3-28-2018

Evaluating Classifiers' Optimal Performances Over a Range of Misclassification Costs by Using Cost-Sensitive Classification

Ramy Al-Saffar
Louisiana State University and Agricultural and Mechanical College, ramyalsaffar@yahoo.com

EVALUATING CLASSIFIERS' OPTIMAL PERFORMANCES OVER A RANGE OF MISCLASSIFICATION COSTS BY USING COST-SENSITIVE CLASSIFICATION

A Thesis

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Master of Science in Computer Science

in

The Division of Computer Science and Engineering

by
Ramy Al-Saffar
B.S., University of Technology in Baghdad, Iraq, 2008
May 2018

ACKNOWLEDGMENTS

I wish to present my sincere thanks to Dr. Triantaphyllou for his valuable and constructive suggestions during the planning and development of this research work. His willingness to give his time so generously has been the main factor to succeed during my study in Louisiana State University.

This work is dedicated to my late father, mom, and brothers who always stood behind me and encouraged me. Also, my profound gratitude goes to Saraa Jawad for all her love and support. This accomplishment would not be possible without all of them. Thank you.

TABLE OF CONTENTS

ACKNOWLEDGMENTS....ii

ABBREVIATIONS....v

ABSTRACT....vi

1. INTRODUCTION....1

2. LITERATURE REVIEW....5
   2.1 The Unweighted Accuracy....5
   2.2 The Receiver Operational Characteristic (ROC) Graph....6
   2.3 The Area Under the Curve (AUC)....11
   2.4 Instances' Misclassification Costs....12

3. FORMAL PROBLEM DESCRIPTION....13
   3.1 Problem Definition....13
   3.2 The Confusion Matrix....14
   3.3 The Cost Matrix....15
   3.4 Additional Formulas....15

4. PROPOSED SOLUTION APPROACH....17
   4.1 Fundamental Notation....18
   4.2 The Minimum Total Misclassification Cost for a Single Cost Ratio (MTMCS)....19
   4.3 Finding the Optimal Classification Threshold....26
   4.4 The Optimal Classification Thresholds Chart....30
   4.5 The Cost-Sensitive Classification Algorithm....30

5. RESULTS AND DISCUSSION....38
   5.1 Robust vs Optimal Classifier....38
   5.2 The First Experiment: German Credit Dataset....39
   5.3 The Second Experiment: Credit Approval Dataset....42
   5.4 The Third Experiment: Messidor Features (Diabetic Retinopathy Debrecen) Dataset...44
   5.5 The Fourth Experiment: SPECTF Heart Dataset....47
   5.6 The Experiments Summary....49
   5.7 The Use of the CST Measure to Compare the Overlapping Cost Curves....53
   5.8 Discussion....57

6. CONCLUSIONS....60

REFERENCES....63

VITA....64

ABBREVIATIONS

- TN: True Negative Instances
- FP: False Positive Instances
- FN: False Negative Instances
- TP: True Positive Instances
- FNcost: False Negative Cost
- FPcost: False Positive Cost
- FNRate: False Negative Rate
- FPRate: False Positive Rate
- ROC: Receiver Operational Characteristic
- AUC: Area Under the Curve
- TMC: Total Misclassification Cost
- MTMCS: Minimum Total Misclassification Cost for a Single Cost Ratio
- MTMCR: Minimum Total Misclassification Cost of a Range of Cost Ratios
- CST: Cost and Sensitivity Tradeoff

ABSTRACT

We believe that using the classification accuracy is not enough to evaluate the performances of classification algorithms. It can be misleading due to overlooking an important element which is the cost if classification is inaccurate. Furthermore, the Receiver Operational Characteristic (ROC) is one of the most popular graphs used to evaluate classifiers performances. However, one of the biggest ROC's shortcomings is the assumption of equal costs for all misclassified data. Therefore, our goal is to reduce the total cost of decision making by selecting the classifier that has the least total misclassification cost. Nevertheless, the exact misclassification cost is usually unknown and hard to determine. To overcome such hurdle, we classify the data against a range of error costs. Thus, we use the cost range and the operating classification threshold range to show any performance differences among classifiers.

1. INTRODUCTION

Along the rapid increase in data storage capacity at the end of the twentieth century, a huge amount of data accumulated massively. As a result, a need rose to analyze this huge amount of accumulating data to turn it into useful information. At the current time, many companies and institutions have big amounts of data and they hire data scientists to analyze and classify data by using software. Data scientists are trying to build models and derive hidden patterns from data by understanding the nature of the system or phenomenon that generated these data. The data analysis process is taking place in many disciplines such as:

- Money laundry detection, where a software application can detect money transformed multiple times with the intention to make money gained from criminal activities look like legitimate assets.
- Credit card fraud detection, where a system classifies customer transactions as legitimate or fraudulent to protect the credit card company itself and its clients. Furthermore, credit card companies analyze certain attributes of an applicant to find out whether this applicant can be a trustworthy customer or not and grant him/her a credit card if the application has low risk.
- Spam email detection, where a system tries to find out whether an email should be placed in the inbox or the junk folder, to reduce clutter and decrease the risk of email hacking.
- Cancer cell detection, where data analysis can help to differentiate between malignant and benign cells.
- Emergency room (ER), where a hospital can decide whether to place a patient in the intensive care unit (ICU) or not after doing some tests to acquire data that can be fed to a computational system to help making the right decision.

- Market segmentation, where a company tries to predict whether a customer is likely to respond to a promotion via postal mail or email.

Unfortunately, in data science regardless of the plethora of classification algorithms, there is no algorithm that can classify all in hand or prospective data accurately. Each one of these algorithms works better and delivers a higher percentage of accurately classified instances than its peers in a certain case scenario but not in all. Therefore, data scientists tend to test many classification algorithms for each problem using some criteria to measure how each classification algorithm performs. Then, they can decide the one with the best outcome according to the criteria used.

There are many criteria used to evaluate the performance of classification algorithms. However, each one of these criteria has pros and cons due to the assumptions they rely on. Usually, these measures greatly rely on one overstated concept according to which the more instances are correctly classified the better the performance of the classifier is. Yet, this high percentage of correctly classified instances by itself only makes sense to the data scientist but not to the client. Clients who hire data scientists to analyze their data expect that the results provided will be highly realistic and accurate. They only care about how to improve his/her business decision-making process to maximize profit and minimize cost. Therefore, many scientists try to overcome these overstated assumptions and reach a more realistic result which clients are seeking.

