative scores for all data points. This argument leads to the conclusion that our method reports a different kind of performance information than that of Wu et al. (2007).

Therefore, our method depicts the performance of ranking weighted by the magnitude of confidence in predictions conveyed by the class membership scores. We develop the actual scored ROC curve, and for the purpose of our experiments, we geometrically compute the area under our curve. At this point and due to time constraints, we have not developed an interpretation of the metric based on the area under our scored ROC curve. However, we are able to argue that Wu et al. (2007) develop the sAUC metric as a metric that summarizes a different kind of performance because it is based on score differences (gaps) not values. Their analysis describe the behavior of the AUC under the standard ROC curve, but we develop generalized version of the ROC curve that includes score magnitudes.

3.4 Summary

This chapter presented a comparative framework that organizes the classification, ranking, scoring and probability estimation tasks based on their performance-related information. Within this framework, good probability estimates can be evaluated with the aid of probability theory. However, when the scores fail to quality as probabilities or are poor estimates,

the task of assessing the performance becomes difficult, and existing evaluation methods avoid it by reducing the learning task to a ranking task. Effectively, this reduction eliminates the scores all together, which identifies a gap in existing machine learning research in which scores are over looked. Our research fills in this gap by incorporating the magnitudes of the scores into the ROC analysis. The resulting evaluation method is able to convey the combined performance of classification, ranking and scoring and can also be used to visualize probabilities if need be.

In the process, our method makes two assumptions: the learning problem is a binary class problem, and the value of these class membership scores are between zero and one. The latter assumption ensures that our normalization of the scores maintains the unit square in the scored ROC space. The details of how to construct our scored ROC curve are formally discussed in Chapter 4.

Chapter 4

Scored ROC analysis

This chapter presents the approach of our proposed solution. For ease of explanation, we begin by defining notation, which we use to illustrate our proposed method. Further, this chapter explores several methods that can be used to construct a scored ROC curve, however, we show that only one of them is able to incorporate class membership scores into the ROC space correctly. Subsequent sections present detailed discussions of how we calculate the parameters required by our algorithm to construct the scored ROC curve. Finally, we also demonstrate that a scored ROC curve can be reduced to a standard ROC curve.

4.1 Notations

For a data set X of n instances, let x_i be the i^{th} instance in X whose label is $L_i \in \{+, -\}$. If we let n^+ and n^- be the number of positive and negative instances of X respectively, then, $n = n^+ + n^-$ is the size of the data set. Also, let S_i represent the positive class membership score, assigned by the learning model to the i^{th} instance x_i, where m^+ and m^- are the averages of scores assigned to all of the positive and all of the negative instances respectively.

With this notation, the following sections explore potential methods for incorporating

Algorithm 1 Plotting of the ROC curve incrementally (Fawcett and Niculescu-Mizil, 2007).

Input:
n is the number of instances
n^+ is the number of positive instances
n^- is the number of negative instances
S is a set of n class membership scores sorted in a decreasing order of $S_i \in [0, 1]$
L is the corresponding set of n labels with $L_i \in \{+, -\}$
start at the origin (0,0)
for $i = 1$ **to** n **do**
  if scores S_i are tied **then**
    move along the diagonal
  else
    if $L_i = +$ **then**
      move up by $\frac{1}{n^+}$
    else if $L_i = -$ **then**
      move right by $\frac{1}{n^-}$
    end if
  end if
end for

the scores S_i into the ROC space. We begin with an incremental algorithm to plot an ROC curve, then, we modify it to develop our algorithm.

4.2 Plotting the standard ROC curve

In their early introduction to machine learning, ROC Curves were constructed by plotting the true positive rates against the false positive rates for all classification threshold values between 0 and 1 Fawcett (2003); Provost et al. (1997). However, Algorithm 1 presents an alternative method that plots the ROC curve incrementally Fawcett and Niculescu-Mizil (2007).

Starting at the origin $(0, 0)$ in the ROC space, Algorithm 1 plots the ROC curve by incrementally examining the set of instances X in a decreasing order of their S_i scores. If x_i is a positive instance, the ROC is plotted one step upward, i.e., the algorithm moves one square unit upward along the y-axis. When x_i is labeled as a negative, the ROC curve is plotted one step to the right, i.e. it moves one square unit to the right along the x-axis.

According to this algorithm, the step size is $\frac{1}{n^+}$ in the vertical direction along the y-

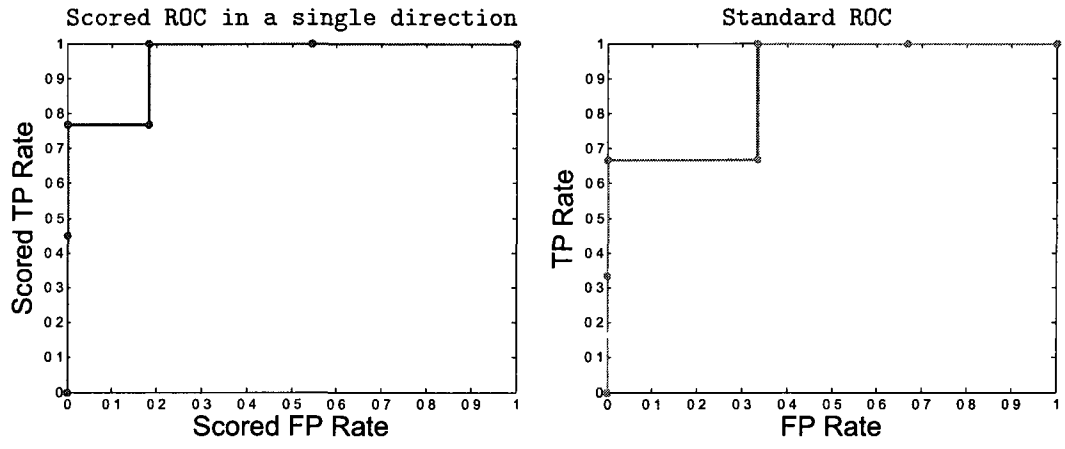

Figure 4.1: The scored ROC in a single direction (left) and the standard ROC curve (right) for Table 2.2 on page 27

axis, and $\frac{1}{n^-}$ in the horizontal direction along the x-axis. It is important to point out that this algorithm plots the ROC curve without relying on a classification threshold value. For illustration, the ROC curve plotted for data shown in Table 2.2, on page 27, appears in the right plot of Figure 4.1.

4.3 Incorporating class membership scores

Our approach, in incorporating the scores into the ROC space, relies on altering the step size in the ROC space, such that, it becomes proportional to score magnitudes. Regardless of what modifications are introduced, we wish to preserve properties and characteristics of the ROC curve, such that, the ROC analysis remains valid by preserving the ranking performance so the curve continues to depict the same performance information.

The following sections examine potential methods for adjusting these step-sizes in two settings. The first approach adjusts the step-size in a single direction along the X or along Y only, and the second modifies it in both directions X and Y simultaneously. We examine this collection of algorithms to investigate their ability to include score values into the ROC curve in an appropriate manner. By appropriate we mean that the curve should depict the correct (positive or negative) portion of score values. This is discussed in further details

Algorithm 2 Plotting a scored ROC curve in a single direction.

Input:
n is the number of instances
n^+ is the number of positive instances
n^- is the number of negative instances
S is a set of n class membership scores sorted in a decreasing order of $S_i \in [0,1]$
L is the corresponding set of n labels with $L_i \in \{+,-\}$
m^+ is the average positive score in S
m^- is the average negative score in S
start at the origin (0,0)
for $i = 1$ **to** n **do**
  if $L_i = +$ **then**
    move up by $\frac{S_i}{m^+n^+}$
  else if $L_i = -$ **then**
    move right by $\frac{1-S_i}{(1-m^-)n^-}$
  end if
end for

in Section 4.3.2.1. Algorithm 2 implements the first approach but is inappropriate due to being biased. It fails to consider how the scores relate to the class label assigned to the associated instances. In a sense, it biases the learning model to produce appropriate scores. Alternatively, we examine the second approach implemented by Algorithm 3 which again fails to include class information. Algorithm 4 addresses this shortfall to construct our scored ROC curve.

4.3.1 Adjusting the step-size in a single direction

We first consider altering the step size, such that, the algorithm moves up proportional to S_i when the instance x_i is positive, or moves right proportional to $(1 - S_i)$ for a negative instance x_i. At each iteration, the algorithm makes a single step either in the upward direction (for a positive instance) or in the horizontal direction (for a negative example). In addition, maintaining the square unit in the space requires the normalization of these steps by $\frac{1}{m^+n^+}$ in the vertical direction, and by $\frac{1}{(1-m^-)n^-}$ in the horizontal direction. Algorithm 2 illustrates this procedure.

This approach can be viewed as a way of sampling the data based on the magnitudes of

their scores. Positive instances are sampled proportionally to the normalized magnitudes of their scores ($\frac{S_i}{m^+n^+}$), which scales the vertical climb of the ROC curve. Similarly, negative instances are also sampled according to their normalized negative scores ($\frac{1-S_i}{m^+n^+}$).

Visually, this approach results in stretching the bottom left and the top right portions of the ROC curve while shrinking the part of the ROC that is towards the top left corner of the space. Such effect can be seen in the left plot of Figure 4.1, and may be interpreted as a bias in favor of the learning model because it does not verify the appropriateness of score assignments (e.g., some positive data points may be assigned low scores and some negatives may be assigned high score values. When unchecked, the scores may be assumed to be assigned appropriately). While positive instances, with high scores (therefore, of high ranks), cause the ROC curve to stretch upward in the bottom left of the ROC space, negative instances, with low scores (of low ranks), cause the ROC curve to stretch in the horizontal direction in the top right of the ROC space. Instances in the middle segment of the ROC curve (instances that form the segment which results in a concavity along the curve in the Figure) are scaled lower because their scores are below the average score.

The resulting curve is optimistic, and thus, mostly dominates the standard ROC curve (it lies higher towards the top left corner of the space). This bias, in favor of the learning model, presents serious issues with the usefulness of this approach, however, it may be used to produce an upper bound on the potential performance of the learning model, and thus, it remains a subject of future investigation. In the next section, this bias is dealt with by adjusting the step-size in both directions simultaneously.

4.3.2 Adjusting the step-size in both directions

An alternative method, to incorporate the scores into the ROC space, can be achieved by adjusting the step-size in both directions, vertically and horizontally, simultaneously. While the move upward can be proportional to S_i, the horizontal step can be scaled proportionally to $(1-S_i)$. In order to maintain the unit square int he space, we must normalize the vertical climb and the horizontal run by $\frac{1}{mn}$ and by $\frac{1}{(1-m)n}$ respectively as illustrated by Algorithm

Algorithm 3 Plotting a scored ROC curve in both directions without class information.

1: **Input:**
   n is the number of instances
   S is a set of n class membership scores where $S_i \in [0, 1]$
   m is the average score in S
2: start at the origin (0,0)
3: **for** $i = 1$ **to** n **do**
4: move up by $\frac{S_i}{mn}$ AND move right by $\frac{1-S_i}{(1-m)n}$
5: **end for**

3.

This approach, however, fails to account for the class information, because positive and negative instances are treated in an identical manner, and the class labels are excluded from the process of plotting the curve. This problem occurs when positive instances are assigned low scores, or when negative instances are assigned high scores. Such scenarios occur when the score estimation process makes mistakes.

In the context of performance analysis, this approach fails to convey an accurate picture of learning performance. When the score assignment process is appropriate, this approach may be useful to determine a form of an upper bound on performance. This approach ignores the class information and makes an implicit assumption that the scores are calculated correctly. Thus, the new curve plotted by Algorithm 3 depicts the magnitude of such scores while ignoring the labels. This depicts the magnitude of the scores only reflecting the maximum learning available. Every S_i score produces a vertical climb and every $(1 - S_i)$ score results in a horizontal shift to the right. Therefore, this curve becomes unable to detect errors in score estimation due to ignoring how score values relate to class information.

To remedy this shortfall, and to include the class information in the analysis, our algorithm should treat correctly classified instances favorably while imposing a performance penalty for incorrectly classified instances. However, we first must understand the meaning of correct classification in the context of using score magnitudes assigned by the model.

To this extent, appropriate score assignments to positive and to negatives instances relies on assigning high scores to the positives and on assigning low scores to the negatives.

Table 4.1: The appropriateness of scores assignments.

Label	High Score	Low Score
+	Appropriate	Inappropriate
−	Inappropriate	Appropriate

In practice, however, this process is not always appropriate and can result in some positive instances being assigned low scores or some negative instances being assigned high scores. Table 4.1 illustrates this situation.

4.3.2.1 The appropriateness of score assignments

A learning task, designed to produce class membership scores, estimates these scores during training, assigns them to instances during testing. Then, it proceeds to making classification decisions by imposing a classification threshold on the assigned scores. Classification errors, false positives and false negatives, occur based on the choice of classification threshold value. While negative instances whose scores lie above the classification threshold result in false positive errors, positive instances with scores below the classification threshold produce false negatives. We argue that such errors can be blamed not only on the choice of classification threshold, but also on the appropriateness of score assignment as per the definition below.

Inappropriate scores are low scores assigned to positive instances, as well as those high scores assigned to negative instances. Distinguishing between high and low scores relies on computing a mid point in the range of these scores. For instance, when the scores are calibrated between 0 and 1, this mid point lies naturally at 0.5. The computation of this mid point is discussed in further detail in a subsequent section of this chapter.

Definition Given an instance x_i, an associated score S_i related to the confidence of x_i being positive, and a mid-point score Mid, then, S_i is called an appropriate score for x_i if $S_i \geq Mid$ when x_i is a positive instance, or if $S_i < Mid$ when x_i is a negative instance. Otherwise, S_i is called an inappropriate score for x_i.

Table 4.2: Classification accuracy and appropriateness of scores.

(Appropriate Scores)			
Actual		Predicted	
Score	Label	Y	N
High	+	Correct	Incorrect
Low	−	Incorrect	Correct

(Inappropriate Scores)			
Actual		Predicted	
Score	Label	Y	N
Low	+	Correct	Incorrect
High	−	Incorrect	Correct

In supervised learning, the designation of performance relies on comparing predictions to a representation of the truth. In classification, truth is represented by class labels and predictions, produced by a scoring model, take on the form of class membership scores. We now discuss how we compare these scores to class labels to assess the accuracy of prediction. Ideally, a scoring task assigns high scores to positive data points and allocates low scores for the negatives. In practice, however, learning models perform far from ideal, and they may err by assigning low scores to some positives or by estimating high scores for some negatives. Definition 4.3.2.1 refers to this by the appropriateness of scores. In order for our evaluation method to designate correct performance information, it must treat appropriate and inappropriate scores differently. While the former are consistent with class information, the latter contradict it, i.e., it is inappropriate for positive data points to be assigned low scores or negative instances to have high scores. In addition, classification decisions are obtained by imposing a decision threshold on the scores values. It is clear that classification errors occur for positives that have scores lower than this threshold, as well as for negatives with scores above it. Such misclassification can occur for either appropriate or inappropriate scores. Therefore, Table 4.2 defines two confusion matrices to address each type of classification error separately. While the first deals with classification accuracy of appropriately assigned scores, the second deals with that of inappropriately scores. With such a distinction between errors, the scored ROC method is sensitive to performance errors resulting from in appropriate scores or from incorrect classification decisions.

In the ROC space, good performance is represented by a vertical climb, and a horizontal run represents a penalty. To incorporate the magnitude of scores into the ROC space,

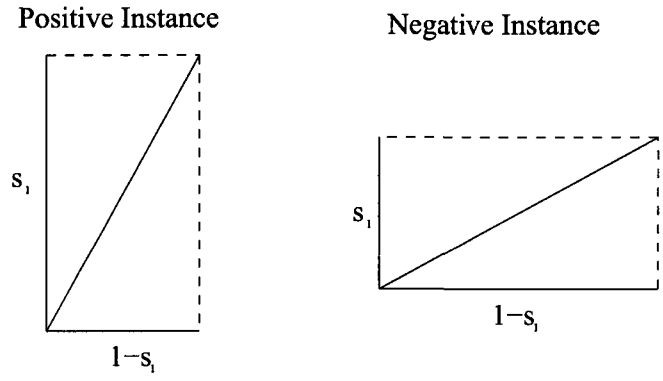

Figure 4.2: Determining the new step size for appropriate scores.

Algorithm 4 makes a distinction between the above two types of score assignment errors by treating instances whose scores are appropriate differently from those whose scores are inappropriate. For data points with appropriate scores, the algorithm climbs vertically by an amount proportional to the magnitude of the score and simultaneously runs horizontally by an amount proportional to one minus the score. This is illustrated in Figure 4.2, and for ease of illustration, we omit the normalization factors at this point. The left plot of Figure 4.2 shows a positive instance with an appropriate score S_i. When $S_i > (1 - S_i)$, the score S_i is higher (more positive) than random. This is equivalent to $S_i > 0.5$, and is depicted by a higher rise than run. When $S_i < (1 - S_i)$, however, the instance x_i is more likely to be negative, and the score can also be considered as appropriate when the L_i label is negative. This is shown in the right plot of Figure 4.2. While the score S_i appropriately indicates, in this case, whether the instance is positive or negative, the $1 - S_i$ score contradicts the labels. Therefore, we plot a vertical climb proportional to S_i to show a gain in performance, and impose a horizontal penalty proportional to $1 - S_i$. The criterion of S_i being greater (or less) than $1 - S_i$ is easiest to illustrate when the scores are calibrated, we can use 0.5 to decide the appropriateness, otherwise, we need to compute the midpoint score. This calculation is discussed in more detail in Section 4.3.2.2 of this chapter.

Earlier, we discussed that data points with inappropriate scores must be treated differ-

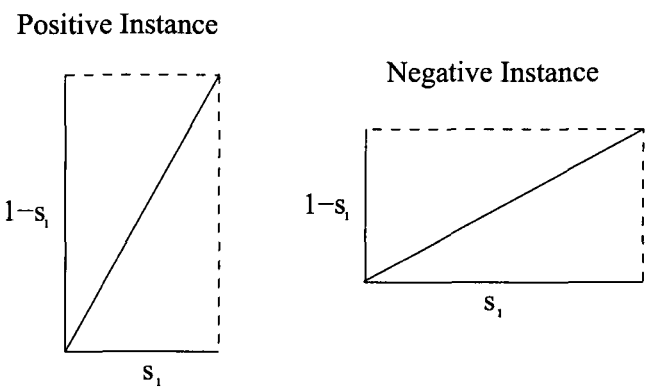

Figure 4.3: The new step size for inappropriate scores.

ently. Such situations occur when some positives are assigned low scores, or when low scores are estimated for some negatives. For these two cases, we reverse the plotting strategy to correct the inappropriateness of scores. As Figure 4.3 shows on the left, a positive instance is assigned a low S_i score where $S_i < (1 - S_i)$. The low score S_i conveys a low likelihood of the instance x_i being positive. In fact, it suggests that the instance is more likely to be negative by contradicting the positive label. The right plot, of the same figure, shows a similar case for a negative instance being assigned a high score value. The high score suggests that it is more likely to be positive. Such contradiction with the label merits some adjustment. In each case, while the score S_i contradicts the label, $1 - S_i$ agrees with the label. Therefore, we apply a performance gain in the form of a vertical climb proportional to $1 - S_i$, and impose a penalty, a horizontal run, proportional to S_i.