The criterion we are concerned about in this thesis is the cost due to misclassified data. Some of the widely used methods to evaluate the classifiers' performances assume equal costs for all misclassified instances which rarely happens. In life, the consequences of making errors in each case scenario have a different impact than others, such as:

- Intuitively, the cost of a legit transaction which is considered fraudulent has a different cost or consequence than if a fraudulent transaction is considered as legit. The first case

may yield a delay and inconvenience to the customer, while in the latter case, it may yield money losses for the company. Moreover, refusing a fraudulent transaction has a nontrivial benefit because it may prevent further fraud and lead to the arrest of a criminal. [7]

- In a credit card company, rejecting a trustworthy customer is not the same as accepting and granting a credit card for an untrustworthy customer. In the first case, the company is losing the benefits resulted when having a good customer, while in the latter case, it may result in financial losses to the company besides any legal consequences.
- Placing a normal email in the junk folder has a different impact than placing a spam email in the inbox folder. In the first case, the user may miss an important email, while in the latter case, the user may click on a potentially dangerous link in the spam email which may lead to email account hacking.
- Regarding the cancer cell detection, considering a benign cell as a malignant has a different impact than considering a malignant cell as a benign. In the first case, it will lead to further unnecessary medical interventions which can vary from patient inconveniency to large bills, while in the latter case, it is endangering the patient's life!
- Relating to the ER, if a patient who needs to enter the ICU is placed instead in a regular room, this may endanger his/her life. On the other hand, if a patient who does not need to be assigned to the ICU gets assigned to one, this will cause a large unnecessary expense to the patient.
- With respect to the market segmentation, targeting the wrong audience means a waste of time and money. Furthermore, it means a loss of prospective customers that have not been targeted.

We can see from these examples, that the consequences of making errors in each scenario have a different ramification than the others. As a matter of fact, in one scenario each error

may have a different consequence than the others. Unfortunately, we do not have the metadata of the exact severity of each misclassified instance. Even so, data scientists can deal with this problem and estimate the cost of misclassified instances by forming a range of costs to capture uncertainty. Therefore, in this thesis, we will show the importance of incorporating cost as a major classifiers' evaluation criterion to reach a more realistic result. Since different errors are associated with different costs, we will show the cost consequences of misclassification on a chart. This cost chart will show how classifiers' performances differ with various error costs. As a result, this cost chart can help data scientists to select the best classifier under the cost range of interest.

In summary, the motivation for this thesis is as follows:

- Usually, errors of misclassification are accompanied by cost, which if we consider as a classification performance measure then we will reach a more realistic result for the client.
- Helping data scientists decide which classifier can produce the best realistic overall result and under what circumstances.
- Tackling problems where the cost of misclassification is not known or hard to determine. Since we usually do not know the exact cost of the misclassification, we need to capture this uncertainty. In other words, we need a method to show classifiers' performances along with different costs of misclassification.
- Measure the classifiers' performances based on quantitative analysis.
- Show the classifiers' sensitivity to cost changes.

2. LITERATURE REVIEW

There are many classification algorithms in the realm of Data Mining. For each one of them, there are points of strength but also shortcomings. Thus, it is the job of the data scientist to select the proper classifier for the task at hand. As a result, many mathematical formulas and charts exist which are used to help decide the best classifier compared to others in a specific case.

2.1 The Unweighted Accuracy

Accuracy is the percentage of the correctly classified instances (of the positive and negative classes) over the total number of instances. It is also known as the Hit Rate. Accuracy is one of the most basic formulas used to evaluate the classifiers' performances. Furthermore, it is widely used in the literature and in practice. Unweighted accuracy is used as a classification evaluation measure by data scientists who think the higher the accuracy of classification is, the better the performance of the classifier is. This approach may overlook the consequences of misclassification errors! Unweighted accuracy does not take into consideration the cost of the false negative and the false positive classified instances. In other words, the accuracy formula assumes equal misclassification cost for all instances. Nonetheless, we cannot reach 100% accuracy because we usually have noise in data or the classifier cannot be perfect all the time. Moreover, if this happens and we reached full accuracy with the training data then we have the overfitting problem. This problem takes place when the model is so complicated and can fit all the training data even noise, but it is not accurate with new unclassified data.

In addition, in extremely unbalanced datasets where instances of a minority class occur rarely, classifiers will show high classification accuracy. This happens simply because classifiers are designed to increase the accuracy and tend to predict the majority class more and overlook the minority class [2]. "As the class distribution becomes more skewed, evaluation based on accuracy breaks down." [4] Such result is inaccurate and misleading because it does

not address the main goal of correctly classifying the most important instances. A typical example of such scenario is when detecting fraudulent transactions which represent a small percentage of the total number of all transactions. Thus, accuracy makes sense and can be considered as a useful evaluation measure only when we care about the percentage of the correctly classified instances. Also, when accuracy is applied to a balanced data distribution.

2.2 The Receiver Operational Characteristic (ROC) Graph

By the same token, one of the most used and well-known graphs to compare classifiers' performances is the ROC. The use of the ROC started in World War II. The ROC is a two-dimensional graph where the Y-axis represents the Sensitivity (true positive rate) and the X-axis represents the 1-Specificity (false positive rate). The ROC curve is the relation between them. Sensitivity is the classifier's ability to identify the true positive instances from the actual positive instances. Whilst Specificity is the classifier's ability to identify the true negative instances. Therefore, points on the ROC curve that are closest to the northwest corner of the graph, are considered better because they have higher hit rate. Furthermore, points in the ROC curves are generated by using different classification thresholds through a range of [0, 1] on a given dataset.

The strength of the ROC lies in its insensitivity to data distribution changes, which means if the skewness of the data changes then the ROC curve will not be affected. The explanation of this powerful phenomenon comes from its reliance on one side of the confusion matrix which is shown in Table 1. [6]

Table 1. This table shows a preliminary confusion matrix for two-class cases. The third column represents the values that Precision relies on. The second row represents the values that 1-Specificity relies on. The third row represents the values that Recall (Sensitivity) relies on.

	Predicted as Negative Class	Predicted as Positive Class
Actual Negative Class	True Negative (TN)	False Positive (FP)
Actual Positive Class	False Negative (FN)	True Positive (TP)

Hence, the decision of selecting an optimal classifier may change when the data distribution shifts if this decision is made based upon a measurement that is sensitive to data distribution.

We are using the four graphs in Figures 1 and 2 to show the strength of the ROC towards data distribution changes. These four graphs have been published in [6]. The two graphs in Figure 1 represent the first scenario where the two classes are distributed equally (balanced dataset). The left chart is the ROC graph and the right is the Precision*Recall curves of the same dataset. Recall definition is the same as Sensitivity, while Precision is the classifier's ability to identify the true positive instances from the predicted positive instances.

The two graphs in Figure 2 represent the second scenario where the two classes distributions are shifted by the ratio 1 to 10. We can see that the ROC graph on the left-hand side of Figure 2 still has the same appearance as the one above it in Figure 1 of equal data distribution. The ROC is not affected by the changes to the class distribution. However, the Precision*Recall graph on the right-hand side of Figure 2 is affected by the shifted distribution after changing the class ratios.

The explanation of this phenomenon is as follows: Sensitivity (Recall) relies on the actual positive class, and 1-Specificity relies on the actual negative class. Whereas Precision relies on both the actual negative and the actual positive classes. As a result, a measurement that relies only on either the actual positive or the actual negative class is insensitive to the class distribution changes.

On the other hand, one of the shortcomings of the ROC is that it assumes equal costs for both false negative and false positive classified instances. Still, it is biased towards smaller false negative (higher true positive ratio) because its Y-axis represents the true positive rate. Therefore, a classifier with less count of false negatives will be considered to have a better performance. At first glance, it is a good result and exactly what we want. However, since the

ROC uses the assumption of equal costs for all misclassified instances, its result cannot be generalized to all variations of misclassification costs.

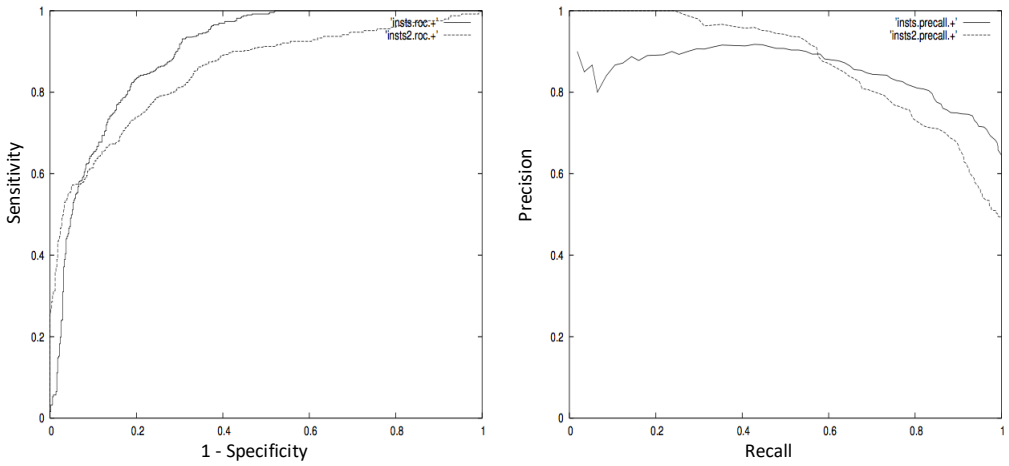

Figure 1. These two graphs have been generated by using the same balanced dataset. The left graph is the ROC and the right one is the Precision*Recall graph. The two curves in each graph represent the performance of two classifiers. [6]

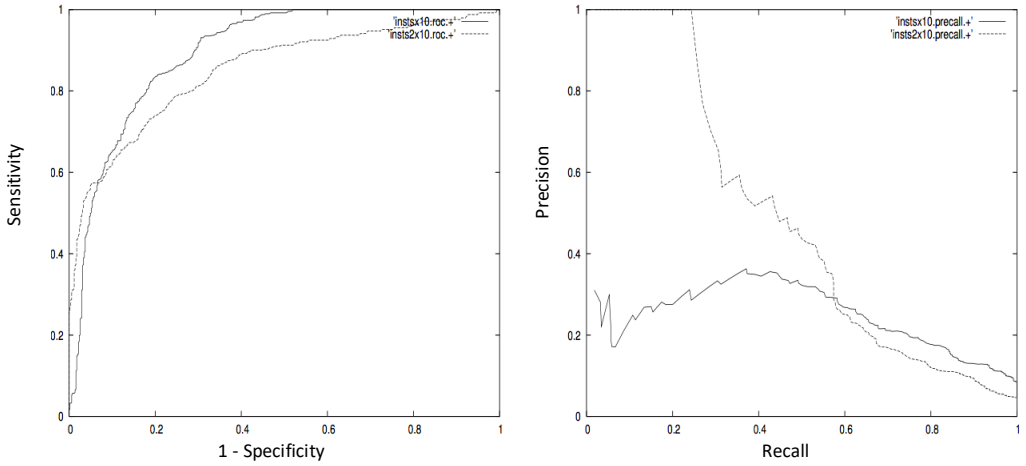

Figure 2. These two graphs have been generated by using the same unbalanced dataset. The left graph is the ROC and the right one is the Precision*Recall graph. The two curves in each graph represent the performance of two classifiers. [6]

Therefore, many researchers tried to improve and enhance the performance of the ROC approach itself. Provost and Fawcett tried to improve the ROC and use the Convex Hull (ROCCH) idea to create a robust classifier and also use the ISO performance lines to represent cost [4]. However, the ROCCH may lead to overfitting since it is a composite of two or more classifiers, and the ISO performance lines are difficult to read and interpret. In spite of this difficulty, the use of the ISO performance lines helps to test the cost sensitivity of classifiers

to changes of the cost ratio. It shows whether the cost sensitivity is high or low but it cannot tell by how much. To decrease the clutter, it is preferred to use only two ISO performance lines to show the start and the end of the cost ratio range of interest. The slope of the ISO performance line is calculated as follows [4]:

$$ISO\ performance\ line\ slope = \frac{Cost\ of\ False\ Positive}{Cost\ of\ False\ Negative}$$

where cost of false positive and false negative > 0.

Therefore, every time the cost ratio of misclassification changes, the ISO performance lines need to be recalculated in order to make its slope show the new cost ratio. Conceptual Figures 3 and 4 show both scenarios of low and high-cost sensitivity using the ISO performance lines. Both conceptual Figures 3 and 4 ideas are derived from [4].

In conceptual Figure 3, we show a classifier that has low-cost sensitivity to cost ratio changes using the ISO performance lines. The two ISO performance lines (the solid and dashed lines) represent the start and the end of a cost range. These two lines are closely intersecting with the classifier curve. Since both ISO performance lines are intersecting close to each other with the ROC curve, then we can say the classifier has low sensitivity to cost ratio changes (but we are unable to tell by how much).

In a similar way, conceptual Figure 4 is showing a classifier that has high-cost sensitivity to cost ratio changes. The two ISO performance lines (the solid and dashed lines) represent the start and the end of a cost range. These two lines are widely intersecting with the classifier curve. Since both ISO performance lines are intersecting on far parts with the ROC curve, then we can say this classifier has high sensitivity to cost changes (but we are unable to tell by how much).