Finally, the complete curve is constructed by connecting individual vectors resulting from the successive assessment of individual instances. This strategy follows that used by Algorithm 1 to plot the standard ROC curve. The main difference is the adjustment of the step-size. In the standard ROC algorithm, individual line segments either rise by $\frac{1}{n^+}$ or run by $\frac{1}{n^-}$. In the scored ROC algorithm, progress is made in both directions simultaneously. It climbs vertically by $\frac{S_i}{\alpha_v}$ while simultaneously running horizontally by $\frac{(1-S_i)}{\alpha_h}$ when the scores are appropriate. When they aren't, this algorithm climbs up by $\frac{(1-S_i)}{\alpha_v}$ and runs

Algorithm 4 Plotting the scored ROC curve in both directions with class information.

Input:
n is the number of instances
S is a set of n class membership scores sorted in a decreasing order of $S_i \in [0,1]$
L is the corresponding set of n labels with $L_i \in \{+,-\}$
Mid = mid point of high and low scores (Equation 4.9)
$\propto_v$ = vertical normalization factor (Equation 4.10)
$\propto_h$ = horizontal normalization factor (Equation 4.11)
start at the origin (0,0)
for $i = 1$ **to** n **do**
  if $(L_i = +)\text{AND}(S_i \geq Mid)$ **then**
    Move up $\frac{S_i}{\propto_v}$ and move right $\frac{(1-S_i)}{\propto_h}$
  else if $(L_i = -)\text{AND}(S_i < Mid)$ **then**
    Move up $\frac{S_i}{\propto_v}$ and move right $\frac{(1-S_i)}{\propto_h}$
  else if $(L_i = +)\text{AND}(S_i < Mid)$ **then**
    Move up $\frac{(1-S_i)}{\propto_v}$ and move right by $\frac{S_i}{\propto_h}$
  else if $(L_i = -)\text{AND}(S_i \geq Mid)$ **then**
    Move up $\frac{(1-S_i)}{\propto_v}$ and move right by $\frac{S_i}{\propto_h}$
  end if
end for

simultaneously by $\frac{S_i}{\propto_h}$. Such curves are illustrated in Figure 4.4. The quantities $\propto_v$ and $\propto_h$ represent the vertical and the horizontal normalization factors respectively. These are used to ensure that progress is made in the unit square measurement of the space. Their calculations are discussed in more detail in a subsequent Section 4.3.2.3.

4.3.2.2 Determining the mid-point score

To decide whether or not a score S_i is assigned to instance x_i appropriately, it is necessary to establish a mid-point value above which scores are deemed to be high. Scores below this mid-point value are judged to be low. The appropriateness of score assignments is decided by consulting the class label so that positive instances are assigned high scores, and negative instances are assigned low scores. Inappropriate score assignments occur when negative instances are assigned high scores, or when positive instances are assigned low scores. The key issue here is calculating this mid-point score. When the scores $S_i \in [0,1]$

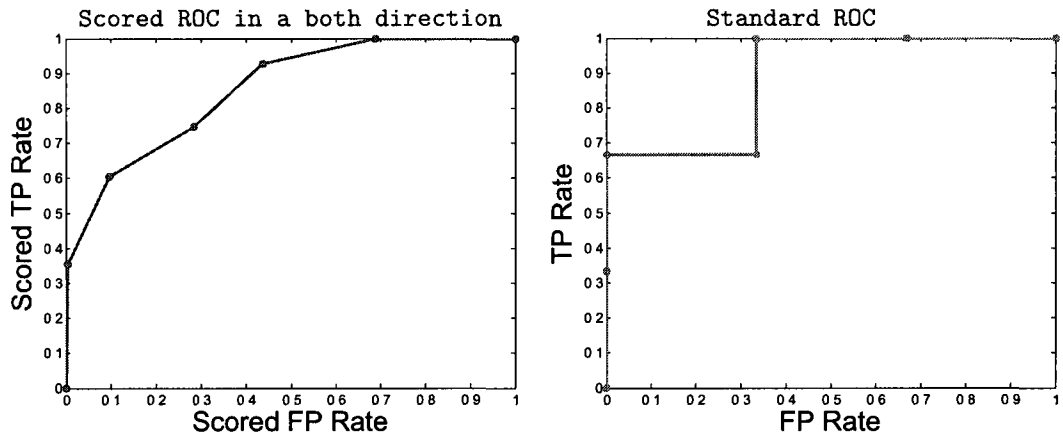

Figure 4.4: The scored ROC curve in both directions with class information (left), and the standard ROC curve (right) for Table 2.2 on page 27

are calibrated, Equation 4.1 can be used.

$$Mid = \frac{max(S) - min(S)}{2} = \frac{(1-0)}{2} = 0.5 \tag{4.1}$$

However, when the scores are not calibrated, one can compute the mid-point, for a given X, by calculating the mid point between the two means, the mean positive score $mean(S_i^+)$, and the mean negative scores $mean(S_i^-)$. Such calculations are justified for a balanced class distribution where $c = \frac{n^+}{n^-} = 1$. Although, the sum of these two means is influenced by the class distribution. We propose to set Mid to the midpoint between the two means weighted by the class distribution c. Thus, if we let m^+ be the mean of S^+, the scores assigned to the positive instances, and if we let m^- represent the mean of S^-, the scores associated with the negative instances, then, the two means can be computed by Equations 4.2 and 4.3 respectively.

$$m^+ = \frac{\sum_{i=1}^{n^+} S_i^+}{n^+} \tag{4.2}$$

$$m^- = \frac{\sum_{i=1}^{n^-} S_i^-}{n^-} \tag{4.3}$$

If the scores S_i are calibrated, and if the class distribution is $c = \frac{n^+}{n^-}$, then, it can be shown that equality 4.4 is true.

$$m^+ + \frac{m^-}{c} = 1 \tag{4.4}$$

The above becomes obvious in the extreme case where $S_i \in \{0, 1\}$ and $c = 1$, which gives Equation 4.5.

$$n^+ \times m^+ + n^- \times m^- = n^+ \tag{4.5}$$

In the general case, however, the S_i scores are not calibrated, and the class distribution is not balanced. Then, we have:

$$m^+ + \frac{m^-}{c} = \frac{\sum_{i=1}^{n^+} S_i^+}{n^+} + \frac{1}{c} \times \frac{\sum_{i=1}^{n^-} S_i^-}{n^-} \tag{4.6}$$

For simplicity, we use:

$$S^+ = \sum_{i=1}^{n^+} S_i^+ \ , \ S^- = \sum_{i=1}^{n^-} S_i^-, \text{ and } S = S^+ + S^- \tag{4.7}$$

Therefore,

$$\begin{aligned} m^+ + \frac{m^-}{c} &= \frac{S^+}{n^+} + \frac{n^-}{n^+} \times \frac{S^-}{n^-} \\ &= \frac{S^+}{n^+} + \frac{S^-}{n^+} = \frac{S^+ + S^-}{n^+} \\ &= \frac{S}{n^+} \end{aligned} \tag{4.8}$$

As a summary, we use Equation 4.9 to set the mid-point Mid.

$$\boxed{Mid = \frac{1}{2}(m^+ + \frac{m^-}{c}) = \frac{S}{2n^+}} \tag{4.9}$$

This method of estimating the mid-point score Mid is data driven because it relies on scores assigned (by the learning model) to the particular data points. We claim that this approach is reasonable because it utilizes information conveyed by the prediction model for the given

data set. However, it can be shown that under some conditions, this Mid point can be calculated outside the interval $[0, 1]$. This poses a limitation to our method, and therefore, alternate methods for estimating Mid remain under investigation for further analysis. This limitation is discussed in Chapter 6.

4.3.2.3 Vertical and horizontal normalization

To ensure that the vertical climb, and the horizontal run rates comply with the unit square of the space, the step-size must be normalized. These are represented by $\propto_v$ and $\propto_h$ respectively in Algorithm 4. In this section, we present calculations of these two normalization factors. First, we need to define some notation.

Let H^+, L^-, L^+, and H^- represent the sets of positive instances where $S_i \geq Mid$, negative instances with $S_i < Mid$, negative instances of $S_i \geq Mid$, and positive instances where $S_i < Mid$ respectively. Their cardinalities are represented by $|H^+|$, $|L^-|$, $|L^+|$, and by $|H^-|$ also respectively. Using this notation, the equations below illustrate how we compute $\propto_v$ and $\propto_h$. Algorithm 4 plots the curve in steps proportional to S_i or $1 - S_i$ scores. We divide each step by the total score contributions in either direction to normalize them. For example, to determine the vertical normalization factor $\propto_v$, we divide by the sum of scores contributing to the upward progress. Lines 7 and 9, of Algorithm 4, show that instances in H^+, and in L^- respectively, contribute their S_i scores towards a vertical climb upward. In addition, Lines 11 and 13, of the same Algorithm, show that instances in L^+, and in H^- respectively, contribute their $1 - S_i$ scores also in the vertical direction. Therefore, the total sum of scores contributions in the vertical direction is calculated by Equation 4.10.

$$\propto_v = \sum_{i=1}^{|H^+|} S_i + \sum_{i=1}^{|L^-|} S_i + \sum_{i=1}^{|L^+|} (1 - S_i) + \sum_{i=1}^{|H^-|} (1 - S_i) \tag{4.10}$$

Thus, it is clear to see that the vertical normalization factor consists of the sum of all S_i, and $(1 - S_i)$ scores that result in a vertical climb. We construct a similar equation for the horizontal direction.

Algorithm 5 Reducing a scored ROC curve to a standard ROC curve

1: **Input:**
   n is the number of instances
   S is a set of n class membership scores sorted in a decreasing order of $S_i \in [0, 1]$
   L is the corresponding set of n labels with $L_i \in \{+, -\}$
2: $T = \frac{max(S)+min(S)}{2}$
3: **for** $i = 1$ **to** n **do**
4: **if** $S_i > T$ **then**
5: $S_i = 1$
6: **else**
7: $S_i = 0$
8: **end if**
9: **end for**
10: Call Algorithm 4 with L, S, and n to plot the scored ROC curve.

For the horizontal normalization factor $\propto_h$, we calculate the sum of all scores contributed toward shifts in the horizontal direction to the right. These can be found on Lines 8, 10, 12, and 14 of Algorithm 4. Instances in H^+, and in L^-, respectively on Lines 7 and 9 of Algorithm 4, contribute their $(1 - S_i)$ scores along the horizontal direction. In addition, instances in L^+, and in H^-, respectively on Lines 11 and 13 of Algorithm 4, contribute their S_i scores also to the horizontal progression. Therefore, the horizontal normalization factor $\propto_h$ is computed by Equation 4.11.

$$\propto_h = \sum_{i=1}^{|H^+|}(1 - S_i) + \sum_{i=1}^{|L^-|}(1 - S_i) + \sum_{i=1}^{|L^+|} S_i + \sum_{i=1}^{|H^-|} S_i \tag{4.11}$$

4.4 The relation to the standard ROC Curves

An important aspect of the scored ROC curve is the ability to reduce it to a standard ROC curve. Such a reduction demonstrates that the scored ROC curve extends the standard ROC curve by, merely, including the class membership scores as weights applied to individual line segments. An extension that preserves valuable properties of the ROC curve, then combines the performance of classification, ranking, and scoring into a single evaluation method. In this section, we illustrate how to perform such a reduction.

Our procedure of reduction relies on ignoring the class membership scores produced by the learning model. The process is illustrated by Algorithm 5, it begins by replacing the scores by ones and zeros. To do so, it must determine a threshold to decide which scores are replaced by ones and which are replaced by zeros. Because the standard ROC curve eliminates the score values form its analysis, the value of this threshold becomes irrelevant. The key is to preserve the order of instances not the scores. Therefore, Algorithm 5 computes the average score on Line 2. Indeed, we can use 0.5 to make such decisions without changing their order.

Having replaced the scores with zeros and ones, Algorithm 5 calls Algorithm 4 on Line 10. The latter plots the scored ROC curve, which we show, is the standard ROC curve. The proof that Algorithm 4 plots, in this case, the standard ROC curve relies on demonstrating that they both behave identically. We begin with the initial calculations of Algorithm 4, which compute the mid-point Mid, the vertical normalization factor $\propto_v$ and the horizontal normalization factor $\propto_h$. The mid-point Mid is calculated on Line 2, of Algorithm 4, by Equation 4.9 as:

$$Mid = \frac{S}{2n^+} \tag{4.12}$$

When the scores are zeros and ones, S is equal to the number of instances whose scores are equal to one. Recall, S is the sum of all scores. These instances may be positives or negatives, however, they all have a score of one, i.e., they all lie above the Mid value, their scores are considered high, and therefore, form the sets H^+ and H^-. Thus, Equation 4.12 becomes:

$$Mid = \frac{|H^+| + |H^-|}{2n^+} \tag{4.13}$$

The condition that $Mid \in [0, 1]$ remains important for the scored ROC method to work. This is a limitation discussed in Chapter 6. Further, Algorithm 4 calculates the vertical normalization factor on Line 3 by:

$$\propto_v = \sum_{i=1}^{|H^+|} S_i + \sum_{i=1}^{|L^-|} S_i + \sum_{i=1}^{|L^+|} (1 - S_i) + \sum_{i=1}^{|H^-|} (1 - S_i) \tag{4.14}$$

When the scores are only zeros and ones, only H^+ and H^- contain instances whose scores are high, and are equal to one. The remaining sets, L^+ and L^-, both have low scores of zero. As a result, we have the following four quantities:

$$\sum_{i=1}^{|H^+|} S_i = \sum_{i=1}^{|H^+|} 1 = |H^+| \tag{4.15}$$

$$\sum_{i=1}^{|L^-|} S_i = \sum_{i=1}^{|L^-|} 0 = 0 \tag{4.16}$$

$$\sum_{i=1}^{|L^+|} (1 - S_i) = \sum_{i=1}^{|L^+|} (1 - 0) = |L^+| \tag{4.17}$$

$$\sum_{i=1}^{|H^-|} (1 - S_i) = \sum_{i=1}^{|H^-|} (1 - 1) = 0 \tag{4.18}$$

Substituting them in Equation 4.14 produces:

$$\propto_v = |H^+| + |L^+| = n^+ \tag{4.19}$$

The above shows that the vertical normalization factor $\propto_v$ in Algorithm 4 becomes equal to the number of positive instances n^+. This is also equal to the vertical normalization factor used in Algorithm 1 on page 65, which plots the standard ROC curve. A similar argument can be used for the calculation of the horizontal normalization factor $\propto_h$. which appears on Line 4 of Algorithm 4. The computation of $\propto_h$ follows Equation 4.11 as:

$$\propto_h = \sum_{i=1}^{|H^+|} (1 - S_i) + \sum_{i=1}^{|L^-|} (1 - S_i) + \sum_{i=1}^{|L^+|} S_i + \sum_{i=1}^{|H^-|} S_i \tag{4.20}$$

Again, replacing the scores with zeros and ones results in H^+ and in H^- to have instances whose scores are one. All remaining instances have scores of zero. Therefore, we compute

the following four quantities:

$$\sum_{i=1}^{|H^+|}(1 - S_i) = \sum_{i=1}^{|H^+|}(1 - 1) = 0 \tag{4.21}$$

$$\sum_{i=1}^{|L^-|}(1 - S_i) = \sum_{i=1}^{|L^-|}(1 - 0) = |L^-| \tag{4.22}$$

$$\sum_{i=1}^{|L^+|} S_i = \sum_{i=1}^{|L^+|} 0 = 0 \tag{4.23}$$

$$\sum_{i=1}^{|H^-|} S_i = \sum_{i=1}^{|H^-|} 1 = |H^-| \tag{4.24}$$

By substituting these quantities into Equation 4.20, we obtain $\propto_h$ as:

$$\propto_h = |L^-| + |H^-| = n^- \tag{4.25}$$

The horizontal normalization factor $\propto_h = n^-$, in Algorithm 4, becomes equal to that calculated by Algorithm 1. We have shown that when the scores are replaced by ones and zeros, our algorithm and the standard ROC algorithm calculate the same vertical and the same horizontal normalization factors. The remaining issue, in demonstrating that the two algorithms behave identically when the scores are zeros and ones, is to show that they both make identical steps, in size and in direction, when they plot their respective curves. To do so, we consider all possible label L_i and score S_i combinations that may occur. There are four such combinations, they include[1]:

1. A positive instance where $S_i = 1$. For this instance, Algorithm 4 computes the vertical step size by $\frac{S_i}{\propto_v}$ on Line 8. This vertical climb reduces to $\frac{1}{n^+}$ because $S_i = 1$ and $\propto_v = n^+$. Simultaneously, the horizontal run is calculated by $\frac{(1-S_i)}{\propto_h}$ on the same line, and becomes $\frac{0}{\propto_h} = 0$ because $(1 - S_i) = (1 - 1) = 0$. Therefore, Algorithm 4 makes a vertical climb of $\frac{1}{n^+}$ only. This is identical to what Algorithm 1 does with a

[1]It is assumed that $Mid \in (0, 1)$

positive instance on Line 8 on page 65.

2. A negative instance with $S_i = 0$, for which Algorithm 4 calculates the vertical climb by $\frac{S_i}{\propto_v}$, on Line 10. This vertical step size becomes $\frac{0}{\propto_v} = 0$ because $S_i = 0$. The horizontal run is calculated by $\frac{(1-S_i)}{\propto_h}$ on Line 10. It becomes $\frac{1}{n^-}$ because $(1 - S_i) = 1$, and $\propto_h = n^-$. Therefore, for this negative instance, Algorithm 3 makes a horizontal run of $\frac{1}{n^-}$ which is identical to what Algorithm 1 does for a negative instance on Line 10.

3. A positive instance with $S_i = 0$ causes Algorithm 4 to calculate a vertical climb equal to $\frac{(1-S_i)}{\propto_v}$ on Line 12. In this case, this is equal to $\frac{1}{n^+}$ because $(1 - S_i) = 1$ and $\propto_v = n^+$. At the same time, the horizontal step is determined by $\frac{S_i}{\propto_h}$ on Line 12, which becomes $\frac{0}{\propto_h} = 0$ because $S_i = 0$. Therefore, Algorithm 3 makes a vertical step of $\frac{1}{n^+}$ only. This is an identical step to that of Algorithm 1 when dealing with a positive instance.

4. The final case is that of a negative instance with $S_i = 1$. Algorithm 4 computes the vertical step size by $\frac{(1-S_i)}{\propto_v}$ on Line 14, which becomes $\frac{0}{\propto_v} = 0$. The horizontal step size is computed by $\frac{S_i}{\propto_h}$ on Line 14, and is equal to $\frac{1}{n^-}$ because $(1 - S_i) = 1$ and $\propto_h = n^-$. The resulting step size is a horizontal shift to the right by $\frac{1}{n^-}$. Such a move is identical to how Algorithm 1 treats a negative instance on Line 10.

All four combinations of binary labels and binary scores are processed identically by both algorithms (both algorithms make identical steps in size and in direction, and they both compute the same normalization factors). Therefore, we can draw a final conclusion: they both behave in an identical manner. When the scores are binary, the two algorithms plot the same curve. Thus, converting the scores into binary values then constructing the scored ROC curve reduces our curve to the standard ROC curve. This identical behavior supports that the scored ROC curve incorporates class membership scores into the ROC curve, because converting the scores to binary values results in the same curves being plotted by both algorithms. This shows that the scored ROC method extends to the standard ROC

curve to incorporate the scores. This is achieved by weighting individual ROC line segments by the scores assigned to their associates data points.

4.5 The area under the scored ROC curve

Section 2.2.3 of Chapter 2 shows how the AUC metric, the area under the ROC curve, can be calculated using an indicator function. This function indicates whether a pair of positive and negative instances are correctly ranked. The idea is presented in Bradley (1997); Hanley and McNeil (1982) and is based on the correct pairwise rankings of the instances. The AUC metric is calculated as a summation of the unit squares contributed by every correctly ranked pair of positive and negative instances.

In the same section, we reviewed a modified version of the AUC metric which calculates the scored area under the ROC curve. This method is presented in Wu et al. (2007) and is based on modifying the indicator function to reflect the difference in the scores of a pair of positive and negative instances being greater than a given margin or fixed score gaps, a parameter τ. The indicator function is true when $S_i - S_j > \tau$ when x_i is a positive instance and x_j is a negative instance. The interpretation of their scored AUC is that it measures how quickly the AUC deteriorates if the positive scores are decreased.

It is important to note that the work in Wu et al. (2007) is based on calculating the scored AUC without constructing the scored ROC curve explicitly. In this thesis, we propose a method to construct this scored ROC curve for which the normalization factors are reasonably complex. Formal derivations of the area under the curve can be obtained using this indicator function approach. Although we defer the analysis of the AUC under our scored ROC to future research due to time constraints, we argued that our AUC depicts a different kind of performance than the AUC presented by Wu et al. (2007). Our AUC is related to the separation of the positive and the negative scores, where as, their sAUC described the behavior of the AUC under the standard ROC under certain conditions (when the positive scores are decreased).

For the purpose of our experiments, we utilize is an alternative method to calculate the area under the scored ROC curve. The AUC can be calculated by successively adding the area of trapezoids under points on the curve Fawcett (2003). We perform such calculations on the scored ROC curve as well.

4.6 Chapter summary

Based on an incremental algorithm used to plot the standard ROC curve, this chapter presents a formal method to incorporate the scores into the ROC curve. Our method relies on modifying the step size, made by the above algorithm, to construct our curve. This chapter shows three possible methods for modifying this step size, they include: modifying the step size in a single direction, in two directions simultaneously without accounting for class information, and finally, modifying the step size in both directions simultaneously while including the class information. The latter, we show, results in the development of the scored ROC curve. We have presented an algorithm to construct this curve and have described how its vertical and horizontal normalization factors are computed. The result is the weighting of the line segments of the ROC curve by the correct magnitude of the scores.

Finally, this chapter present a proof and describes an algorithm for reducing the scored ROC curve into the standard ROC curve. Combined with the notion of modifying the step size, this algorithm demonstrates that the scored ROC curve is an extension of the standard ROC method. In the following chapter, we describe our experiments and presents discussions of their results.

Chapter 5

Experiments

The principle contribution of this research relies on analyzing and interpreting performance information conveyed by class membership scores assigned to instances by a learning model. The score assignment process is a result of learning class membership in the domain. This thesis develops a method to evaluate the performance of such models based on the scores they produce. This chapter presents experimental results to demonstrate the effectiveness of the performance analysis method developed in this dissertation.

The hypothesis being tested in our experiments is that the scored ROC curve measures performance similarities and differences of scoring classifiers, and it does so more effectively than the standard ROC curve. Furthermore, this new method is sensitive to changes in performance due to changes in the domain. Therefore, the design of the experiments comprises two components. The first aims to demonstrate the new method's ability to measure performance similarities and differences for various learning models on a collection of benchmark data sets. The second component focuses on its sensitivity to changes in data between training and testing. For the latter purpose, we use a synthetic data generator to artificially introduce such changes in the domain so that score differences can be reasonably expected. Controlled changes allow the evaluation method the opportunity to detect them.

This chapter describes these two experimental components. For each component, we discuss the experimental design, the data included in the experiments, the learning mod-

els being used, and the evaluation approach of the results followed by presentations and discussions of the results. Finally, the chapter concludes with a summary.

5.1 Component I: measuring similarities and differences

In this research, the ability to assess score similarities relies on their relevance to class memberships of individual instances. This relationship is fundamental to incorporating their magnitudes into the evaluation measure. In our experiments, some models are expected to produce similar performance while anticipating others to show differences. Subsequently, our evaluation method is applied to their predictions in the form of class membership scores to assess their similarities or differences.

The rationale behind this experimental design follows that judging two sets of scores for similarities is a difficult task for which supporting information is rarely available. This difficulty becomes amplified when the class is described by a binary label rather than a score, because it requires comparing a score to a label. Such issues relate to the information content and its granularity discussed in Chapter 1.

Detecting similarities has several applications beneficial to many tasks. On an abstract level, examining scores can help reveal characteristics of learning in a domain. While score similarities describe the model's scoring tendencies, a lack there of indicates the presence of alterations, drifts, or shifts of various kinds. On a practical level, representing user preferences as scores and analyzing them for similarities can be useful for the development of search engines, information retrieval systems, and recommendation systems.

5.1.1 Design

Our experiments rely on the consistency of the learning method to simulate score similarities. This simulation is based on fixing the learning algorithm and on fixing the distribution from which the training and testing data are drawn. It is reasonable to expect that a learning algorithm should consistently produce similar class membership scores when trained on

Algorithm 6 Generating score similarities with randomized fold splitting.

1: **Input:**
D is a set of n instances.
C is a classification algorithm.
XV is a set of 10 random seeds.
2: **Output:**
S is a set of 10 arrays where S_i is a set of n scores produced by applying C on D and $S_{ij} \in [0, 1]$.
3: **for** $i = 1$ **to** 10 **do**
4: $D' = 10$ random folds of D using seed XV_i
5: $S_i =$ class membership scores of applying C to D' by cross-validation.
6: **end for**

Algorithm 7 Generating score similarities for ensemble methods.

1: **Input:**
D is a set of n instances.
C is an ensemble-based classification algorithm C.
E is a set of 5 random seeds.
2: **Output:**
S is a set of 5 arrays where S_i is a set of n scores produced by applying C on D and $S_{ij} \in [0, 1]$.
3: **for** $i = 1$ **to** 5 **do**
4: $D' = 10$-fold splits of D using random seed E_i.
5: $S_i =$ class membership scores of applying C on D' by cross validation.
6: **end for**

sufficient data randomly drawn from the same domain. Therefore, we measure similarities by evaluating scores resulting from multiple runs of 10-fold cross-validation. As shown in Algorithm 6, each cross-validation run uses a different random seed to split the data into ten folds. The idea is to construct, more-or-less, similar classifiers from randomized versions of the same data. For example, a collection of ten Naive Bayes classifiers, resulting from ten runs of cross-validation, should produce similar class membership scores. The same argument applies to building ten Probability Estimating Trees, or PETs, in the same way. We want to show that classifiers in one group, constructed by the same algorithm, produce similar scores among their ten runs which also differ from those produced by classifiers in the other groups. Thus, scores produced by the ten Naive Bayes classifiers should be similar to each other, but they are expected to differ from those produced by the ten PET models.

In the above settings, score similarities arise from comparing the performance of learning models resulting from repeatedly applying one learning algorithm to various samples of the same data. However, it can be argued that the scores, which we compare, are assigned to different data points due to the random splitting into folds. We now show that this is not the case due to the very nature of cross-validation testing. Consider how the cross-validation method works. It begins by randomly splitting the data into N equal folds. For a fold i, the learning algorithm is trained on the remaining folds after setting fold i aside. Then, the resulting model is tested on this fold i. This process repeats for every fold i generating testing results on all N folds. Therefore, in a single run of cross-validation, every instance in the data set is assigned a class membership score by one of the various classification models constructed in one of the N iterations. For this purpose, we compare scores between ten complete runs of cross-validation so that at the end of each run, every instance in the data set is assigned a score. None-the-less, their ranking order may differ because various classifiers produce different score distributions for a given set of data. We are interested in the over all performance despite differences in their ranking order. With the assumption that their training data is similar, these random variations of the similar learning models should produce similar scores.

In the above strategy, variations between classifiers occur due to splitting the training data. However, variations can also result from the method used to construct the learning model itself. For example, a learning algorithm which utilizes an ensemble of random trees can generate slightly different trees from identical data, and can be expected to add variations to their performance. In our experiments, we wish to evaluate scores in the presence of both types of variations, and therefore, we introduce variations in the learning algorithm itself. For this purpose, we perform other experimental runs using two additional learning algorithms for which similarities are produced by building various ensembles of classifiers. We construct various bags of decision trees and various random forests from the same data. Effectively, we construct various versions of classification models trained and tested on the same data. The strategy of these runs is illustrated in Algorithm 7.

To construct a bag of decision trees, the algorithm selects a random subset of instances with which a decision tree is trained. This process repeats until the specified number of decision trees for the bag is reached. In our settings, we construct ten trees in each bag. We expect that these various bags, trained with the same data, produce similar scores among the bags. The same expectation applies to a collection of random forests when the data is fixed, and the learning algorithm is varied.

The main difference between these two ensemble algorithms lies in how they construct the individual decision trees. In the case of random forests, a random variable is used to randomly select features and instances to train individual trees in the ensemble which are finally combined to form a random forest. For the bagged trees, the seed is used for the random selection of instances to build the individual decision trees. Because random forest models bootstrap the data and the features, as opposed to only the data for the bag of trees, we expect the former to show higher degree of variations in performance. Our experiments compare the performance of these two ensemble-based schemes to measure similarities within the same learning scheme and differences between the two schemes.

5.1.2 The UCI data

This component of our experiments involves comparing scores obtained by applying our learning models on a collection of benchmark data listed in Table 5.1. All these data sets represent binary classification problems obtained from the UCI Machine Learning Repository (Asuncion and Newman, 2007).

This thesis develops a method to analyze class membership scores for a binary class problem. For multi-class problems, our method can still be used to measure the performance on one class (the positive class) versus the remaining classes. However, we suspect such grouping of classes may introduce unnecessary noise in the scoring method from the point of view of performance assessment. Therefore, we select binary class data sets from the UCI repository. The `Balloon` data was discarded due to its small size of only 20 records. Table 5.1 shows the details of the selected data sets whose sizes ranges from 100 to over

Table 5.1: UCI binary classification data.

Abbr.	Name	n	$\|+\|$	$\|-\|$	Features	%+
prom	promoters	106	53	53	57	50
echo	echocardiogram	132	43	88	7	33
hepa	hepatitis	155	32	123	19	21
prks	parkinsons	195	147	48	22	75
hart	statlog heart	270	120	150	13	44
hrth	heart disease hungarian	294	188	106	13	64
hors	horse-colic reduced	296	188	108	21	64
habr	haberman	306	81	225	3	26
iono	ionosphere	351	225	126	34	64
vots	house-votes-84	435	168	267	16	39
jcrx	japanese crx	690	307	383	15	44
aust	statlog australian	690	307	383	14	44
wisc	breast cancer wisc	699	241	458	9	34
blod	blood transfusions	748	178	570	4	24
diab	pima-indians-diabetes	768	268	500	8	35
mamo	mammographic masses	945	434	511	5	46
tic	tic-tac-toe	958	626	332	9	65
ger	statlog german	1000	700	300	20	70
oz8h	ozone eighthr	2534	160	2374	72	6
oz1h	ozone onehr	2536	73	2463	72	3
chss	chess kr-vs-kp	3196	1669	1527	36	52
ads	internet ad	3279	459	2820	1558	14
spam	spambase	4601	1813	2788	57	39
mush	mushroom	8124	3916	4208	23	48
mgic	magic04	19020	12332	6688	10	65
adlt	census adult	32562	7841	24720	14	24

30000 instances. Most of these data sets have fewer than 50 features with the exception of the **ads** data set which contains over 1500 features. The last column of Table 5.1 shows the percentage of positive instances found in each set whose value ranges from 3% to 75%.

5.1.3 The learning models

Our experiments focus on understanding the learning performance based on class membership scores. The ROC curve and the area under the ROC curve deal with performance issues associated with ranking one class relative to another. The assessment of scores and their

magnitudes has been limited to probability estimation tasks. In this context, the primary debate that has captured the attention of researchers revolves around comparing probability estimates produced by commonly used models (Flach and Matsubara, 2007; Niculescu-Mizil and Caruana, 2005; Ferri et al., 2003; Provost and Domingos, 2003; Margineantu and Dietterich, 2002; Zadrozny and Elkan, 2002, 2001). The most common is comparing scores of Naive Bayes to those produced by probability estimating trees (PET). These studies show that probability scores produced by these two methods differ greatly from each other. Therefore, we include them in our experiments to give rise to differences between scores. Our objective is to illustrate how our new evaluation method accounts for such differences. In the following sections, we present a brief description for each classification algorithm used in the experiments.

5.1.3.1 Naive Bayes (NB)

The Naive Bayes classification method is a probabilistic classification algorithm based on Bayes rule of conditional probability to estimates the posterior probabilities of class membership, given the training data points. Historically, the Naive Bayes classifier has been found to be very effective in making classification decisions (Domingos and Pazzani, 1997; Zhang, 2004), notwithstanding violations of its assumption of independence among the features. However, the assessment of posterior probabilities, produced by Naive Bayes method, suggests that they are poor and are notoriously uncalibrated (Bennett, 2000).

5.1.3.2 Probability Estimating Trees (PET)

The decision tree algorithm produces a tree as series of tests on individual attribute values to represent a decision boundary. A decision tree is constructed in two phases; a growth phase and a pruning phase. In the growth phase, an initial tree is built then reduced to a simpler tree to reduce the estimated error rate. This reduction occurs in the pruning phase. Pruning removes tree nodes resulting from noise in the data and helps counter over-fitting to produces more accurate models (Quinlan, 1993; Mitchell, 1997; Witten and Frank, 2005).

Generally, a decision tree classifier is designed to maximize decision accuracy. Class membership probabilities are usually obtained by taking the ratio of positives to negatives in the leaf node where the instance is classified. Studies show that the standard decision trees (using pruning without smoothing) produce poor probability estimates (Margineantu and Dietterich, 2002; Provost and Domingos, 2003) because they are designed to reduce classification error. Improving classification decisions produces poor probability estimates Wu et al. (2007); Provost and Domingos (2003). Consequently, using standard decision trees to produce scores is useless. Instead, studies recommend using using unpruned decision trees with Laplace correction (Margineantu and Dietterich, 2002; Provost and Domingos, 2003; Ferri et al., 2003). Smoothing probability estimates produced by decision trees has been shown to be a reasonable approach (Smyth et al., 1995; Bradley, 1997). Therefore, we use the PET method to compare scores it produces against those produced by the Naive Bayes classifier described above. Our expectations rely on these two algorithm to produce similar scores for various seeds of cross validation individually (when compared to themselves), and to produce very different scores when compared to each other. Our experiments will confirm conclusions made in existing studies that PET method produces different scores than the Naive Bayes algorithm (Margineantu and Dietterich, 2002).

5.1.3.3 Bagged PET (BPET)

Several machine learning studies show that using an ensemble of models improves the classification performance over the use of a single model (Provost and Domingos, 2003). Constructing an ensemble of models involves training several models and then combining them into a single model by voting or by averaging their scores. Bagging is such an algorithm, which stands for *Bootstrap Aggregation*, and is commonly used in conjunction with decision trees. It builds an ensemble of classifiers whose individual members are constructed on randomly selected subsets of instances using the bootstrap method (Breiman, 1996). Provost and Domingos (2003) show that BPET improves the performance over using a single PET. They argue that this improvement is due to averaging the probability scores across members

in the ensemble (Provost et al., 1998; Bauer and Kohavi, 1999). Our analysis compares the performance of several BPET models, where each bag contains ten non-pruned decision trees with Laplace correction. The objective is to generate random variations among the bags using random training data samples. Our experiment will show that our evaluation method captures similarities among these various ensembles (bags).

5.1.3.4 Random Forest (RF)

Introduced also by Breiman (2001), random forests represent a variation on the bagging algorithm. The random forest method uses random decision trees as the main model for building individual members in the ensemble. Each such tree is constructed from randomly selected subsets of instances and features. For our experiments, the random forest models (RF) are expected to be similar to the bagged PET models (BPET). However, there is a difference among them. Variations among members of the ensemble are due to selecting random subsets of features, as well as, random subsets of instances. Only the latter is used by the BPET model. Therefore, we include these two schemes in our experiments. Our experiment will measure performance similarities among the various random forests using the scored ROC method developed in this research. More importantly, our evaluation method will show that scores (class membership scores) produced by these random forests are similar to those resulting form BPET models described above. This similarity is also evident in (Caruana and Niculescu-Mizil, 2006).

5.1.4 Evaluation

To assess how well our evaluation method, the scored ROC method, measures performance similarities or differences among learning models, we compare its conclusions to those drawn from the standard ROC analysis. The selected learning algorithms are trained and tested on data sets included in the experiment. To support the analysis, we plot the scored and the standard ROC curves, and we compute the area under them respectively. Due to the use of several data sets, the number of possible plots becomes large. Therefore, we only show plots

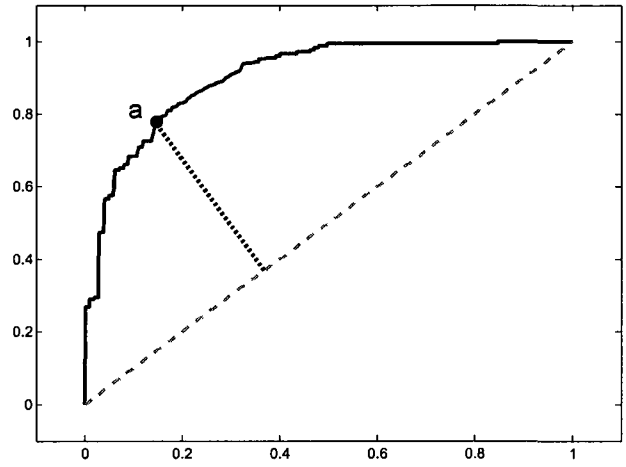

Figure 5.1: Pairwise separation between points along two curves in the ROC space.

representative of the results. In addition, it is necessary to examine curve dominance, or the lack there of, to conclude similarities or differences between curves in the scored and in the standard ROC plots. Therefore, we summarize our plots using two scalar metrics which together depict these results. First, we compute the average and the standard deviation of the area under the curves in each space for each group of random cross-validation runs. When a curve produces higher area under it is more likely to dominate the other. However, the area under the curve alone is insufficient to show dominance due to the possibility of overlapping and crossing over the other multiple times. Therefore, we need a second metric to measure the separation between curves to show dominance by a higher area under the curve coupled with a higher degree of separation. Curve separation near 100% suggests dominance. To this extent, we develop a second scalar metric to estimate this separation between curves in the following section.

5.1.4.1 Measuring curve separation

When evaluating machine learning methods in the ROC space, the task of determining curve dominance is challenging and cumbersome. In this section we develop an adequate metric to measure the separation between two ROC curves. Accompanied with the AUC metric,

this separation score enables us to express dominance of one curve over another. The idea is based on comparing the pairwise-distances of points on both curves to the increasing diagonal. Figure 5.1 shows such distances for a pair of points on two ROC curves. In this space, an good learning model results in a convex curve that lies northwest of the increasing diagonal line (the random decision line). When comparing two curves, they both begin at the origin (0,0) with no separation. They also end at point (1,1) with no separation. With respect to performance, we are interested in the region between the curves and above the increasing diagonal line. The emphasis is on how they are separated. Such is the region between the dark and the light curves in Figure 5.1, which can be formally measured by the Gini index (Gastwirth, 1972).