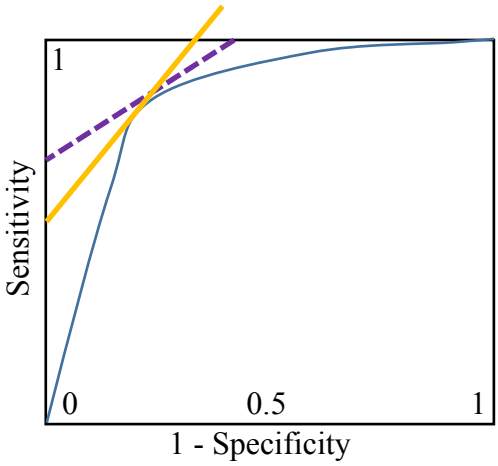

Figure 3. This figure shows the low-cost sensitivity analysis in the ROC graph using two ISO performance lines which represent the start and the end of the cost ratio range. [4]

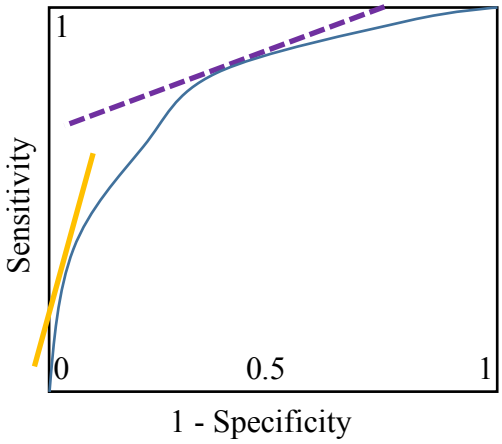

Figure 4. This figure shows the high-cost sensitivity analysis in the ROC graph using two ISO performance lines which represent the start and the end of the cost ratio range. [4]

"ROC curves do not visually depict the quantitative performance analysis of a classifier or the difference in performance between two classifiers." [5] Besides the assumption of equal costs in the ROC, there are many other shortcomings of quantitative measures, such as [5]:

- What is the total misclassification cost from a specific classifier when given each class misclassification error costs.
- By how much total misclassification cost, classifier 1 contrasts classifier 2.
- Under what misclassification cost ratios classifier 1 outmatches classifier 2.

Hence, the ROC does not give much of a specific guidance to data scientists to choose one classifier over another when their ROC curves overlap. Unless one classifier prevails all the

other models throughout all classification threshold cutoffs. “Only when one classifier clearly dominates another over the entire performance space can it be declared better.” [4]

2.3 The Area Under the Curve (AUC)

The AUC is used in conjunction with many charts such as the ROC to evaluate classifiers’ performances. The easy use of the AUC (it is a single number) from the user’s perspective made it widely used to compare curves. The AUC of the ROC is used in many disciplines such as medicine, radiology, psychology, credit scoring, and bioinformatics, and others. [9] However, it is not an accurate measurement especially when two classifiers cross each other because one of them may be best only under specific conditions. “Only in the case that one classifier dominates another will the AUC be universally valid in a comparison of classifiers.” [3] In other words, it is possible for a classifier who has larger AUC to perform worse in a specific region of the ROC space than other classifiers who have smaller AUC. [6] Conceptual Figure 5 shows how the AUC can be deceiving when two ROC curves cross each other. Clearly, we can see that the AUC of classifier B is larger than the AUC of classifier A. Therefore, classifier B should be considered as with superior performance. In spite of that, we can see that classifier A performs better when 1-Specificity is approximately smaller than 0.2.

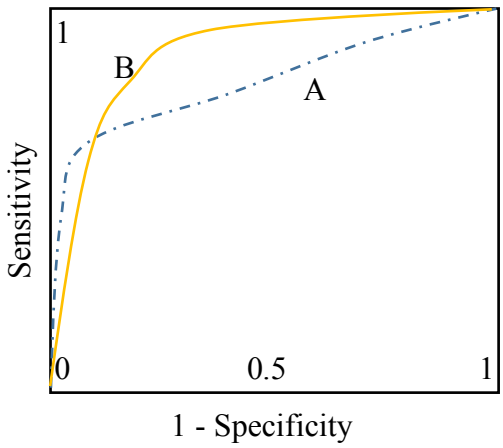

Figure 5. This is the ROC graph. Classifier A performance is represented by the dashed curve, and classifier B performance is represented by the solid curve.

Furthermore, the AUC considers the performance of regions that might be rarely used. [8] The AUC measure the performance of a classifier over all classification thresholds. [11] While in practice, we either use one classification threshold or a specific range of classification thresholds. Because of all these shortcomings, we will not rely on the AUC to compare classifiers' cost curves in our proposed cost chart.

2.4 Instances' Misclassification Costs

Since in real-world the scenario of equal error costs is very rare, the need to embrace the cost of misclassification as a criterion to compare classifiers' performances is significant. [3] As a result, scientists realized the importance of incorporating the cost of misclassified instances, and they came up with many charts that reflect misclassification cost variations. However, some methods are difficult to read, understand, and some carry less useful information to the user.

"In real-world environments, it usually is difficult to specify target operating conditions precisely, for example, target misclassification costs. This uncertainty makes building robust classification systems problematic. "[4] Nevertheless, not knowing the exact cost is not a dead-end for more realistic evaluation methods of classification modules, and an alternative strategy can be used. The cost ratio range of interest is always known. [3] Data scientists can estimate the cost of errors by the help of their clients and show all the possible variations of cost ratios on a cost chart. Even when we estimate the cost ratio of errors regarding a problem in hand, we cannot assume that it is neither specific nor static. [4] Therefore, this cost chart will facilitate the task of selecting the classifier that performs better in the area of interest and under specified conditions. Furthermore, not all wrong-actions costs appear in a monetary form. The ramifications can also be in the form of time wasted, life quality, diagnosis of illness severity, or other forms. [7]

3. FORMAL PROBLEM DESCRIPTION

As we mentioned in the introduction chapter, data mining techniques are being used in many disciplines, where data scientists work with professionals to find solutions for their problems. However, data scientists may face the difficulty of having various perspectives acquired from different professionals, which may result in contrasting cost estimates for the same problem. For example, in the medical field, data scientists do not consider the cost of misclassified instances in their classifiers evaluation. The reason behind that is due to having various error cost estimations for one problem [1]. Therefore, we decided in this thesis to show the importance and necessity of considering a range of costs for misclassified instances to evaluate a classifier's performance.

3.1 Problem Definition

In this thesis, we are only considering binary classification problems with two classes usually labeled as 1 and 2. Where 1 represents the negative class and 2 represents the positive class.

As the wise man says, "Define the problem correctly, and enjoy the solution process." We have to ask ourselves a cornerstone question, what is our goal from the classification process? We believe there are two answers to this question as follows:

- Accuracy-wise

Is to correctly classify the highest number of instances, and the classifier with the highest accuracy will be considered as superior. Here, we are not considering the severity of the consequences of misclassified instances as a performance measure. Thus, we may reach a high classification accuracy but we may misclassify important instances.

- Cost-wise

Is to minimize the total misclassification cost of errors, and the classifier with the smallest total cost will be considered as superior. However, this approach does not necessarily imply

reaching the highest correctly classified instances. In other words, we may commit more errors, but errors that are cheap which may make the total misclassification cost the lowest.

Intuitively, the best classifier is the one that correctly classifies more instances than its peers. This is a very abstract definition for the classification task because each classifier can be optimal only when certain conditions are met. A classifier may not be optimal across all misclassification costs and data distribution changes. Therefore, regardless of the approach we will choose to evaluate classifiers whether it is accuracy or cost-wise (cost-sensitive classification), making classification errors is almost unavoidable. Hence, we believe selecting the cost-wise approach as a criterion to compare classifiers' performances is a better choice. This way will help data scientists to pick the classifier with the minimum total misclassification cost and as result will minimize the cost of the decision-making process.

3.2 The Confusion Matrix

In two-class problems, the confusion matrix considers four scenarios as follows (see Table 2):

- If the instance is actually negative and was predicted as negative, then it is denoted as true negative.
- If the instance is actually positive and was predicted as positive, then it is denoted as true positive.
- If the instance is actually negative and was predicted as positive, then it is denoted as false positive.
- If the instance is actually positive and was predicted as negative, then it is denoted as false negative.

Table 2. The confusion matrix. A false positive represents a false alarm, and a false negative represents a miss.

	Predicted as 1 Class	Predicted as 2 Class	
Actual Class 1	True Negative (TN)	False Positive (FP)	= Total Actual Negative
Actual Class 2	False Negative (FN)	True Positive (TP)	= Total Actual Positive
	= Total Predicted Negative	= Total Predicted Positive	

3.3 The Cost Matrix

In two-class problems, the associated cost matrix considers four scenarios as follows:

- If the instance actually belongs to class 1 and was predicted as 1, then its cost is zero.
- If the instance actually belongs to class 2 and was predicted as 2, then its cost is zero.
- If the instance actually belongs to class 1 and was predicted as 2, then its cost is FPcost.
- If the instance actually belongs to class 2 and was predicted as 2, then its cost is FNcost.

Table 3. This is the cost matrix. There is no cost for correctly classified instances (TP and TN).

	Predicted as 1 Class	Predicted as 2 Class
Actual Class 1	0	False Positive cost (FPcost)
Actual Class 2	False Negative cost (FNcost)	0

3.4 Additional Formulas

All the following formulas can be found in [11]. They all can be derived from the confusion matrix.

$$Accuracy = \frac{TP + TN}{FP + FN + TP + FP} = Hit\ Rate$$

$$Error\ rate = 1\text{-}Hit\ Rate$$

$$True\ Positive\ Rate = \frac{Correctly\ Predicted\ Positive\ Instances\ (TP)}{Total\ Actual\ Positive\ Instances(= TP + FN)} = Sensitivity$$

$$True\ Negative\ Rate = \frac{Correctly\ Predicted\ Negative\ Instances\ (TN)}{Total\ Actual\ Negative\ Instances\ (= TN + FP)} = Specificity$$

$$False\ Positive\ Rate = \frac{FP}{FP + TN} = (1\text{-}Specificity)$$

$$False\ Negative\ Rate = \frac{FN}{FN + TP}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = True\ Positive\ rate = Sensitivity$$

4. PROPOSED SOLUTION APPROACH

In the literature review chapter, we pointed out the shortcomings of using the accuracy and the ROC graph to compare the performances of classifiers. In this chapter, we are going to use the cost-sensitive classification approach to compare the performances of classifiers. Basically, the lower the minimum total misclassification cost of a classifier is, the better its performance compared to other classifiers is assumed to be. Since we are not sure about the exact cost of each misclassified instance, we are going to test a range of costs for each class.

In order to find the minimum total misclassification cost for a specific cost ratio, we need to test all the classification thresholds in the range of [0, 1]. This means we need to classify the data with different classification thresholds. The FP and FN rates are generated for each classification threshold after performing the classification process. We are calculating the total misclassification cost for the given cost ratio with each classification threshold. This is done through summing the multiplication of the FP cost by the FP rate with the multiplication of the FN cost by the FN rate. Then, we can select the classification threshold that leads to the smallest total misclassification cost. This threshold will be considered as the optimal classification threshold at the given cost ratio. Thus, we say that the optimal performance of a classifier for the given cost ratio is achieved at the optimal classification threshold.

The final result of the minimized total misclassification cost at each cost ratio will be shown on a cost chart. This cost chart is called the Minimum Total Misclassification Cost for a Single Cost Ratio (MTMCS). Also, we are showing the optimal classification thresholds that led to the minimum total misclassification cost at each cost ratio on another chart. This chart is called the Optimal Classification Thresholds.

Next, we will describe in detail how we calculate the total misclassification cost for each classification threshold. After that, we will explain how we are carrying out the search for the optimal classification threshold which led to the minimum total misclassification cost for a

given cost ratio. Later on, we are going to propose a new measure to evaluate a classifier's performances when cost curves overlap. This new measure uses the minimum total misclassification cost for a range of cost ratios and the cost sensitivity.

4.1 Fundamental Notation

The problem we address in this thesis is how to compare different classifiers when we have different unit of misclassification costs for FP and FN errors.

- Let FPcost denote the unit of misclassification cost of false positive errors when an FP case occurs. Similarly, let FNcost denote the unit of misclassification cost of false negative errors when an FN case occurs. Since professionals usually provide their FPcost and FNcost estimations in the form of ratios [3], we are denoting the cost ratio as r and as follows:

$$r = \frac{FPcost}{FNcost}$$

where FPcost > 0 and FNcost > 0.

- Let FNRate denote the rate of FN (false negative) errors when they occur. Similarly, let FPRate denote the rate of FP (false positive) errors when they occur. Hence, the corresponding total misclassification cost denoted as (TMC) for a specific cost ratio r (FPcost / FNcost) can be determined as follows:

$$TMC\ (r) = (FNRate * FNcost) + (FPRate * FPcost)$$

where FPcost > 0 and FNcost > 0.