In the experiments, we ensure that both curves consist of the same number of points. Using simple trigonometry, we calculate and compare the pairwise-distances shown by the dotted lines in Figure 5.1. Such pairwise comparisons produce a test that indicates whether a point a lies above or below point b. To compute the separation score for the dark curve's dominance over the light one, we count how many points on the dark curve lie above points on the light curve. Then, we normalize this count as a percentage of the total number of points on the curve itself while excluding the end points. In Figure 5.1 and with the exception of the end points, all point on the dark curve are separated from points that lie on the light curve.

In addition to measuring the separation in one direction, we also calculate its value for the inverse dominance. We compute this separation score for the light curve dominance over the dark one as well. In Figure 5.1, this inverse separation is 0% because the light curve lies completely below the dark curve. However, it is possible that curves may frequently cross over each other, or they may overlap along some segments of their curves. Measuring this separation score in both ways depicts the size and direction of how the two curves are separated, and when considered alongside the value of the area under the curve, they collectively identify the curve and estimate the magnitude of its dominance.

Referring back to the example in Figure 5.1, the separation between the dark and the

light curves is 100% while the inverse is 0%. This represents separation in favor of the dark curve. In addition, their AUC values are 78.64% and 63.49% respectively. If we consider both metrics, we see that a 100% separation score accompanied with a higher AUC value suggests that the dark curve dominates the light curve. This example shows a situation where the curve is dominant (100% separation). However, performance curves may differ without being completely separated. Therefore, we relax the dominance criteria to require at least 80% curve separation and more than 2% difference in the AUC. This criteria of curve dominance is ad hoc and is only used to highlight certain results for ease of presentation. However, we list all results for the reader to draw their conclusions.

For the experimental results, we tabulate the average AUC values and average separation scores calculated for curves in the scored and in the standard ROC spaces. For a given model, our tables show the average AUC value computed for a collection of scored ROC curves. These curves represent the performance obtained from ten runs of 10-fold cross-validation. The same average AUC is calculated for each model in the ROC space as well. The discussion compares these two averages on our data set. We also compute separation scores for every pair of curves associated with a given pair of models in a particular run. These resulting separation scores are averaged over the ten runs of 10-fold cross-validation.

5.1.5 Results

This section examines relationships between scores and their corresponding classification decisions. First, we discuss the score-based performance of individual learning models over various runs of cross-validation. The objective is to show that the scored ROC method captures similarities among scores generated by the same model on variations of data obtained from the same domain. Subsequently, we consider the pairwise performance of different models. We compare Naive Bayes (NB) scores to PET scores followed by BPET scores to those of the random forest (RF) models. These pairwise comparisons show that the scored ROC method is sensitive to differences between different models. Collectively, these results demonstrated that our method detects performance similarities among variations of similar

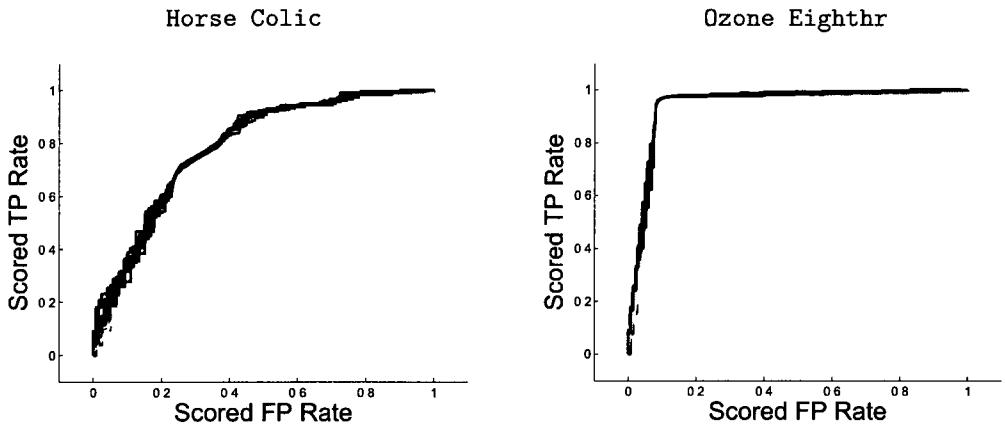

Figure 5.2: Scored ROC curves for NB (dark) and PET (light).

models, as well as, differences between different models.

For ease of presentation, selected results are plotted as performance curves for models applied to data sets. In the discussions, comprehensive tables show summaries of results for all data sets listed in Table 5.1.

5.1.5.1 Similarities of variations of the same model

This section presents results to demonstrate that the scored ROC method detects similarities among scores produced by the same learning model in a given domain. To this purpose, we consider the standard deviation of the area under the curve. In the next section, we examine differences between learning methods by examining their dominance as measured by the AUC and by our curve separation metric. At this point, we only examine curves generated by the same learning method for multiple runs of random 10-fold cross-validation.

We begin with the scored ROC curves generated for models applied to the `Horse Colic` and the `Ozone Eighthr` data shown in Figure 5.2. The dark bands show the scored ROC curves resulting from applying ten runs of the Naive Bayes method tested by 10-fold cross validation. The lighter bands represent the scored ROC curves resulting from executing the same experiment using the PET learning method.

At a first glance, it is clear that curves of the same color form a tight distinct band.

This means that scores generated by the same model are similar to each other over multiple runs of cross-validation. This is illustrated by the dark and by the light sets of curves which form tight (with low variance), consistent bands respectively. The dark band shows that scores produced by the Naive Bayes models are similar to each other. A similar conclusion also applies to the light curves generated by PET scores. This suggests that the random splitting of data into folds has little influence on how the two models compute their scores. Their scoring mechanisms behave consistently and produce scores with little variations.

While ignoring the issue of dominance for the moment, comparing the respective curves to those plotted in the standard ROC space shown in Figure 5.3 reveals an obvious increase in the variance. The bands become wider and are more erratic than for scored ROC. This indicates that, although their classification decisions are similar (the curves trail each other), the random splitting variations have influenced the smoothness of the ROC curves resulting in higher variance. This unsurprising higher variance in the ROC space is due to the exclusion of the scores from the analysis. Small changes in the score magnitudes impact classification decisions and cause the ROC to appear more ragged. In the scored ROC curve, the slope of a line-segment in the normalized unit-square is proportional to the magnitude of the score assigned to the instance. Therefore, small changes to the scores result in small changes to these slopes on the curve. Hence, the scored ROC curves appear smoother and tighter. This variance can also be measured by calculating the standard deviation in the area under the curves in either space.

The results of similarities between various curves generated by the same learning method are consistent for all data sets. To illustrate this comprehensively, we compare the standard deviations of the average AUC in both spaces. Similar curves are expected to be close to each other resulting in small deviations in the average AUC. For our selected data sets listed in Table 5.1, there are 52 possible plots to compare. We generate a single plot for each space (scored and standard ROC spaces) for 26 data sets. The results are summarized by averaging the AUC of the ten curves resulting from 10 cross-validation runs, and we compute their standard deviation for individual models on each set. The results for the

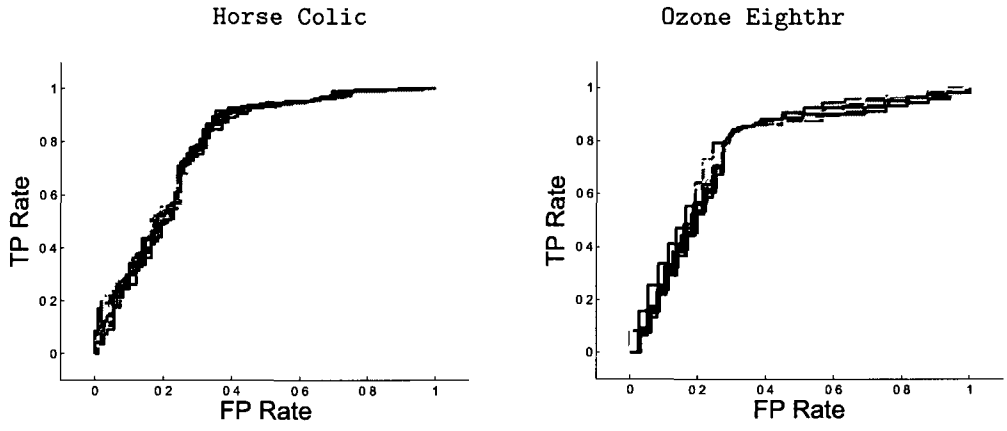

Figure 5.3: Standard ROC curves for NB (dark) and PET (light).

Naive Bayes and the PET models are shown in Table 5.2.

In this table, the first column shows the abbreviated name of the domain, the following two columns show the average and the standard deviation of the AUC as percentages for the PET models in both spaces respectively. The last two columns of the same table show the same values for generated by the Naive Bayes models in both spaces. In Table 5.2, the bolded values represent the lowest standard deviations. For instance, the standard deviation in AUC values for PET on the `horse colic` (*hors*) data is 0.52% of the scored ROC space, and it is 1.39% of the standard ROC space. The former is the lowest of the two. On the same row, this AUC standard deviation for Naive Bayes is 0.45% of the scored ROC space, and it is 0.44% of the standard ROC space, the latter of which is the lowest.

Inspecting values in Table 5.2 reveals two observations. First, the standard deviation values are relatively small compared to their respective average AUC values. This means that the individual AUC values, that are being averaged, are similar because the standard deviations are small. This suggests that both scored and standard ROC methods measure the similarities between curves with relative ease. An intuitive question is; which of the two methods, the scored or the standard ROC, measures these similarities better? The second observation provides an answer to this question. The bolded values in Table 5.2 show that these deviations are often smaller in the scored ROC space than in the standard

Table 5.2: Average AUC ± standard deviation for PET and NB over ten runs of cross-validation on UCI data. The lowest deviations are in bold.

	PET		NB	
Data	Scored ROC	Standard ROC	Scored ROC	Standard ROC
prom	**1.55**		**0.48**	
echo	**4.12**		**0.32**	
hepa	**2.31**		**0.77**	
prks	**2.23**		**2.00**	
hart	**1.37**			**0.16**
hrth	**0.86**		**0.22**	
hors	**0.52**			**0.44**
habr	**1.21**		**0.96**	
iono	**1.35**		**0.90**	
vots	**0.23**		**0.44**	**0.44**
jcrx		**0.52**		**0.36**
aust		**0.47**	**0.25**	
wisc		**0.18**	**0.32**	**0.32**
blod	**0.67**		**0.36**	
diab	**0.95**		**0.17**	
mamo	**0.32**		**0.08**	
tic	**0.69**		**0.13**	
ger	**0.45**		**0.19**	
oz8h	**0.80**		**0.27**	
oz1h	**0.58**		**0.11**	
chss	**0.01**	**0.01**	**0.03**	**0.03**
ads		**0.31**	**0.47**	**0.47**
spam	**0.19**	**0.19**	**1.30**	**1.30**
mush	**0.00**	**0.00**	**0.04**	**0.04**
mgic		**0.11**	**0.08**	
adlt	**0.06**		**0.08**	**0.08**

ROC space. This shows that similarities are measured better by the scored ROC curve than by the standard ROC because the former results in lower AUC variance. Lower AUC variance suggests similar AUC values, hence, similarities among curves.

Considering the above results, we can conclude that the scored ROC curve captures similarities among scores produced by the same model reasonably well. Further, it measures them better than the standard ROC does. In more general terms, measuring model similar-

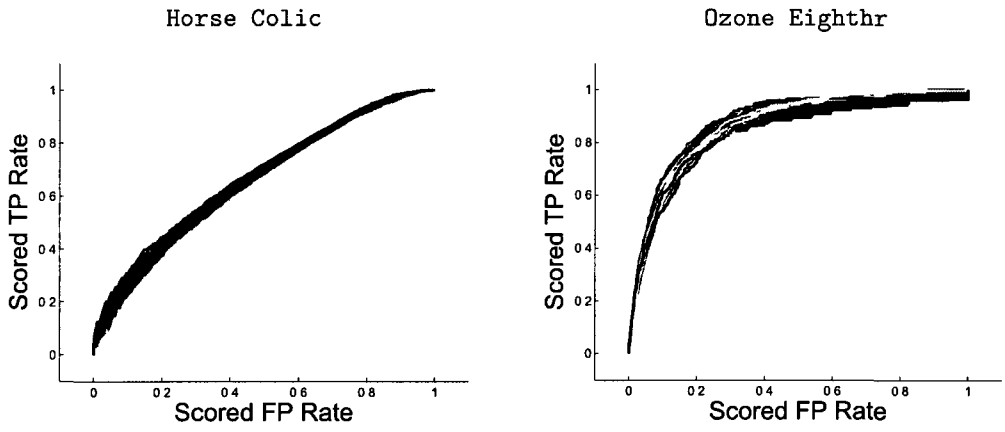

Figure 5.4: Scored ROC curves for RF (dark) and BPET (light).

ities by comparing the magnitudes of its class membership scores is better than comparing the classification decisions based on them.

Now, we shift our focus to the other two learning models; the BPET and the RF models. For each of these two ensemble methods, we construct five variations of it, then, we evaluate them using ten runs of 10-fold cross-validation. Effectively, we introduce variations in two ways. The first lies in the learning algorithm itself by randomly constructing multiple trees and forests, and the second relies on randomly splitting the data into folds for cross-validation. Therefore, we expect to observe higher variations that should result in higher standard deviations in the AUC for both types of curves.

Figure 5.4 shows scored ROC curves obtained for these two models on the two selected data. In this case, both models produce reasonably good scores because the dark and the light curves are consistent, convex, and smooth. In addition, there is little difference between them. The dark bands closely follow the light bands, a situation different from the previous results. In contrast, curves in Figure 5.5 represent the standard ROC curves obtained for the same runs. They form bands that are noticeably thicker. This suggests that variations introduced in this part of the experiment have an obvious impact on the standard ROC curves in particular. In addition, the BPET model results in a slightly better ranking performance than the Rf models because the light curves, for the most part, dominate the

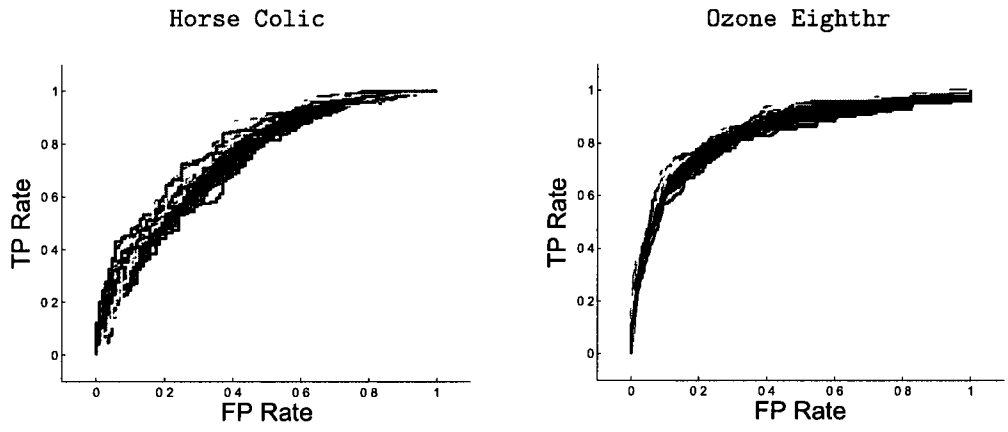

Figure 5.5: Standard ROC curves for RF (dark) and BPET (light).

dark ones. However, the previous Figure 5.4 shows that these two learning method produce similar scores. Given that the increased random variations impacts the deviations of the ROC curve, and given that scores produced by these two models are found similar, this slight dominance of BPET over the RF models in the ROC space becomes questionable.

Values listed in Table 5.3 illustrate the impact of the increased random variations for these two ensemble methods on the AUC in both spaces. The organization of this table is similar to the previous one with the exception of using different learning models. Again, bolded entries show the lowest standard deviations in AUC values. Despite the added variations, our previous observations remain. The AUC deviations obtained in the scored ROC space for both models are, in most cases, less than or equal to those computed in the standard ROC. This supports our conclusion that random variations have greater impact on classification decisions than on class membership scores. The scored ROC curve is more sensitive to similarities between models than the standard ROC curve. The latter eliminates these scores. Therefore, the scored ROC curve is the logical choice when assessing the scores.

It is important to note that the similarity of performance between RF models and BPET models appears consistent in both spaces. Their respective average AUC values in either space are consistently close (within 5%). To see this effect, compare entries in the second column to those in the third column, and compare those in the third column against those

in the fifth column of Table 5.3. These results suggest that BPET and RF models compute similar scores and make similar classification decisions. This issue will be revisited again in the next section when we compare their dominance in both spaces.

Having illustrated the ability of the scored ROC method to represent similarities among scores produced by variations of the same learning model, we proceed to investigate score similarities among different models. The following section reviews the same results in the context of performance dominance of one model over another.

5.1.5.2 Differences between two models

In this section, we compare the performance of different learning models based on assessing the dominance of their performance curves. While their scored ROC curves rely on the scores they produce, their standard ROC curves use the classification decisions they make.

A curve is said to dominate another when all its points are optimized higher (dominance) or the same (weak dominance) than those of the inferior curve. The task of defining dominance is difficult in nature and must be given a criteria to enable its measurement. To this effect, the definition of dominance described in Section 5.1.4.1 on page 94 is followed in the presentation of results. First, we examine the dominance of PET and Naive Bayes models then proceed to discuss that of BPET and Random Forests.

The issue of dominance is referred to earlier in Figure 5.2 on page 97 and the lack thereof in Figure 5.3 on page 99. While dominance of score curves produced by the Naive Bayes model (dark band) can be observed in the former figure. there appears little to no dominance in the latter figure. Consider the scored ROC curve of the Naive Bayes model (dark band) on the `Ozone Eighthr` data shown in Figure 5.2. Starting at the origin, this scored ROC curve, climbs steeply and reaches its peak along the y-axis where it changes directions and proceeds flat along the x-axis. To interpret this behavior, we refer to analysis of appropriate and inappropriate scores presented in Figures 4.2 and 4.3 on page 73 of Chapter 4. In this analysis, a vertical climb is a result of assigning high score to instances. In this case, the model predicts these instances to be positive examples. Similarly, a horizontal run results

Table 5.3: Average AUC ± standard deviation for 5 BPET and 5 RF models over ten runs of cross-validation on UCI data. The lowest deviations are in bold.

	BPET		RF	
Data	Scored ROC	Standard ROC	Scored ROC	Standard ROC
prom	**0.86**		**1.94**	
echo	**1.42**		**1.56**	
hepa	**1.24**		**1.76**	
prks	**0.97**		**1.29**	
hart	**0.86**		**1.09**	
hrth	**0.46**			**0.89**
hors	**0.48**		**0.69**	
habr	**1.04**		**1.70**	
ıono	**0.48**			**0.57**
vots	**0.33**			**0.20**
ȷcrx		**0.35**		**0.46**
aust	**0.31**			**0.51**
wisc	**0.17**	**0.17**	**0.18**	**0.18**
blod	**0.53**		**0.63**	
diab	**0.43**		**0.73**	
mamo	**0.24**		**0.39**	
tic		**0.23**		**0.39**
ger	**0.44**		**0.82**	
oz8h	**0.45**		**1.01**	
oz1h	**0.55**		**1.60**	
chss	**0.01**	**0.01**		**0.05**
ads	**0.22**	**0.22**	**0.30**	
spam	**0.09**		**0.12**	
mush	**0.00**	**0.00**	**0.00**	**0.00**
mgic	**0.08**		**0.21**	
adlt	**0.05**	**0.05**	**0.18**	

from assigning low scores to instances predicted to be negative examples.