We need to apply this formula with all the classification thresholds (by using a user-defined step size within the range [0, 1]) to find the minimum TMC at r. The minimum TMC for a single cost ratio is denoted as MTMCS.

- Suppose that we are interested in the performance of a classifier when the cost ratio r varies in the range $[r_1, r_2]$ with a user-defined step size. As a result, the minimum total misclassification cost of the area covered by the cost ratio range of interest will be the

sum of the MTMCS values for each cost ratio within the specified range. Let MTMCR denote the minimum total misclassification cost of the area covered by the cost ratio range $[r_1, r_2]$ as follows:

$$MTMCR\ ([r_1, r_2]) = \sum_{r=r1}^{r2} MTMCS\ (r)$$

In order to cast away any confusion, we have three types of total misclassification costs:

- TMC (r), which represents the total misclassification cost of cost ratio r at any classification threshold.
- MTMCS (r), which represents the minimum TMC of cost ratio r at the optimal classification threshold.
- MTMCR ($[r_1, r_2]$), which represents the sum of the MTMCS values of a range of cost ratios $[r_1, r_2]$.

4.2 The Minimum Total Misclassification Cost for a Single Cost Ratio (MTMCS)

In the MTMCS chart, a data scientist can compare cost-wise the classifiers' performances using the MTMCS values on the Y-axis. The Y-axis represents the percentage of misclassification cost on a scale of [0 to 100]. We only show the MTMCS values at each cost ratio r. The X-axis represents the cost ratio variations in the range $[r_1, r_2]$. Moreover, one of the main reasons to use a range of cost ratios is because even if we know the precise cost (which is a rare scenario), this cost may change in time and we want to see what will happen if this scenario occurs. Thus, we need to test various cost ratios because:

- It is rare to know the exact cost ratio. Usually, we are not certain about the exact cost of misclassification.
- In time, the cost ratio of misclassification may change.

Furthermore, another use of this chart is that it enables us to quantify how much each classifier is sensitive to cost changes. This is done by subtracting the highest MTMCS from

the lowest MTMCS on a cost curve. This process requires a normalized cost ratio axis in order to make a comparison between points on the same axis possible.

4.2.1 The Cost Ratio (FPcost / FNcost) Normalization

The MTMCS chart has two axes, where the X-axis uses normalized cost ratios, and the Y-axis shows the percentage of the MTMCS values at each cost ratio r.

Regarding the X-axis, we used normalized cost ratios, which means that the costs FNcost and FPcost add up to 1. We must normalize r because using the actual (not normalized) ratios can be problematic in practice. The cost ratio range of interest elicited from domain experts can vary from one expert to another. [3] The following are examples of different cost ratio ranges that can be provided from domain experts:

- One expert may say that the cost ratio range of the misclassified instances of the positive class is between 1 and 10 times more expensive than the misclassified instances from the negative class. Thus, the cost ratio range of interest is between [1, 0.1].
- Other domain experts may say that the cost ratio range of misclassification is simply between [0, 1]. In this example the cost ratio range starts from 0, while in the previous example the cost ratio will always be bigger than 0.
- Others may say that one class is more expensive than the other, therefore, the cost ratio range of interest would be between [1, ∞] for the expensive class. It is impractical to have an axis on a chart that goes until ∞.

Furthermore, normalizing the cost ratio of misclassification will not affect the result. Thus, the TMC function will be after normalization as follows:

$$TMC\ (r) = (FNRate * FNcost) + (FPRate * FPcost)$$

where FPcost > 0, FNcost > 0 and FPcost + FNcost =1.

Regarding the Y-axis, we used the FP and FN rates in the TMC formula instead of the falsely classified instances numbers (positive and negative). The reason behind that is to make

the cost of misclassification varies only between [0, 100] for all dataset sizes. The Y-axis in Figure 6 shows the MTMCS values using the falsely classified instances numbers. Whilst the Y-axis in Figure 7 shows the percentage of MTMCS values using the falsely classified instances rates.

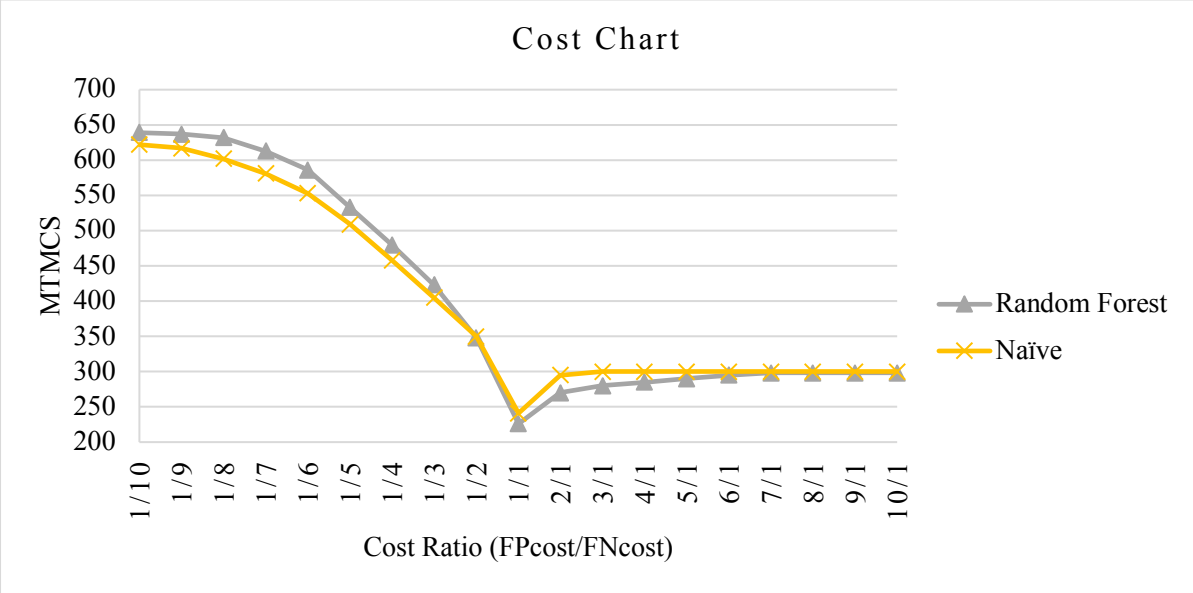

Figure 6. In this cost chart, the Y-axis represents the MTMCS values by using the falsely classified instances numbers. The X-axis represents the cost ratios [r_1, r_2] and the costs are not normalized. The curves represent the optimal performances of two classifiers.

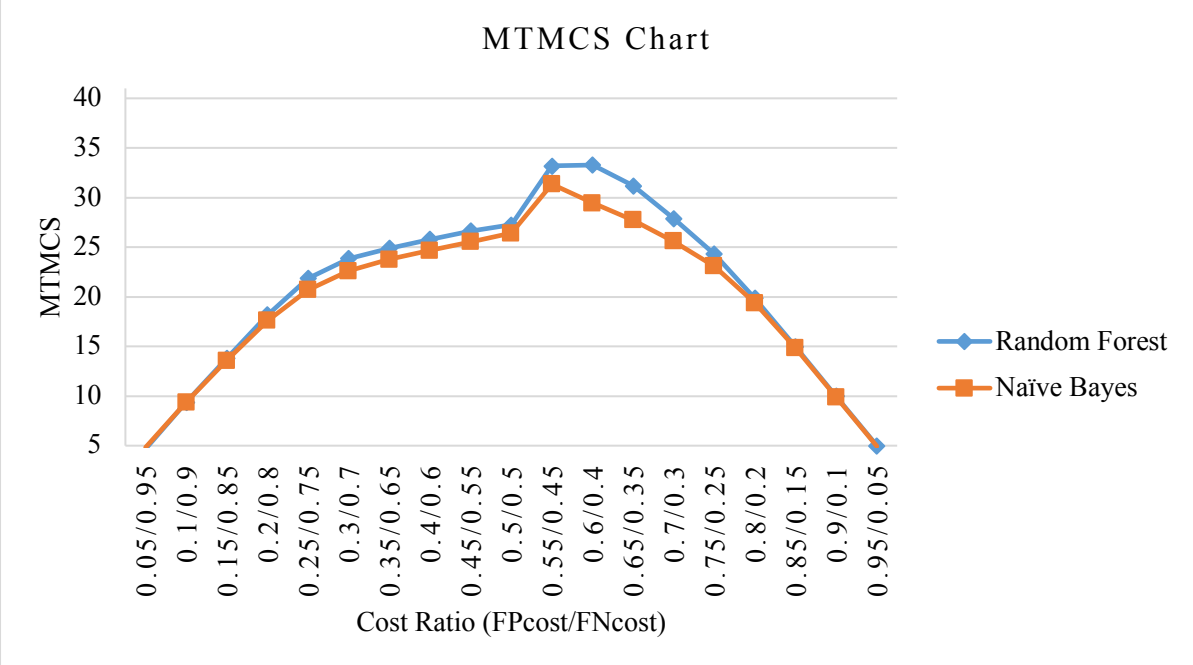

Figure 7. This is the MTMCS chart. The Y-axis represents the percentage of the MTMCS values by using the falsely classified instances rates. The X-axis represents the cost ratios [r_1, r_2] and the costs are normalized. The curves represent the optimal performances of two classifiers.

4.2.2 The Problem of Comparing Overlapping Cost Curves and Cost Sensitivity Analysis

When a cost curve dominates others across all cost ratios, then it is assumed to be of superior performance. However, this case is rare, and curves usually overlap. A common method used to measure the performance of classifiers is the Area Under the Curve (AUC). For instance, the AUC is used in conjunction with the ROC curves; the bigger the AUC for any ROC curve is, the better its performance is. Having said that, the AUC has a shortcoming because it considers the whole area covered by the curve, while a classifier will perform best only under specific conditions and not under all. Therefore, instead of using the AUC to measure the area under the cost curves in the MTMCS chart, we will use the MTMCS values themselves to solve the problem of comparing overlapping cost curves.

4.2.2.1 The Minimum Total Misclassification Cost of a Range of Cost Ratios (MTMCR)

We propose that when cost curves overlap within the range of interest of cost ratios [r_1, r_2], we simply have to derive the MTMCR value for that cost range and for each classifier. The MTMCR value is derived by adding up the MTMCS values associated with each cost ratio within the range [r_1, r_2]. The MTMCR value represents the minimum total misclassification cost of the covered area by the cost ratio range [r_1, r_2]. Afterwards, we compare the results of the MTMCR values for each classifier to select the classifier with the smallest MTMCR value.

The idea of the MTMCR is an approximation of the AUC. However, the MTMCR is the summation of discretized values while the AUC is the integration of the area covered by a curve. In addition, the MTMCR can be calculated for the area of interest, while the AUC represents the whole area covered by the entire curve.

4.2.2.2 Cost Sensitivity Analysis

The solution we proposed that relies only on the MTMCR values to tackle the problem of comparing overlapped cost curves has a weakness. This weakness stems from the *assumption* that all overlapping classifiers' cost curves have equal cost sensitivity to cost ratio

changes. Therefore, it lacks another important factor which is, the classifiers' cost sensitivity to cost ratio changes.

First, the term sensitivity refers to how much a classifier's performance is responsive to cost ratio changes. We can discover whether a classifier is cost sensitive or not by changing the cost ratio (FPcost and FNcost) applied to data; if the MTMCS value changes significantly with different cost ratios, then the classifier is considered sensitive to cost ratio changes; otherwise, it is not sensitive. The MTMCS chart can help us to easily know by how much each classifier is sensitive to cost ratio changes. The following formula shows how cost sensitivity is calculated:

$$\textit{Cost Sensitivity} = \textit{Highest MTMCS} - \textit{Lowest MTMCS}$$

where both MTMCS values are within the cost ratio range of interest $[r_1, r_2]$.

Here, we must ask ourselves is sensitivity useful to help evaluate classifiers' performances? The answer is yes; cost sensitivity is a significant factor to measure the performance of a classifier and to select the best one for the problem at hand. Let us consider conceptual Figure 8 to understand why we need to consider the cost sensitivity to evaluate classifiers' performances. In Figure 8, the MTMCS chart shows the performance of classifiers 1 and 2 over three cost ratios A, B, and C. Obviously, classifier 1 has a higher cost sensitivity to cost ratio changes compared to classifier 2. In order to derive the MTMCR of the range of cost ratios of interest, we need to sum up the MTMCS values of the cost ratios A, B, and C. The MTMCR value of classifier 1 is calculated as follows:

$$MTMCR_1 = MTMCS_{1,A} + MTMCS_{1,B} + MTMCS_{1,C}$$

$$= 13 + 26 + 40$$

$$= 79$$

The MTMCR value of classifier 2 is calculated as follows:

$$MTMCR_2 = MTMCS_{2,A} + MTMCS_{2,B} + MTMCS_{2,C}$$

= 20 + 28 + 32

= 80

Thus, we can say that classifier 1 has the smallest MTMCR value compared to classifier 2. Therefore, it should be selected as the best classifier. The problem here is that what if most future errors are going to have a cost ratio that correspond to point C where classifier 1 performs worst, and rarely correspond to cost ratios A and B. Then the MTMCR value of classifier 1 is basically:

$$MTMCR_1 = MTMCS_{1,C}$$

= 40

And the MTMCR value of classifier 2 is basically:

$$MTMCR_2 = MTMCS_{2,C}$$

= 32

So, from Figure 8 we can say that classifier 2 has the smallest MTMCR value compared to classifier 1.