The progress made by the scored curve from the origin to the north-east corner of the space depicts the change in score values while observing their ranking order. The Naive Bayes model, in this case, assigns high scores to instances it predicts as positives, leading to the steep vertical climb. At some point, this model begins to assign low score values to instances abruptly. This situation is illustrated by the dark band in the right plot of

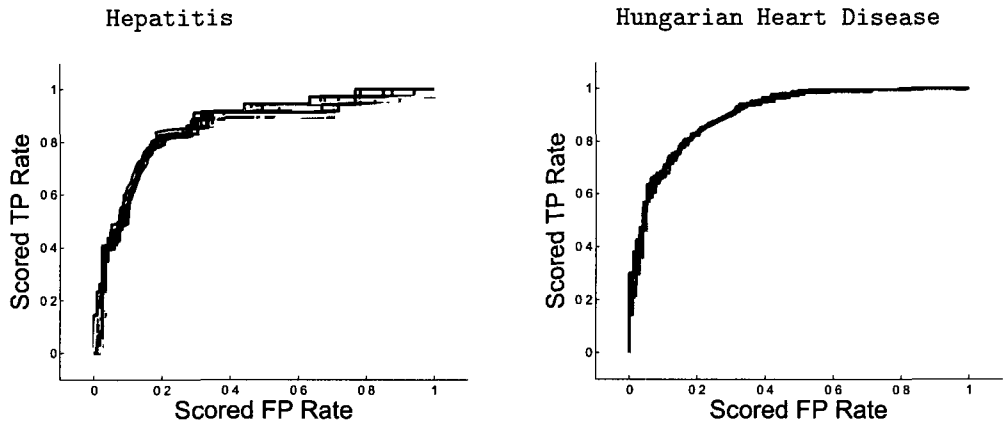

Figure 5.6: Scored ROC curves for NB (dark) and PET (light).

Figure 5.2 on page 97. This means that this Naive Bayes model can be effective when used for classification (or for ranking). It assigns high scores for the positive examples and low ones for the negatives. This change from high to low score values is indicated by the sharp change of direction in its scored ROC curve. When these scores are calibrated probabilities, they are described by DeGroot and Fienberg (1983) as "*refined probabilities*" – the values of probabilities assigned to the positives are near one and those to the negatives are near zero. Borrowing this terminology from probability theory, by neglecting that they may be uncalibrated probabilities, permits these scores to be called "*refined scores*".

Contrast the above to the light band of scored ROC curves associated with the PET model shown in the same plot (right plot of Figure 5.2). The PET model produces gradual changes in scores values without compromising their ranking order. The latter can be concluded from the lack of dominance in the standard ROC space in the right plot of Figure 5.3 of page 99. This shows that score refinement remains, by design, undetected by the standard ROC curve because score magnitudes are eliminated from the analysis. In fact, due to this refinement property, researchers prefer using PET scores, over NB scores, as probability estimates because their values change more smoothly over the ranking order. The visualizing of this property is a significant advantage of the scored ROC curve.

This disagreement between the scored ROC and the standard ROC curves can also be

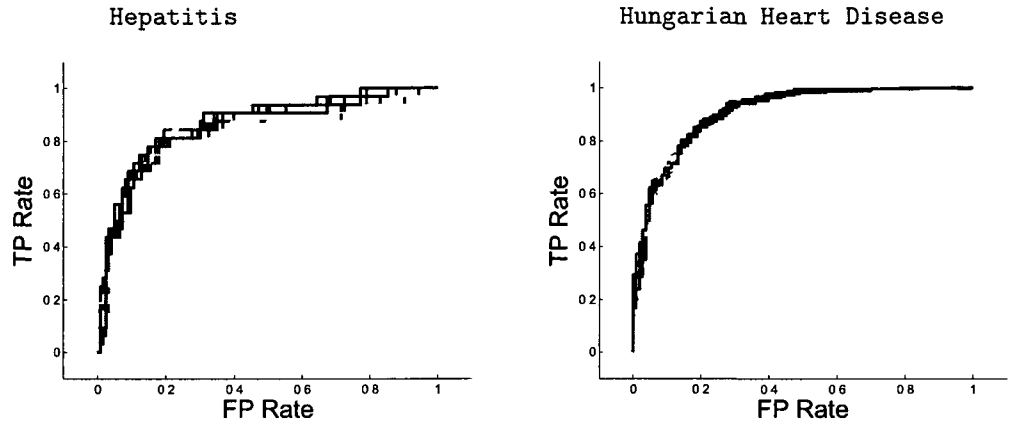

Figure 5.7: Standard ROC curves for NB (dark) and PET (light).

observed from several data sets, such as the `Hepatitis` and the `Hungarian Heart Disease` data shown in Figure 5.6 and 5.7. Curve separation is more clear in the scored ROC space but is absent (or less obvious) in the standard ROC space. This suggests that the models differ more in how they compute scores rather than in how they rank instances. The failure of the standard ROC curve to detect this separation is related to their design which depicts the ranking performance of classification. They are not meant to detect properties related to score values excluded by imposing a classification threshold. The dominance results for the UCI data are summarized in Table 5.4 using the average AUC and the separation score metrics discussed earlier Section 5.1.4.1 of this chapter.

In Table 5.4, column 1 shows the data set as listed in Table 5.1, and entries in columns 2 and 3 show the average AUC for PET and NB models obtained in the scored ROC space respectively. In the same scored ROC space, columns 4 and 5 show the two-way separation scores for these two models also respectively with PET dominating NB first followed by Naive Bayes over PET, the inverse. Columns 6 through 9 repeat these same values for the standard ROC space in the same order. Entries in bold show dominance according to the criteria we define in the experimental design which requires an AUC difference more than 2% and a separation score of at least 80%. The light entries fail to meet this criteria. In the scored ROC space, dominance is clear more often than not and can be measured in 18

Table 5.4: Dominance of PET and NB where AUC difference > 2% and separation ≥ 80% (in bold). AUC and separation are averaged over ten runs of cross-validation on UCI data.

	Scored ROC		Standard ROC	
Data	AUC PET vs. NB	Separation PET vs. NB	AUC PET vs. NB	Separation PET vs. NB
prom	**96.33**	**100**	**97.07**	**84**
hrth	**90.26**	**100**		
mamo	**89.07**	**100**	**90.16**	**98**
hors	**78.64**	**99**		
biab	**78.04**	**99**	**81.31**	**89**
echo	**77.27**	**97**	**78.67**	**90**
blod	**70.37**	**96**		
hart	**89.92**	**95**		
hepa	**85.18**	**90**		
ger	**75.04**	**86**		
oz1h	**97.47**	**83**		
jcrx	**87.66**	**80**		
oz8h	**94.37**	**80**		
aus				
wisc				
prks				
iono				
adlt				
habr				†
mush				‡
vots			**98.38**	**81**
ads	**95.00**	**80**	**96.23**	**87**
chss	**99.60**	**90**		
spam	**95.25**	**99**	**96.68**	**98**
mgic	**86.21**	**99**	**89.56**	**99**
tic	**87.95**	**100**	**93.24**	**83**

† Random performance (AUC is close to 50%)

‡ Perfect performance (AUC is close to 100%)

out of the 26 data sets. From these 18 sets, Naive Bayes dominate PET in 13 sets, for the remaining 5 sets the inverse dominance is true.

In general, there appears to be an agreement between dominance results obtained in the scored ROC and in the standard ROC spaces. The separation scores and the AUC

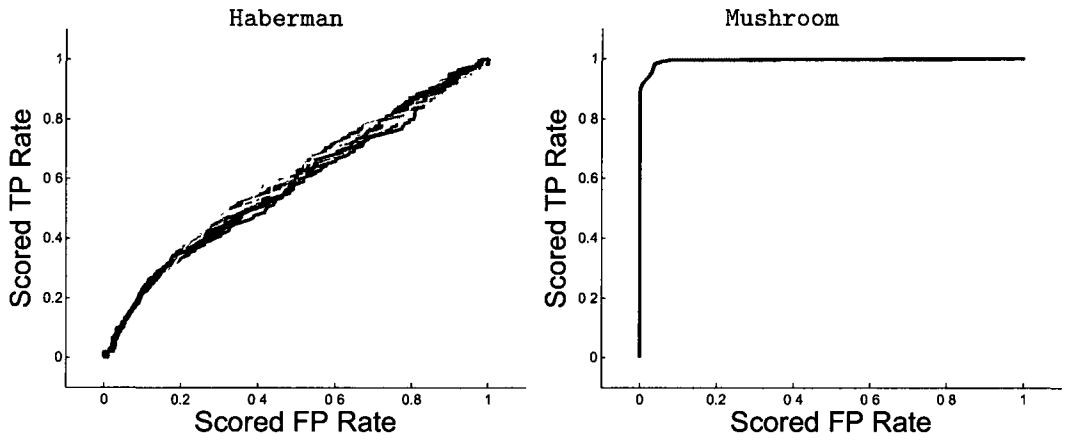

Figure 5.8: Scored ROC curves for NB (dark) and PET (light).

differences between the two models agree, however, their magnitudes differ. According to our criteria of dominance, standard ROC curves struggle to show consistent dominance whereas the scored ROC curves appear more consistent. For example, for the first three rows of Table 5.4 the AUC difference is approximately 10% with a separation score of 100%[1] in the scored ROC space. Entries in the standard ROC space, on the same three rows, show lower separation scores despite the obvious difference in their corresponding AUC values. These observations are consistent for data sets in the top portion of this table. Thus, dominance in the standard ROC space is more difficult to achieve because the curves are less separated than they are in the scored ROC space.

In the middle portion of the Table 5.4, dominance between the Naive Bayes and the PET models fails to meet the criteria in both spaces. Their AUC difference is less than or equal to 2%, or their separation score is less than 80%. Two exceptions can be made for the `Haberman` and the `Mushroom` data. Their performance curves are shown in Figures 5.8 and 5.9. The performance of both models is either poor due to their random performance (`Haberman` data), or they achieve optimal performance (`Mushroom`). For the former set, the average AUC suggest that both models perform close to 50% (no better than random). On the latter set, both models achieve average AUC close to 100% which makes it difficult to

[1]The sum of the two way separation scores may be over 100% due to rounding errors.

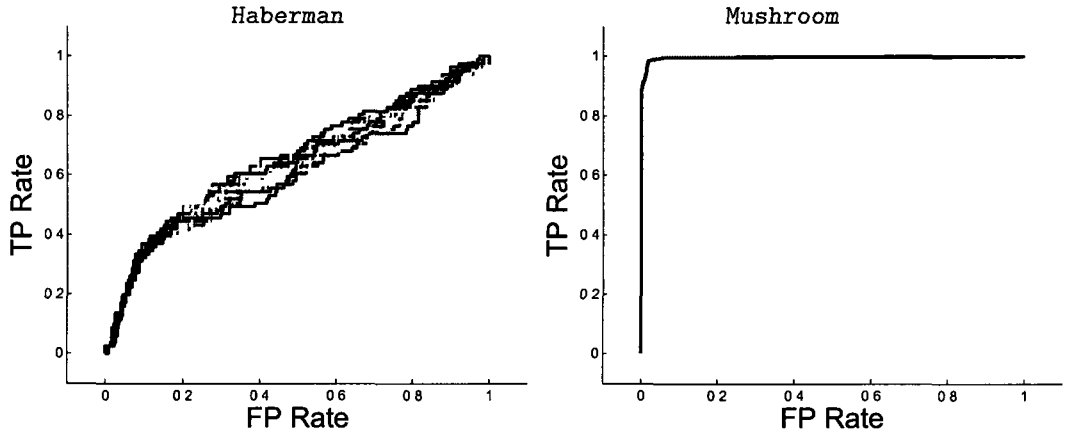

Figure 5.9: Standard ROC curves for NB (dark) and PET (light).

distinguish their individual performance. Their separation scores also remain within 50% in both spaces for both data sets.

The bottom portion of Table 5.4 shows the performance on data sets where the PET model dominates the Naive Bayes model. An interesting observation lies in that the average AUC and the average separation are higher for PET models in both spaces. This indicates that on some data sets, the PET model produces a better classifier with good scores than Naive Bayes.

In a similar manner, the dominance results comparing the Bagged PET (BPET) and the Random Forests (RF) models are presented in Table 5.5. The presentation format remains the same as that used in the previous table. If we consider the dominant scores (values in bold), we see that dominance is observed less frequently than in the previous table. This suggests performance similarities between RF and BPET in both spaces.

The RF models dominate the BPET models in the lower portion of the table more often than the inverse. However, their AUC differences along with differences in their separations scores are relatively small suggesting that these two models produce similar classification decisions (standard ROC space) and similar scores (scored ROC space). Such result are supported by the finding that BPET and RF models are equally effective classification models in Caruana and Niculescu-Mizil (2006). This can now be extended to include similarities

Table 5.5: Dominance of BPET and RF by AUC difference > 2% and separation ≥ 80% (in bold). AUC and separation are averaged over ten runs of cross-validation on UCI data.

	Scored ROC		Standard ROC	
Data No.	AUC BPET vs. RF	Separation BPET vs. RF	AUC BPET vs. RF	Separation BPET vs. RF
prom	**78.65**	**90**		
oz1h	**86.95**	**81**	**87.73**	**95**
oz8h				**91**
adlt			**90.59**	**99**
chss				
ger				
diab				
aus				
habr				
jcrx				
hart				
tic				
mgic				
blod			**71.87**	**82**
spam				
ads				
hrth				
echo	**65.50**	**81**		
mamo			**88.86**	**92**
iono	**94.64**	**82**		
hepa	**80.75**	**86**		
prks	**91.10**	**87**		
vots				
wisc	**97.80**	**92**		
hors	**65.91**	**99**		
nush				

in the scores they produce.

In this first component of experiments, we have constructed learning models on benchmark data selected form the UCI Machine Learning Repository Asuncion and Newman (2007). We have shown that these models are similar to themselves (by means of applying the same learning algorithm on various versions of the same data), and they are different from each other (by means of applying different learning algorithms to the same data). In

either case, the results show that comparisons of class membership scores are more sensitive to performance similarities or differences than comparisons of classification decisions. In the next experimental component, we use a synthetic data generator to implement performance differences to illustrate the ability of the scored ROC curve to measure performance differences between training and testing for a given learning model.

5.2 Component II: detecting changes in the domain

5.2.1 Design

In our experiments this far, training and testing data have been drawn from the same distribution. However, an alternative source of differences in performance may be changes between training and testing data. This can easily occur in changing data distributions. In addition, it has been established that determining the mid point with which we determine score appropriateness is sensitive to changing data distribution (Forman, 2005; Bennett, 2003). Therefore, it is reasonable to expect that the scored ROC curve to be sensitive to such changing environments. With such rationale, we conduct the experiments presented in this section to verify such sensitivity of the scored ROC method.

For this component of our experiment, we utilize a synthetic data generator to obtain data from a changing distribution. The purpose is to demonstrate that our evaluation method is capable of capturing performance differences between training and testing. Such performance differences occur when the learning model is tested on data drawn from a distribution different from that used for training. Such models are expected to produce similar scores when tested on data drawn from the same distribution, however, testing them on a modified distribution should result in different scores. Our experiments show that the scored ROC curves are more suited for measuring such differences. To this extent, the following section describes the design of the synthetic data generator used in this experiment followed by presentations of our experimental results.

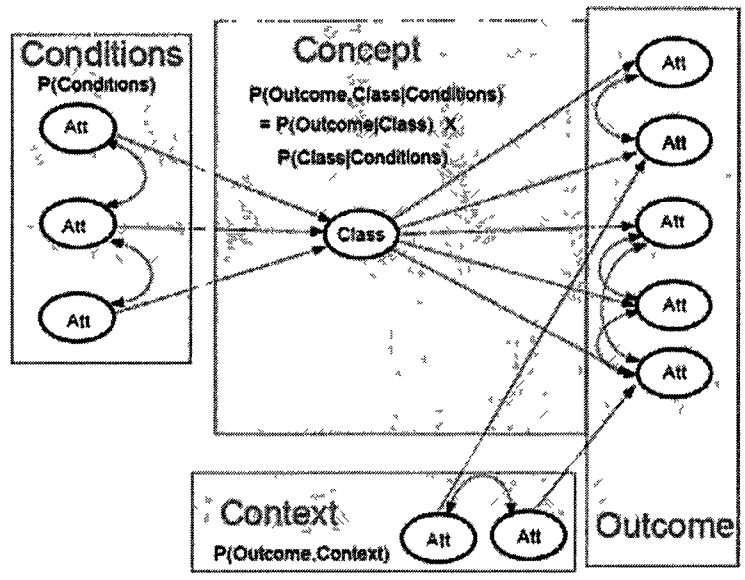

Figure 5.10: The heart attack model used to generate synthetic data (Seifi, 2009).

5.2.1.1 Heart attack generation model

This synthetic data generator is described by Seifi (2009) This is an internal report prepared by a fellow graduate student at the Ottawa-Carleton Institute for Computer Science in collaboration with C. Drummond, S. Matwin, and N. Japkowicz. The principle of this data generator is to build a realistic probabilistic data generation model that simulates Heart Attack patients based on distributions approximated from real-life data. As illustrated in Figure 5.10, this generator consists of four components which categorize data attributes as shown in Table 5.6. These components include.

1. Conditions describe features on which the class (the concept) depends probabilistically. This category represents the population of heart attack patients.

2. Outcome features describe symptoms common to heart attack patients.

3 Contextual attributes affect the outcome of a concept and make the diagnosis more difficult They are symptoms, similar to those of a heart attack, which arise due to other diseases or conditions.

Table 5.6: Attribute categories for heart attack data (Seifi, 2009).

Conditions	Concept	Outcome	Context
Gender	Heart Attack	Chest Pain and tightness	Stomach
Age		Upper body discomfort	Lung
Weight		Shortness of Breath	Asthma
Smoking Type		Sweating	Shock
		Dizziness	Infection
		Nausea	Stress
		Sleep Disorder	Flu
		Weakness	Diabetes

This data generator relies on several sources of real life data to estimate conditional probability distributions and dependencies between the various components of its probabilistic. These sources include; Statistics Canada, StatLib Data Archive (University of Carnegie Mellon), the Behavioral Risk Factor Surveillance System Project (Centers for Disease Control and Prevention – U.S.), and heart disease data from the UCI Machine Learning Repository (Asuncion and Newman, 2007).

5.2.1.2 Differences from training to testing

For simplicity of presentation, we omit details of probabilities used to generate data in the default settings. These are found in (Seifi, 2009). However, we describe probabilities involved in our experiments. We only describe probabilities being modified for the purpose of our experiments, which implement three types of modifications:

A. Concept Drifts occur when the risk of having a heart attack, among patients with heart attack symptoms, changes from the original default risk.

B. Contextual Drifts (or Noise) are represented by changes in probabilities of contextual conditions. They produce symptoms similar to those of a heart attack patient but in non-heart attack patients.

C. Population Shifts are changes in proportions of particular types of patients generated.

Algorithm 8 Identifying a candidate attribute for modification.

1: **Input:**
D is a set of n instances generated with the default settings.
C is a set of attributes belonging to one category (see Table 5.6).
2: **Output:**
A is an attribute $\in C$ whose probability is to be changed.
3: aPET = construct a PET model by training on D.
4: for all nodes in aPET, return $A \in C$ closest to the root.

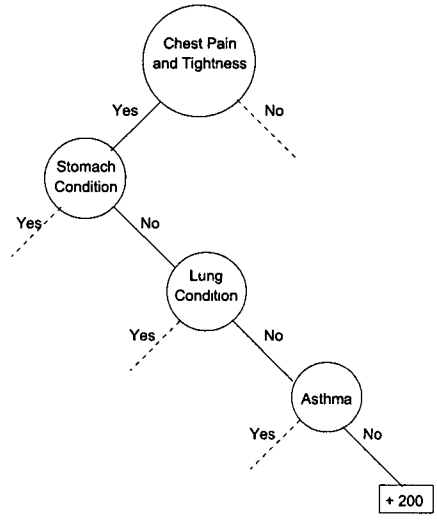

Figure 5.11: PET model constructed from the original heart attack distribution.

5.2.2 The synthetic data

In our experiments, we generate data sets that contain these distributional changes individually to prevent interactions of multiple modifications. Therefore, we make a single change to the distribution after training, and then, we measure the corresponding effect on testing performance. In addition, we predict the effect of this modification on the performance so that we are able to verify that differences are being captured accordingly. Therefore, we generate these modified data in a controlled manner using a specific model.

We first generate a sample of 1000 points obtained with the default settings for the construction of a PET model. We analyze this PET to identify candidate attributes whose associated probabilities can be changed. This PET model offers several advantages in

Algorithm 9 Experiment design for modifying the distribution.

Input:
A is an attribute in the data.
$P(A)$ is the default probability of A.
M is a learning method.
Output:
S a set of 13 arrays where S_i is an array of n scores and $S_{ij} \in [0, 1]$.
L is a set of 13 arrays where L_i is a array of n labels and $L_{ij} \in \{+, -\}$.
D = 1000 data points generated with the default settings.
C = an instance of M trained on D.
{Testing 11 holdout sets of modified distributions.}
for $i = 0$ **to** 10 **do**
  $P'(A) = \frac{i}{10}$
  T_i = generate 1000 points of data with $P'(A)$
  L_i = labels of T_i
  S_i = class membership scores of testing C on T_i
  $i = i + 1$
end for
{Testing one holdout set of the original distribution}
T_i = generate 1000 points of data with $P(A)$
L_i = labels of T_i
S_i = class membership scores of testing C on T_i
$i = i + 1$
{Testing on the original distribution with cross validation}
L_i = labels of D
S_i = class membership scores of testing C on D by 10-fold cross validation

identifying important attributes. It enables the inspection of its structure and offers a reasonable quality of scores. Therefore, the PET shown in Figure 5.11 is constructed and inspected to determine its most important attribute located at the root of the tree. In this case, this attribute is the `Chest Pain and Tightness` symptom.

In this PET model, there are 200 (out of 1000) positively labeled patients located in a pure leaf-node. These patients suffer from chest pain, have no other contextual conditions (stomach, lung, or asthma conditions), and are correctly classified as positives for having a heart attack. According to this PET, this attribute is a reasonable candidate whose probability, when changed, can impact the performance of this PET classifier.

Algorithm 8 identifies candidate attributes whose probabilities are selected for modification. Algorithm 9 shows the procedure of producing scores for the test sets including;

holdout sets with modified probabilities, a single holdout set with the original probability, and 10-fold cross validation on the training set. For the purpose of this experiment, the testing performance is measured by computing the AUC under the scored ROC and under the standard ROC curves. Then, we compute the differences of these respective AUC values to that resulting from the cross validation testing. In other words, the 10-fold cross-validation test is used as a baseline for measuring differences in testing performance.

The curve separation score is omitted in this experiment due to its unreliability when AUC differences are small. In such a case, the curves may trade positions above and below each other frequently, and they may overlap with small gaps. In addition, curve separation results coincide with those obtained by measuring AUC differences. Therefore, the curve separation measure is unnecessary in this part.

Returning to the attribute identified earlier in this section, the `Chest pain and tightness` symptom is the most important attribute in determining the risk of having a heart attack according to the PET model presented in Figure 5.11. In the default settings of the data generation model, the probability of a heart attack among patients who suffer from `chest and pain tightness` is 0.78 for men and 0.45 for women. In this case, the concept is the risk of having a heart attack whose probability is fixed for both men and women. If we wish to change the probability distribution of this concept to generate concept drift, we should modify this probability associated with it. The objective of this part of our experiment is to measure performance differences for testing with all values of heart attack risk. A minor issue lies in the default setting of this risk being different for men than it is for women. To simplify the analysis, we only modify this risk for men not for women so we can build an expectation of how the performance of the PET model, trained on the default distribution, is affected by this concept drift.

Following Algorithm 9, we generate holdout sets containing data points generated with the above probability modified to each of $\{0, 0.1, 0.2, \cdots, 1.0\}$. The default risk is 0.78, and we expect the performance difference to be lowest near the default risk and highest near 0 and 0.1, because the error should be higher. The same effect can be expected when the

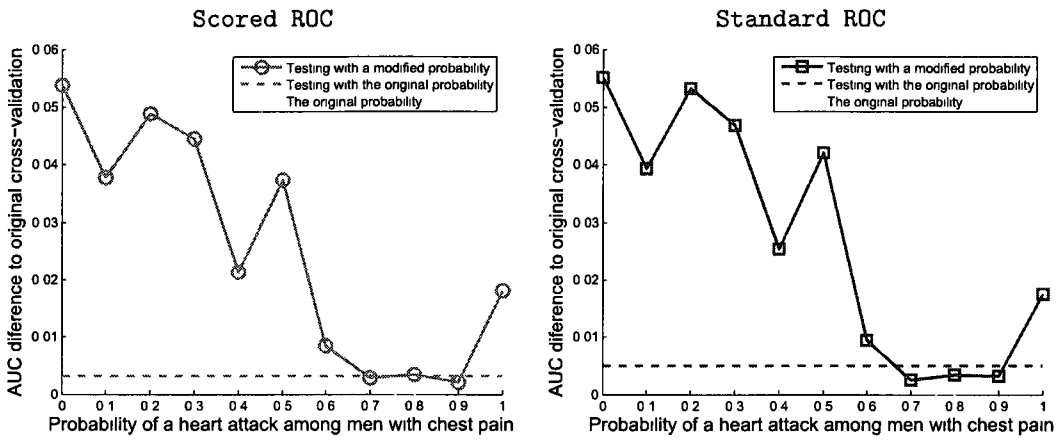

Figure 5.12: AUC differences for PET model with a concept drift.

risk increases well beyond 0.78, however, the performance difference should drop close to 0 near 0.78. This behavior can be expected due to the fixed probability value with which the learning model is trained. Presumably, the trained model classifies 78% of men with chest pain as positives for suffering a heart attack. As this probability decreases towards 0 in the test sets, the classification (and scoring) errors should gradually increase. Similarly, these errors should increase when this risk increases above 0.78. The interesting point lies near the 0.78 probability where the performance difference is expected to be near 0 and within the observed difference obtained by testing in the original (default) probability distribution.

In this experiment, we apply the PET model shown previously for the purpose of verification only. Because changes in the data distribution can possibly be irrelevant to the learning model being tested, the experiment should use a different learning algorithm for testing. If the same model, used to identify important attributes, is used for testing, it can be argued that modifications introduced to the data distribution are known to be relevant to the model. Therefore, we repeat the same experiment using Naive Bayes learning to ensure that the modified attributes are selected independently from the model being tested.

To verify our expectations, we test our PET model (discussed earlier) on the holdout sets produced by Algorithm 9. For each data set, we measure the performance difference between testing with the set itself and with the baseline, 10-fold cross-validation. These

AUC differences are measured between performance curves in both spaces shown in the plots of Figure 5.12. The solid curves represent AUC differences for test sets containing modified risk probabilities. The dashed line represents the AUC difference obtained by testing on a hold out set with the default probability. The gap from the x-axis to the dashed line represents the amount of AUC difference observed when testing in the original, unmodified distribution. This depicts the tolerated AUC difference due to random chance.

Points belonging to the solid curves that lie below the dashed lines show an AUC difference no more than that observed by testing on data drawn from the original distribution. In Figure 5.12, testing the PET model on data with risk probability between 0.7 and 0.9 produces an AUC difference no more than that observed by the baseline testing. Hence, we can conclude that this PET model performs similarly for heart attack risks, among men with chest pain, between 0.7 and 0.9 probability. Outside this interval, the situation differs in both plots. The scored ROC and the standard ROC curves produce higher AUC differences when the risk is low (close to 0). This performance difference decreases gradually as the risk probability increases towards 0.7. When the risk increases beyond 0.9 probability, this AUC difference climbs up again. The small circle shown on the x-axis represents the value of the default risk used in training.

Technically, these results are less surprising because we modify the data distribution based on the structure of the PET tree. This PET is trained to classify men with chest pain as positives for a heart attack with probability of 0.78. It is intuitive that the farther the risk probability increases (or decreases) away from its original value, the higher the scoring, and therefore, the classification error will be higher. When the risk probability is close to its original value, the AUC difference should be minimal to within values observed by testing in the original distribution. Figure 5.12 illustrates exactly these effects. Curves in both spaces capture these AUC differences successfully in a similar manner. The spikes in the solid curves may be attributed to the model being a decision tree because decision tree classifiers tend to be definite in their classification rather than smooth.

An interesting observation lies in the gap between the solid curves and the corresponding

Table 5.7: Modifications to the heart attack data distribution.

Type	Probability of	Category	Original Value	Modified Values
Drift	heart attack among men with chest pain symptom	concept	0.78	$[0, 0.1, 0.2, \cdots, 1.0]$
Noise	stomach condition	context	0.1	$[0, 0.1, 0.2, \cdots, 1.0]$
Shift	men in population †	condition	0.496	$[0.1, 0.2, \cdots, 0.9]$

† The only population (or condition) related attribute used by our PET model.

dashed lines within the probability interval [0.7,0.9]. The AUC differences, in this interval and in both plots, remain close or below the dashed line. They remain within the tolerated AUC difference. However, in the right plot, this segment of the solid curve fails to reach the dashed line. The same curve segment in the left plot gets much closer to its corresponding dashed line. The left plot is based on scores assigned to the test data and is more accurate in detecting performance differences because it detects AUC difference equal to that of the baseline. This discrepancy between the left and the right plots may be attributed to the use of classification thresholds that ignore the magnitudes of the scores. In addition, it is important to note that the dashed line in the left plot is lower than that in the right plot. This means that the observed AUC difference by testing in the original probability distribution produces higher difference in the standard ROC space (the right plot) than it does in the scored ROC space (the left plot). This suggests that measuring performance differences using the score-based method is not only more appropriate but more sensitive than standard ROC method due to taking the magnitudes of scores into account. The standard ROC curve is more prone to fluctuations due to ignoring the scores.

5.2.3 The learning model and evaluation

Having verified the ability of both methods, the scored ROC and the standard ROC methods, to measure performance differences between training and testing, we now proceed to the experiments themselves. Using the above method of identifying candidate attributes,

whose probabilities we modify, we perform three experiments which aim to measure AUC differences resulting from the modification of the attributes listed in Table 5.7.

For each type of modification, we repeat the above experiment using Naive Bayes. The previous PET model is used for the sole purpose of identifying candidate attributes relevant to the concept being learned, however, such identification method may produce results specific to this PET model. In addition, the naive Bayes learning method is a probabilistic method better suited for the data generation model. Therefore, in subsequent runs of this experiment, we use Naive Bayes learning instead. Once again, we measure AUC differences resulting from making a single change according to Table 5.7 in each run.

5.2.4 Results

5.2.4.1 Domain changes due to a concept drift

This experiment run aims to measure performance differences in either space due to a concept drift in the domain. We modify the concept attribute representing the risk of having a heart attack among men who suffer from chest pain, see Table 5.7. We repeat the previous experiment using a Naive Bayes learning method instead of the PET model. After training a Naive Bayes classifier on data drawn with the default (original) risk probability, we generate holdout sets containing modified probability values as shown in the last column of the same table. On these holdout sets, we test the trained Naive Bayes model to measure its performance based on comparing the scores it produces (by building the scored ROC curves) and based on its classification decisions (by building the standard ROC curves). We hope to obtain performance differences similar to that produced by the PET model.

Using the same presentation described earlier, the results for this run are shown in Figure 5.13. At first glance, we see that testing the Naive Bayes model in the original distribution produces higher performance difference in the standard ROC space than it does in the scored ROC space. This is illustrated by the dashed line in the right plot being higher than that in the left plot. This remains consistent with our previous observations that the standard ROC depicts higher AUC differences between models that are supposedly

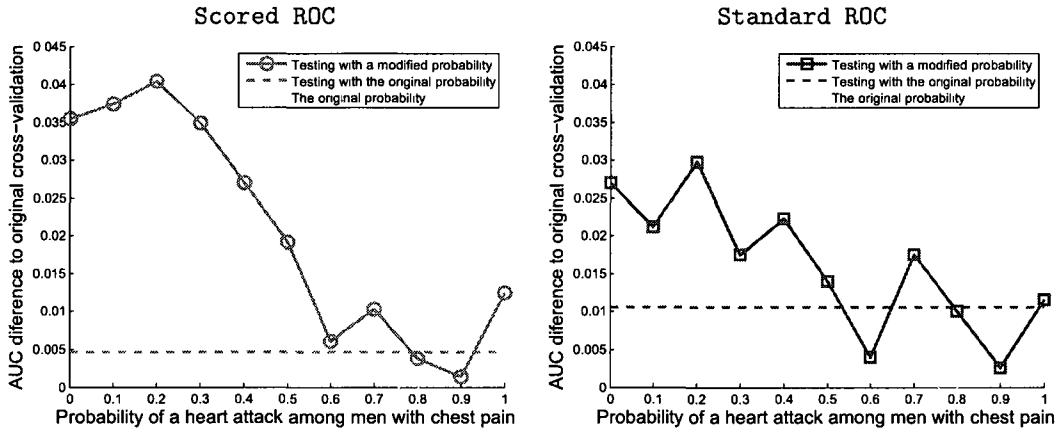

Figure 5.13: AUC differences for Naive Bayes model with a concept drift.

similar. Their training and testing data are drawn from the same distribution.

Comparing the solid curves in the left to right plots of Figure 5.13 shows that AUC differences in the scored ROC space (on the left) gradually decrease in values as the risk probability increases from 0 to 0.6. In contrast, the same solid curve, computed for AUC differences under the standard ROC curves in the right plot, appears more ragged and lacks consistency. However, it too does eventually fall towards and more importantly below the dashed line at probability value 0.6. For risk probability values in the interval [0,0.8], the resulting AUC differences in the scored ROC space are higher than those measured when testing within the default probability distribution, because the original risk probability is 0.78. In the standard ROC space, however, these differences drop below the tolerated AUC difference at risk probability value 0.6, the solid curve in the right plot dips below the dashed line. A similar discrepancy appears when testing with risk probability equal to 1, the solid curve in the scored ROC space climbs high above the dashed line, whereas the same curve barely goes over the dashed line in the standard ROC space, in the right plot.

Another disagreement between results in the two spaces lies in how close the AUC difference is measured near the original value of the probability distribution. For instance, in the scored ROC space in the left plot of Figure 5.13, the solid curve approaches the same performance difference measured when testing with the original probability value, because

the solid curve is on the dashed line and close to the small circle on the x-axis indicating the original probability value. In the standard ROC space, however, the same situation can be observed but with a higher AUC difference (0.01), which is almost double of that observed in the scored ROC space (approx. 0.005). Essentially, the curve in the left plot comes closer to AUC differences measured with the original probability distribution than in the right plot. Such disagreement between the two spaces suggests that AUC differences measured in the scored ROC space are more sensitive to changes the probability distribution of this concept. We can conclude that using the scores to detect changes in the performance due to changes between training and testing distributions is more sensitive than using classification decisions.

5.2.4.2 Domain changes due to a contextual drift - noise

In this section, we present results of testing the same Naive Bayes classifier, used in the previous experiment, on test data with a changing probability for a contextual attribute. A contextual attribute affects the outcome of the concept but is not part of the concept itself. For this risk of heart attacks, the attribute `Stomach Condition` (identified in Table 5.7) can cause symptoms similar to those commonly associated with a heart attack. In the default settings of the generator, the probability of a patient to have a stomach condition is 10% and 35% of these show `Chest pain and Tightness` symptom. Since the `chest pain symptom` is an important attribute for this data model, modifying this attribute directly impacts the risk of having a heart attack. Generating patients with stomach conditions who show chest pain symptoms will effectively generate noisy data. These patients have the same symptom as heart attack patients but they do not have heart attacks.

To maintain control over noise generation, we only modify the probability of developing a stomach condition. The probability of showing the chest pain symptom among those suffering from stomach condition remains unchanged from the default setting of 35%. While generating test data with 0 probability of stomach condition results in less noise, increasing the probability of this condition beyond the 10% increases noise. Therefore, we can expect

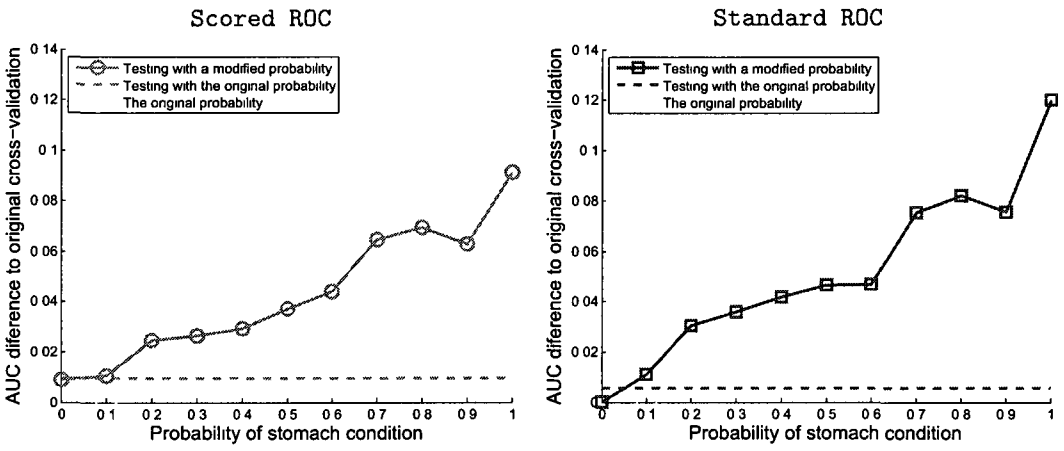

Figure 5.14: AUC differences for Naive Bayes model with noise (contextual drift).

the performance to gradually deteriorate as the test data becomes more noisy, and the AUC difference should gradually increase due to the deteriorating performance.

The resulting AUC differences for this experiment are presented in Figure 5.14. The AUC difference increases as the probability of the condition increases. These differences behave more or less in a similar manner in the scored ROC space on the left as they do in the standard ROC space on the right. Both curves maintain a similar shape and magnitude in both plots. The agreement in the two spaces suggests that performance differences due to testing with noisy data is the same in both spaces. However, a small but important difference occurs in the probability interval [0,0.1]. In the scored ROC space (in the left plot), the AUC difference remains equal to that observed by testing in the original distribution where the solid curve trails along the dashed line in this probability interval. In the standard ROC space, the solid curve begins at the origin and climbs above the dashed line for probability 0.1. When the probability of the `Stomach Condition` reaches the default settings of 0.1, the AUC difference is higher than the tolerated difference (the dashed line). This conclusion, drawn from the standard ROC space and shows that when the modified probability reaches the same value as that of the default original distribution, the AUC difference remains higher than that measured by testing in the original distribution. This reinforces our previous results that measuring performance differences based on classification

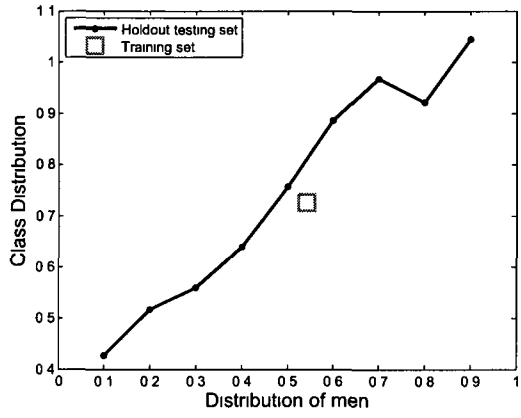

Figure 5.15: Class distribution (+/−) for generated population shifts in men.

decisions is less sensitive to changes in the probability distribution.

5.2.4.3 Domain changes due to a population shift

In this final experimental run, we illustrate the ability of the scored ROC method to capture performance differences due to a shift in the population. The focus of this run is the distribution of men in the data as listed in Table 5.7 on page 119. Although the distribution of men does not appear high in the PET tree model shown in Figure 5.11 on page 114, it is the only population-related (or condition related) attribute that appears in the model. Inspecting the PET tree fully, we have determined that this attribute appears low in the tree just above the leaf level. For this population shift, changing the proportion of men in the population is expected to impact the performance of the Naive Bayes model due to several factors. The Naive Bayes classifier is, by design, sensitive to the class distribution and is affected by changes in the population particularly when the probability of getting a heart attack is different for men than it is for women.

Figure 5.15 shows the class distributions in data generated for this experiment. The ratio of positives to negatives is plotted on the y-axis as the portion of men increases in the population on the x-axis. Initially, there is a sharp climb in the class distribution between 0.1 and 0.2 proportion of men followed by a slightly lower rate of climb for men proportion

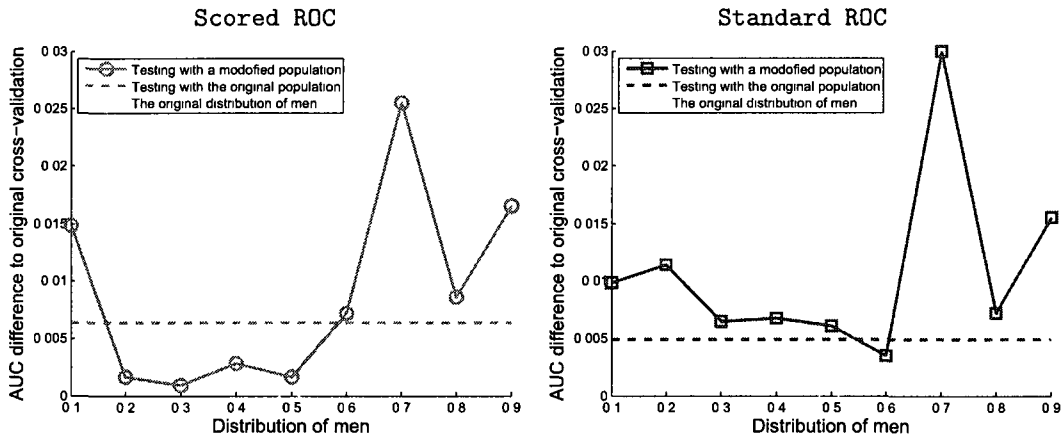

Figure 5.16: AUC differences for Naive Bayes model with a population shift.

0.2 to 0.4. A steady increase in the class distribution builds up to a high local peak at 0.7 distribution of men. A sharp drop[2] begins after 0.7 probability of men with a final rapid climb at 0.9. Over all, it is clear that as the proportion of men increases in the population, the class distribution rises. The single square point in the centre of the plot depicts the class distribution measured from the training data. Of course, this point does not lie on the curve due to the randomization in the data generator.

Naive Bayes algorithm multiplies individual conditional probabilities of attribute values by the class probability, which depends on the class distribution. It is intuitive to expect its performance to be affected by this change. Our objective is to show how such changes in the population affect the performance of the Naive Bayes classifier in the scored ROC space and then in the standard ROC space. Similar to the previous runs, performance differences are measured by computing the AUC difference between testing in the modified distributions and testing in the original distribution. These results are shown in Figure 5.16.

At first glance, we observe high AUC differences, in both spaces, for either low men population (0.1 to 0.2), or for high men population, well above the original population value approx. 0.5. Furthermore, the increase or decrease in the AUC difference appears to

[2]The class distribution of heart attach patients drops for a high (above 0.7) distribution of men in Figure 5.15. The is based on the realistic distribution of data used by the generator. It can explain the drop of AUC difference apparent at 0.8 distribution of men in Figure 5.16

correlates with the rate of climb of the class distribution shown in Figure 5.15. In a sense, the high rate of climb in the class distribution (for intervals [0.1,0.2], [0.6,0.7], and [0.8,0.9] of men population) appears to produce a high AUC difference in both, the scored and the standard ROC spaces. When this rate of climb drops in Figure 5.15 (intervals [0.2,0.3] and [0.7,0.8] men population), the AUC difference also decreases in both spaces. Such apparent correlation supports our intuition of performance differences resulting from shifting the population of men particularly if such differences can be observed in both spaces.

However, a disagreement between results obtained in the scored ROC space and in the standard ROC space, lies in the interval [0.2,0.6] of men population. The scored ROC space depicts AUC differences lower than those observed by testing with the original population of men, the tolerated difference represented by the dashed line. The standard ROC, on the other hand, shows AUC differences higher than this respective tolerated AUC difference in the same interval of population [0.2,0.6]. Over this interval of population, the solid curve in the scored ROC space drops well below the dashed line whereas, it remains above the respective dashed line in the standard ROC space. When comparing both plots, it is clear that the solid curve on the left, gets closer to zero when the population of men equals to the original value (approx. 0.5). The same curve in the standard ROC space barely dips below the dashed line for men population of 0.6. This difference suggests that standard ROC curves detect higher performance differences despite the population of men being close, in value, to the original distribution. Thus, the scored ROC curves appear more accurate in depicting low differences between testing on data of equal populations of men.

Although AUC differences (the solid curve) drop close to zero at the original population in the left plot they remain below the dashed line. They remain within the tolerated performance difference. It is intuitive that the difference in performance is related to the changing population of men. These changes are more pronounced because gender is a binary attribute which means that the population of women is also affected by this shift. This is depicted in the scored ROC space by high and by low AUC differences above the dashed line. However, near the original value of the men population, these differences drop close to zero.

If we wish to determine the value of population of men in the data with which the Naive Bayes classifier is trained, the scored ROC plot shows that it lies inside the interval [0.2,0.5]. In other words, the performance of this classifier remains unaffected for those proportions of men in the population. The standard ROC plot, however, shows a single possible value 0.6 because it is the only value for which the observed AUC difference drops below the observed difference from testing in the original population. This behavior of the standard ROC curves illustrates the premise of this thesis that measuring performance differences using the scores generated by the model provides additional valuable performance information ignored by the standard ROC curve.

5.3 Chapter summary

This chapter presents experiment design, evaluation and results. The experiment consists of two components; the first measures performance similarities and differences among learning models constructed from UCI data (Asuncion and Newman, 2007), and the second illustrates the ability of our method to capture performance differences, artificially introduced into the domain with the aid of a synthetic data generator. In both components, we compare performance conclusions made in the scored ROC space against those made in the standard ROC space using several evaluation metrics, criteria, and performance measures.

Experiments, described in this chapter, assess performance in the presence of three types of variations resulting from randomly splitting the data, randomly selecting a subset of features over data, and finally, systematically modifying the data distribution between training and testing. In each case, experimental runs are performed to measure performance similarities and differences whenever appropriate.

The results show that the scored ROC method measures performance similarities with tighter standard deviations compared to the standard ROC curve. This is a result of being more sensitive to model differences than the standard ROC curve. The scored ROC method produces more accurate estimates of performance comparison due to utilizing score

magnitudes whereas the standard ROC method eliminates the margins between scores by imposing classification thresholds. Due to this latter design difference between the two methods, the standard ROC appears less reliable and more erratic in depicting similarities and dominance for scoring classifiers. When measuring differences among classifiers, again the scored ROC method appears to be more sensitive to variations of classifier performance than the standard ROC. The latter measures unnecessary differences when changes are absent, and it inconsistently measures performance differences when changes are present. A clear conclusion we make is that the standard ROC method performs reasonably well for what it was designed to measure. It is designed to measure performance based on classification decisions. We conclude that comparing the performance of learning models, using the scores they produce, results in a more refined measure capable of conveying additional information ignored by classification decisions alone.

With respect to our tested hypothesis, presented at the beginning of this chapter, the scored ROC curve measures performance similarities and differences based on the scores produced by the learning model. When compared to the standard ROC curve, the scored ROC offers superior sensitivity even in changing environments.

Chapter 6

Conclusions

This thesis described a categorization of common machine learning tasks based on the information content of their outcome. This categorization allows for the identification of performance measures that can be used to evaluate them. This organization provides an understanding of how to convert learning tasks to access information conveyed by their outcome to help the performance measure cope with practical issues.

Using the above categorization, this thesis develops the scored ROC curve to incorporate class membership scores into the ROC curve. This novel method combines the performance of classification, ranking, and scoring and forms a performance visualization method that preserves features of the ROC analysis. It extends the ROC curve by introducing the scores as weights applied to individual line segments of the curve. Effectively, the new curve is a smoothing of the ROC curve based on these scores, which expresses the confidence of the model in its predictions. Such an extension addresses a gap that exists among performance measures in the machine literature for measuring the performance of scoring as illustrated in Chapter 1. Our analysis shows that the scored ROC curve can be reduced to the standard ROC curve by converting the scores, even when they are probabilities, to binary values. This reduction provides users with flexibility in performance analysis.

The results of our experiments demonstrate the ability of the scored ROC curve to measure similarities and differences among learning models. The experiments illustrate

how the scored ROC curve can be used to visually compare scores obtained from various learning models. Such comparisons reveal how models assign scores to instances by visualizing the distribution of scores to data points. The use of our method to compare scores produced by the Naive Bayes method to those obtained from the Probability Estimating Trees (PET) confirms results obtained by studies found in the literature. Comparing scores of the Random Forests to those obtained from the Bagged Probability Estimating Trees (BPET) reveals that they produce similar scores. However, Random Forests result in a slightly higher AUC variance. This result is not surprising due to similarities in the two algorithms. They both construct decision trees on randomly selected subsets of instances. However, the Random Forest goes a bit further and selects a random subset of features as well. This explains the higher variance observed.

Having illustrated our method's ability to measure similarities and differences among learning models, our experiments compare how well such characteristics are measured by the scored ROC and by the standard ROC curves. The results show that the scored ROC curve measures performance similarities with a smaller standard deviation of the area under the curve than the standard ROC method. The scored ROC curve is more sensitive to differences between models than the standard ROC curve and produces more consistent results. The last component of our experiments, performed on synthetic data, shows that the scored ROC curve is sensitive to performance changes when data is obtained from changing environments.

With these results, we conclude that the method, developed by this research, enhances the performance evaluation of learning models using class membership scores rather than their classification decisions. The scored ROC curve captures performance information with a higher granularity than the standard ROC curve. Therefore, it conveys more specialized performance information, yet it remains more abstract than dealing with probability estimates that are parametric in nature. Our method, however, has limitations. These are summarized in the next section followed by a discussion of future research.

6.1 Limitations

The scored ROC curve is a visualization method that extends the ROC analysis to include classification scores as weights that smooth the ROC curve. Our approach is a novel method that addresses a gap in the performance analysis literature. However, it features some limitations caused by various circumstances ranging from assumptions made to characteristics of the problem itself. The following sections discuss these limitations.

6.1.1 Limitations due to assumptions

An obvious limitation to our approach is the assumption of a binary classification problem. This assumption is necessary because the scores are produced by the learning model to indicate how positive a given instance is. In a sense, the scores are understood to express the confidence of the classifier in its predictions in two classes, the positive and the negative class (Fawcett, 2003). When dealing with these scores, our method uses their magnitudes and computes the magnitudes of their complements. While the score depicts how positive the instance is, the compliment score indicates how negative the instance is thought to be. Our method uses these two score values to assess the performance. In a multi-class problem, computing the compliment of the score may alter its interpretation. In this case, further interpretation may be necessary and has not been addressed in this work.

Despite this difficulty, our approach can handle multi-class problems when they are represented in a binary form. This can be achieved by considering a single class as positive and the remaining classes are combined into a single negative class. Such approach has been used in the standard ROC analysis (Fawcett, 2003).

Another challenge is the assumption of the scores being in the $[0, 1]$ interval. Most learning models assign scores to instances between 0 and 1. In fact, many methods claim to estimate class membership probabilities, which we argue, is not necessarily correct. For instance, the `Weka` software package (Witten and Frank, 2005) produces the scores as though they are class membership probabilities. We argued that computing these scores may not

produce probabilities (or provides poor estimates of probabilities) for various reasons. In any case, `Weka` computes such scores between 0 and 1. However, one can imagine a general value scoring function which assigns scores outside the desired interval. In this case, a transformation between general score values into the $[0, 1]$ interval is needed for our method to work. While a mapping between general score into the desired interval may be difficult to obtain, we can attempt to normalize them if we are given a maximum and a minimum values. However, the effect of this normalization on their interpretation becomes questionable.

Having dealt with the score values, other limitations may exist due to the distribution of score values over the data points. Such limitations are related to characteristics of the domain. These are discussed in the following section.

6.1.2 Limitations from the domain

Traditionally, training and testing a learning model is performed on instances which are assumed to be randomly drawn and representative of the domain. Therefore, data driven analysis are usually limited to characteristics of the data sample with which the model is trained and tested. Like with most evaluation methods, this situation occurs with the scored ROC curve as well. It constitute a limitation particular to calculating the midpoint value discussed in Section 4.3.2.2 on page 74.

The midpoint value determines the appropriateness of score values where appropriate scores have values greater than or equal to it. The score values are assumed to be in the $[0, 1]$ interval, and therefore, this mid value is required to remain within the same interval. However, its value is calculated using the average score value (weighted by the class distribution). Thus, it is plausible that this calculation results in a midpoint value outside the desired $[0, 1]$ interval. This situation is particularly obvious for severely imbalanced data set (with a skewed class distribution). Such effect can be detrimental to our analysis because it cases all instances of the data to fall on one side of the midpoint value.

Related to this same situation is the distribution and sparseness of score values above and below this midpoint value. These can also affect our analysis. For instance, if the

majority of score values fall on the one side of this midpoint value, our analysis will become less sensitive to scoring errors. Ideally, the midpoint should be able to divide the instances above and below its value sufficiently. An important issue is determining such sufficiency.

An additional point of discussion lies in the performance of the learning model. Our method is designed to measure differences in score magnitudes for models that perform reasonably well. If these class membership scores are estimated with perfect performance, where positive instances are assigned ones and negatives are given zeros, the scored ROC curve becomes identical to the standard ROC curve. In such a case, our method conveys the correct performance in an identical manner to the standard ROC curve. Similarly, if the performance of the learning model is close to being random, where score values are near 0.5, the scored ROC curve will simply resort to the random performance, which in itself, is uninteresting. The value of using the scored ROC curve is more apparent when the learning model performs better than random but less than perfect.

In our experiments, we have observed both of these conditions. In particular, models constructed from the `mushroom` and the `Haberman` data sets produce these two conditions respectively. Perfect performance, on the one hand, does not leave room for score assessment analysis because perfect scores of zero and one are assigned to the negatives and the positives respectively. In this case, the model does not produce inappropriate scores. On the other hand, random performance results with an almost equal proportions of appropriate and inappropriate scores. Thus, our method becomes unable to provide useful performance information. Therefore, it is recommended that the scored ROC curve be used when the model performs reasonably well and produces scores between 0 and 1.

6.2 Future work

This dissertation identifies a gap in the performance analysis of learning methods. The class membership scores, produced by the model, are the center of this evaluation process. As a remedy, this research develops a novel evaluation measure that incorporates such scores

into the ROC curve. Effectively, our contribution sheds a light on several issues worthy of further investigations. They are focused in two main areas; the methodological design of our method and the performance analysis of learning models. While the former relates to how the scored ROC curve is constructed and interpreted, the latter focuses on investigations that utilize our method. Both are discussed in details in the following sections.

6.2.1 Methodological design

The discussion in this section is related to several topics including the calculation of the midpoint of score values, the possibility of varying this midpoint value to determine the confidence of the scored ROC curve, and the utilization of score appropriateness as an indicator function for performance analysis rather than using the score magnitudes themselves.

The construction of the scored ROC curve relies on computing a midpoint of class membership score values to determine their appropriateness. This establishes the basis on which our method determines the "correct" score magnitude to be visualized on the curve. This midpoint value can be viewed as a property associated with the data set itself and relates to how the two classes overlap in the domain. Estimating this midpoint during training can potentially lead to over-fitting the training sample. Therefore, we call for estimating this midpoint from a validation set and applying it during the model testing phase. In our experiments, this midpoint is measured from the test data.

However, calculating this midpoint has limitations and deficiencies. Ideally, the midpoint should be estimated directly from the domain. Like the notion of labeling instances to indicate the correct classification outcome in supervised learning, the midpoint value depicts the appropriateness of score assignment. When estimated accurately, the performance can be measured more precisely. Obviously, if its exact value is given, the estimation issue can be put to rest. In real-life applications unfortunately, such knowledge is likely to be unavailable. Therefore, we need to explore methods of how to determine this value. In other fields, including economics, operation research, psychology, and medicine, several models have been developed to produce such a threshold value. It may be feasible to consider

some of their methods in addressing this issue. Alternatively, calibrated scores have been shown to be more valuable, and they may be desired in some applications (Cohen and Goldszmidt, 2004). When the scores are calibrated, it is intuitive to use the 0.5 as a midpoint value. Thus, calibrating the scores avoids the estimation of this midpoint all together. Furthermore, score calibration produces a convex scored ROC curve, which can be useful (Fawcett and Niculescu-Mizil, 2007). To this extent, several methods have been proposed to calibrate the scores (Zadrozny and Elkan, 2002; Niculescu-Mizil and Caruana, 2005; Caruana and Niculescu-Mizil, 2006).

Another issue concerning this midpoint value deals with varying its value over the $[0, 1]$ interval in a manner similar to the classification threshold used to generate the standard ROC curve. Class membership scores are assumed to be in the $[0, 1]$ interval. Therefore, considering all possible midpoint values generates multiple scored ROC curves which may form a band of curves. Such bands can depict the confidence of the learning model particularly when combined with data sampling and curve averaging techniques. This approach can be viewed as a measure of confidence in the scored ROC space. Similar techniques have been applied to the ROC curve (Macskassy and Provost, 2004; Macskassy et al., 2005a,b). Potentially, such studies are useful for determining a measure of confidence for our method.

Along the same line of thought, Chapter 4 presents methods to incorporate class membership scores in a single direction of the ROC space. We can show that setting this midpoint value to one extreme can produce the exact same effect produced by these single-direction algorithms. Therefore, using such settings may provide an upper (or lower) bound on the learning performance which can be relevant to measuring the confidence.

Our method incorporates class membership scores into the ROC curve, so that appropriate score magnitudes are depicted along the Y-axis and inappropriate score magnitudes are shown on the x-axis. Instead of using the score magnitudes themselves, an indicator function can be used to depict the appropriateness of scores only. Thus, this new curve will depict the performance of the learning model based on the appropriateness its score assignment rather than their magnitudes. One can think of this as a method to visualize

the trade off between how often the learning model assigns appropriate versus inappropriate scores to instances. This method may provide a useful evaluation measure of a lesser granularity of information compared to the scored ROC curve developed in this thesis.

Finally, researchers in the field of forecasting have developed extensive and comprehensive evaluation methods which can be used to assess probability estimation based on several characteristics. For instance, Murphy (1993) lists several characteristics and measures which describe a "good" weather forecast. Borrowing and customizing such criteria to suit the needs of machine learning systems will benefit the performance assessment task.

6.2.2 Performance analysis

In the context of performance analysis, there may be several studies of interest. These future studies include but are not limited to comparing the area under the scored ROC curve to the scored AUC presented by Wu et al. (2007), analyzing the characteristics of the area under the scored ROC curve using the discriminancy measure (Ling et al., 2003b; ?), and investigating the value, nature, and conditions of imposing partial assumptions on the learning model or on the domain. In addition, it is important to examine characteristic of well-established learning algorithms based on their scoring behavior, analyze the effect of various data sampling methods on the performance of scoring tasks, and assess the impact of the our evaluation method on evaluating models in real-life changing environments.

In Chapter 2, we reviewed the work of Wu et al. (2007) which calculates the scored AUC without having to construct the scored ROC curve itself. This research constructs this scored ROC curve. Therefore, a comparison between the area under our curve to their sAUC is important to establish a formal link. Furthermore, Ling et al. (2003b); ? compare performance measures based on two criteria, consistency and discriminancy. The latter indicates how discriminant a performance measure is. The area under the scored ROC curve can be related to their measure of discriminancy. The reason lies in that the area under the scored ROC curve is maximized when the scores are most discriminant between classes. Therefore, we plan to compare the discriminancy of the standard AUC, our AUC,

and the scored AUC developed by Wu et al. (2007) to reinforce our results.

With respect to information content, our categorization of learning tasks positions the scored ROC curve to suit the evaluation of scores produced by the learning model. In this thesis, we argue that assessing the scores as though they are probabilities imposes assumptions and may not be necessary. However, when some assumptions can be made (in the presence of additional information), a natural question would be: what and how can we exploit such added information? Our method avoids making assumptions but if additional knowledge is available, what and how can we exploit such knowledge? In a sense, if we knew that the scoring function exhibits some systematic properties, the question is: how can we exploit them to enhance the performance analysis? Investigating these issues will lead to the development of evaluation measures capable of dealing with various levels of uncertainties in scoring tasks.

From an experimental perspective, it may be valuable to conduct extended experiments aimed at examining various scoring characteristics of well-established machine learning algorithms. With our method in hand, we are able to observe and conclude systematic behavior of models which produce class membership scores. For instance, we now are able to study properties of scores produced by a neural network and compare them to other models as well. The same argument applies to studying various characteristics of data sets drawn from changing distributions. This thesis illustrates that our proposed method is sensitive to changes in data distribution. An empirical study of how various machine learning models behave in changing environments may yield interesting results.

Our novel method can be used to compare scores in medical domains. For example, various scoring methods being used in emergency departments as part of medical practice guidelines. These scoring scales are developed by conducting costly and lengthy clinical trials. If we are able to compare patient diagnostic scores produced by such guidelines to those produced by machine learning systems, we will be able to demonstrate the value of utilizing machine learning algorithms in such domains. Machine learning models require significantly less time and are less costly to develop. If we are able to show that they can

produce effective scoring models, practitioners will take notice. Our long term objective is to enhance the evaluation of machine learning systems so that we are able to demonstrate and document their effectiveness to persuade practitioners to utilize them.

Based on this research, and along with future research investigations, we anticipate to publish several articles. The scored ROC method is a novel method worthy of publication for the machine learning community. Its effectiveness in analyzing the performance of learning models and in changing environment is also important form the point of view of publication. The comparisons of the various performance measures based on the area under the various curves, studied in this dissertation, can provide significant insights for the machine learning community and are potential subjects of publications. Finally, the use of our method to assess the quality of data sets will, most certainly, capture interest in the community.

6.3 A final note

This dissertation has utilized a comparative categorization of common machine learning tasks based on their information content. It has demonstrated a need for an evaluation method capable of dealing with class membership scores so that researchers are able to visualize and compare score values produced by the learning model. This research has developed such a method which extends the Receiver Operating Characteristics (ROC) curve to include class membership scores. This thesis has shown the ability of this novel method, the scored ROC curve, to capture similarities and differences in the performance of different learning models, and to detect changes in the data distributions. Although the scored ROC curve remains in its infancy stage, it makes a fundamental step towards developing a comprehensive evaluation measure which demonstrates the value of machine learning methods effectively. We expect that future research, along with existing work, will expand and refine our approach to address many issues brought forward by this dissertation.

Bibliography

N. Adams and D. J. Hand. Comparing classifiers when the misallocations costs are uncertain. *Pattern Recognition*, 32:1139–1147, 1999.

A. Asuncion and D. J. Newman. UCI machine learning repository. `http://www.ics.uci.edu/~mlearn/MLRepository.html`, 2007.

M. F. Balcan, N. Bansal, A. Beygelzimer, D. Coppersmith, J. Langford, and G. B. Sorkin. Robust reductions from ranking to classification. In *proceedings of the 20th Annual Conference on Learning Theory (COLT)*, pages 604–619, 2007.

E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36:105–142, 1999

P. N. Bennett. Assessing the calibration of naive bayes' posterior estimates. Technical Report No. CMU-CS00-155, 2000.

P. N. Bennett. Using asymmetric distributions to improve text classifier probability estimates. In *proceedings of he 26th Annual International Conference on Research and Development in Information Retrieval, ACM SIGIR 2003*, pages 111–118, 2003.

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.

L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

D. Brenner and E. Hall. Computed tomography an increasing source of radiation exposure. *The New England Journal of Medicine*, 357(22):2277–2284, 2007.

G. Brier. Verification of forecasts expressed in terms of probabilities. *Monthly Weather Review*, 78:1–3, 1950.

T. Calders and S. Jaroszewicz. Efficient auc optimization for classification. In *proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases PKDD 2007*, pages 42–53, 2007.

R. Caruana and A. Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *proceedings of the 23rd International Conference on Machine Learning ICML 2006*, pages 625–632, 2006.

K J Cios and G W Moore. Uniqueness of medical data mining. *AI in medicine*, 26(1-2): 1–24, 2002.

I. Cohen and M. Goldszmidt Properties and benefits of calibrated classifiers. In *Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 125–136, 2004.

C. Cortes and M. Mohri. Auc optimization vs. error rate minimization. In Sebastian Thrun, Lawrence Saul, and Bernhard Scholkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, 2004.

C. Cortes, M. Mohri, and A. Rastogi. Magnitude-preserving ranking algorithms. In *proceedings of the 24th international conference on Machine learning*, pages 169–176, 2007.

M. DeGroot and S. Fienberg. The comparison and evalution of forecasters. *The statistician*, 32:12–22, 1983.

P. Domingos. Metacost: A general method for making classifiers cost-sensitive. In *proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining KDD 1999*, pages 155–164, 1999.

P. Domingos and M. Pazzani. Beyond independence: Conditions for the optimality of the simple bayesian classifier. *Machine Learning*, 29:103–130, 1997.

C. Drummond and R. C. Holte. Cost curves: An improved method for visualizing classifier performance. *Machine Learning*, 65:95–130, 2006.

C. Drummond, N. Japkowicz, and W. Elazmeh. The second workshop on evaluation methods for machine learning, AAAI 2006, 2006. `Http://www.site.uottawa.ca/ICML06WS/`.

C. Drummond, N. Japkowicz, W. Elazmeh, and S. Macskassy. The second workshop on evaluation methods for machine learning, AAAI 2007, 2007. `Http://www.site.uottawa.ca/ICML07WS/`.

C. Drummond, N. Japkowicz, W. Klement, and S. Macskassy. The third workshop on evaluation methods for machine learning, ICML 2008, 2008. `Http://www.site.uottawa.ca/ICML08WS/`.

C. Drummond, N. Japkowicz, W. Klement, and S. Macskassy. The fourth workshop on evaluation methods for machine learning, ICML 2009, 2009. `Http://www.site.uottawa.ca/ICML09WS/`.

J. Egan. *Signal Detection Theory and ROC Analysis.* Series in Cognition and Perception. New York: Academic Press, 1975.

T. Fawcett. Using rule sets to maximize ROC peformance. In *proceedings of the IEEE International Conference on Data Mining ICDM 2001*, pages 131–138, 2001. IEEE Computer Society.

T Fawcett. ROC graphs: Notes and practical considerations for data mining researchers, 2003. Technical Report HPL-2003-4, HP Labs.

T. Fawcett and A. Niculescu-Mizil. Pav and the ROC convex hull. *Machine Learning.* 68 (1):97–106, 2007.

C. Ferri, P. Flach, and J. Hernandez-Orallo. Improving the auc of probabilistic estimation trees. In *proceedings of the 14th European Conference on Machine Learning ECML 2003*, pages 121–132, 2003.

M. Fisher, J. E. Fieldsend, and R. M. Everson. Precision and recall optimisation for information access tasks. In *proceedings of the 1st workshop on ROC Analysis in AI. ROCAI 2004, ECAI 2004*, pages 45–54, 2004.

P. Flach. The geometry of roc space: Understanding machine learning metrics through roc iso-metrics. In *proceedings of the 20th International Conference on Machine Learning ICML 2003*, pages 194–201, 2003.

P. Flach. The many faces of roc analysis in machine learning. ICML 2004 Tutprial, 2004. `http://www.ke.informatik.tu-darmstadt.de/events/ICML-04/tutorials.html`.

P. Flach and E. T. Matsubara. Obtaining calibrated probability estimates from simple lexicographic rankings. In *the 18th European Conference on Machine Learning ECML 2007*, pages 575–582, 2007.

P. Flach and S. Wu. Repairing concavities in roc curves. In *proceedings of UK Workshop on Computational Intelligence*, pages 38–44, 2003.

P. Flach and S. Wu. Repairing concavities in roc curves. In *proceedings of the 19th International Joint Conference on Artificial Intelligence IJCAI 2005*, pages 702–707, 2005.

P. A. Flach. Putting things in order: On the fundamental role of ranking in classification and probability estimation. Invited talk, the 18th European Conference on Machine Learning and 11th European Conference on Principles and Practice of Knowledge Discovery in Databases ECML/PKDD 2007, 2007. ISBN 978-3-540-74975-2, pp. 23.

G. Forman. A method for discovering the significance of one's best classifier and the unlearnability of a classification task. In *proceedings of the 1st International Workshop on Data Mining Lessons Learned DMLL 2002*, 2002.

G. Forman. Counting positives accurately despite inaccurate classification. In *proceedings of the 16th European Conference on Machine Learning, ECML 2005*, pages 564–575, 2005.

J. Frünkranz and P. Flach. ROC'n' rule learning – towards a better understanding of covering algorithms. *Machine Learning*, 58:39–77, 2005.

J. L. Gastwirth. The estimation of the lorenz curve and gini index. *Machine Learning*, 54 (3):306–316, 1972.

D. Grossman and P. Domingos. Learning bayesian network classifiers by maximizing conditional likelihood. In *Proceedings of the Twenty-First International Conference on Machine Learning ICML'04*, pages 361–368, 2004.

D. J. Hand. Classifier technology and the illusion of progress. *Statistical Science*, 21(1): 1–15, 2006.

J. A. Hanley and B. J. McNeil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143(11):29–36, 1982.

E. Hsieh and B. W. Turnbull. Nonparametric and semiparametric estimation of the receiver operating characteristics curve. *Annals of Statistics*, 24:25–40, 1996.

L. Jiang and H. Zhang. Learning naive bayes for probability estimation by feature selection. In *proceedings of the Nineteenth Canadian Conference on Artificial Intelligence. CAI'06.*, pages 503–514, 2006.

M. G. Kelly, D. J. Hand, and N. M. Adams. The impact of changing populations on classifier performance. In *proceedings of ACM SIGKDD*, pages 367–371, 1999.

I. Kononenko and I. Bratko. Information-based evaluation criterion for classifier's performance. *Machine Learning*, 6:67–80, 1991.

T. Lane and C. E. Brodley. Approaches to online learning and concept drift for user identification in computer security. *KDD*, pages 259–263, 1998.

P. Langely. Machine learning as an experimental science. *Machine Learning*, 3:5–9, 1988.

N. Lavrac, P. Flach, and B. Zupan. Performance: Rule evaluation measures: A unifying view. In *proceedings proceedings of the 9th International Workshop on Inductive Logic Programming ILP 1999*, pages 174–185, 1999.

E. Law. The problem of accuracy as an evaluation criterion. In *proceedings of the 3rd workshop on Evaluation Methods for Machine Learning ICML 2008*, 2008.

D. Lewis. Representation quality in text classification: An introduction and experiment. In *proceedings of the Workshop on Speech and Natural Language*, pages 288–295. Morgan Kaufman, 1990.

D. Lewis. Evaluating text categorization. In *proceedings of the Workshop on Speech and Natural Language*, pages 312–318. Morgan Kaufman, 1991.

C. X. Ling, J. Huang, and H. Zhang. Auc: a statistically consistent and more discriminating measure than accuracy. In *proceedings of 18th International Conference on Artificial Intelligence (IJCAI-2003)*, pages 512–526, 2003a.

C. X. Ling, J. Huang, and H. Zhang. Auc· A better measure than accuracy in comparing learning algorithms. In *proceedings of Canadian Conference on artificial Intelligence, CAI'03*. pages 329–341, 2003b.

S A. Macskassy and F. Provost. Confidence bands for ROC cui ves: Methods and an empirical study. In *proceedings of the 1st Workshop on ROC Analysis in Artificial Intelligence ROCAI2004*, 2004.

S A. Macskassy, F. Provost, and S Rosset. Roc confidence bands: An empirical evaluation. In *proceedings of the 22nd International Conference on Machine Learning ICML 2005*, pages 537–544, 2005a.

S. A. Macskassy, F. Provost, and S. Rosset. Point-wise ROC confidence bounds. An empiri-

cal evaluation. In *proceedings of the 2nd workshop on ROC Analysis in Machine Learning ROC at ICML 2005*, 2005b.

D. D. Margineantu and T. G. Dietterich. Improved class probability estimates from decision tree models. *in D. D. Denison, M. H. Hansen, C. C. Holmes, B. Mallick, and B. Yu (Eds.) Nonlinear Estimation and Classification; Lecture Notes in Statistics*, 171:169–184, 2002.

A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. The det curve in assessment of detection task performance. In *proceedings of Eurospeech'97*, pages 1895–1898, 1997.

T. M. Mitchell. *Machine Learning.* WCB McGraw-Hill, 1997.

H. Motulsky. *Intuitive Biostatistics.* Oxford University Press, 1995.

A. H. Murphy. What is a good forecast? an essay on the nature of goodness in weather forecasting. *Weather and Forecasting*, 8(2):281–293, 1993.

A. Narasimhamurthy and L. I. Kuncheva. A framework for generating data to simulate changing environments. In *proceedings of the International Conference on Artificial Intelligence and Applications IASTED*, pages 384–389, 2007.

A. Niculescu-Mizil and R. Caruana. Predicting good probabilities with supervised learning. In *proceedings of the 22nd International Conference on Machine Learning ICML 2005*, pages 625–632, 2005.

J. Platt. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. *Advances in Large Margin Classifiers*, pages 61–74, 1999.

F. Provost and P. Domingos. Tree induction for probability-based ranking. *Machine Learning*, 52:199–215, 2003.

F. Provost and T. Fawcett. Robust classification systems for imprecise environments. *Machine Learning*, 42(3):203–231, 2001.

F. Provost, T. Fawcett, and R. Kohavi. Analysis and visualization of classifier performance: Comparison under imprecise class and cost distribution. In *proceedings of 3rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining KDD 1997*, pages 43–48, 1997.

F. J. Provost, T. Fawcett, and R. Kohavi. The case against accuracy estimation for comparing induction algorithms. In *proceeding of the International Conference on Machine Learning ICML 1998*, pages 445–453, 1998.

J. R. Quinlan. *C4.5: Programs for Empirical Learning.* 2nd Edition, Morgan Kaufmann, San Francisco, CA, 1993. San Francisco, CA.

J. D. M. Rennie, L. Shih, J. Teevan, and D. R. Karger. Tackling the poor assumptions of naive bayes text classifiers. In *the Twentieth International Conference on Machine Learning ICML'03*, 2003.

C. J. Van Rijsbergen. *Information Retrieval.* Butterworths, 1979.

F Seifi. Heart attack data set. Internal Report., 2009. Ottawa-Carleton School of Computer Science, University of Ottawa.

P Smyth, A. Gray, and U. Fayyad Retrofitting decision tree classifiers using kernel density estimation. In *proceedings of the 12th International Conference on Machine Learning ICML 1995*, pages 506–514, 1995.

K. A. Spackman. Signal detection theory: Valuable tools for evaluating inductive learning. In *proceedings of the 6th International Workshop on Machine Learning*, pages 160–163, 1989.

J. Swets. Measuring the accuracy of diagnostic systems. *Science*, 240:1285–1293, 1988.

J. A. Swets, R. M. Dawes. and J. Monahan. Better decisions through science. *Scientific American*, 283:82–87, 2000.

G. Widmer and M. Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23:69–101, 1996.

G. Widmer and M. Kubat. Special issue on context sensitivity and concept drift. *Machine Learning*, 32:101–126, 1998.

I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques.* Morgan Kaufmann, 2005. 2nd Edition.

S. Wu, P. Flach, and C. Ferri. An improved model selection heuristic for auc. In *the 18th European Conference on Machine Learning ECML 2007*, pages 478–489, 2007.

B. Zadrozny and C. Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *proceedings of the 18th International Conference on Machine Learning ICML 2001*, pages 906–616, 2001.

B. Zadrozny and C. Elkan. Transforming classifier scores into accurate multi-class probability estimates. In *proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining KDD 2002*, pages 694–699, 2002.

H. Zhang. The optimality of naive bayes. In *proceedings of the 17^{th} International FLAIRS conference (FLAIRS2004)*, 2004.

H. Zhang and J. Su. Naive bayesian classifiers for ranking. In *Proceedings of the 15th European Conference on Machine Learning (ECML2004)*, 2004.

H. Zhang and j. Su. Learning probability decision trees for AUC. *Pattern Recognition Letters*, 27:892–899, 2006.

X. H. Zhou, N. A. Obuchowski, and D. K. McClish. *Statistical Methods in Diagnostic Medicine.* John Wiley and Sons, Chichester, 2002.

K. H. Zou. Receiver operating characteristics (ROC) literature research. On-line bibliography available at: `http://splweb.bwh.harvard.edu:8000/pages/ppl/zou/roc.html`, 2002.