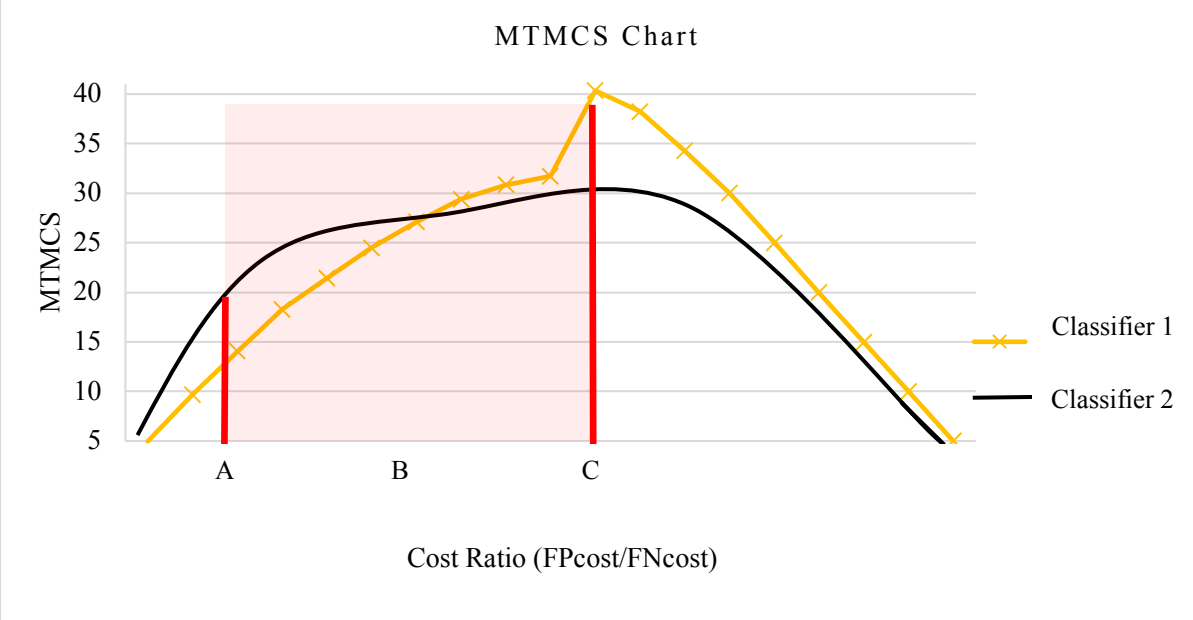

Figure 8. This is a conceptual figure to show the MTMCS values of three cost ratios: A, B, and C. The range of cost ratios we are interested in is between the two vertical lines.

Hence, a classifier with the smallest MTMCR value compared to other classifiers may not be the best classifier after all if it has a high sensitivity to cost ratio changes! Therefore, the lower the cost sensitivity the lower the chance of having a higher MTMCR value than expected during training and testing phases. Nonetheless, not every classifier with low-cost sensitivity is with better performance (lowest MTMCR value).

4.2.2.3 The Cost and Sensitivity Tradeoff (CST) Measure

Simply put, classifiers that have high-cost sensitivity pose a higher chance to have a higher cost compared to other classifiers than the expected MTMCR value. Thus, the MTMCR value can be deceiving if the classifier is sensitive to cost ratio changes. Therefore, we need to consider this scenario of having sensitive classifiers through incorporating cost sensitivity to evaluate classifiers' performances. As a result, we propose the following heuristic formula by incorporating both, the MTMCR value and the cost sensitivity to measure the performance of classifiers.

$$CST = MTMCR + (MTMCR * \frac{Cost\ Sensitivity}{100})$$

where 100 is the maximum possible value of the MTMCS.

After simplifying the equation, it is as follows:

$$CST = MTMCR\ (1 + \frac{Cost\ Sensitivity}{100})$$

The tradeoff between the MTMCR and cost sensitivity values is achieved through the CST measure. Since the CST measure relies on the misclassification cost and the cost sensitivity, then of course, the lower it is the better the performance of a classifier is. A classifier with the lowest CST value is considered to be the best choice. Thus, if we use the CST measure with the example mentioned in the previous sub-section related to Figure 8, the solution will be as follows:

$CST_1 = 79 * (1 + 0.27)$

$= 100.33$

$CST_2 = 80 * (1 + 0.12)$

$= 89.6$

Hence, based on the CST measure results, we recommend selecting classifier 2 as the optimal classifier for this problem.

4.3 Finding the Optimal Classification Threshold

Each classifier used in this thesis can produce the probability of class membership of each instance in a dataset. Intuitively, the higher the membership probability of an instance is, the bigger the chance it belongs to the given class. Based on this posterior probability, we are going to search through the classification threshold range of [0, 1] to select the one threshold with the smallest TMC value. This optimal threshold is where the classifier performs the best. We repeat this process to find the smallest TMC values at each cost ratio.

4.3.1 The Thorough Search to Find the Optimal Classification Threshold

With each classification threshold we use from the range [0, 1], some of the instances of each class may be misclassified. By changing the classification threshold at cost ratio r, the misclassified instances may differ, which as a result may lead to a different TMC value. The threshold that leads to the smallest TMC value at cost ratio r will be considered as an optimal classification threshold. Thus, the classifier with the overall cheapest total misclassification cost will be considered as the superior classifier.

The search for the classification threshold with the smallest TMC value is done by changing the threshold with a user-defined step size within the default range of [0, 1]. Thus, finding the smallest TMC value for a specific cost ratio r requires two phases, as follows:

- Phase 1: generating an array with the TMC values for each and all classification thresholds. The time complexity to generate this array is calculated as follows:

$$O_2 = (m * n)$$

where *n* is the number of instances, and *m* is the number of the classification thresholds.

- Phase 2: finding the smallest TMC value from the generated array in phase 1. The search time complexity to find the smallest TMC is calculated as follows:

$$O_1 = m$$

where *m* is the number of the classification thresholds.

Hence, the total time required to find the smallest TMC value at cost ratio r is calculated as follows:

$$O = O_1 + O_2$$

For example, let us consider a dataset of 10,000 instances to find the smallest TMC value at cost ratio r. We need to classify this dataset with different classification thresholds from the range [0, 1]. If the predefined classification threshold step size is 0.0001, then, the total time required to find the smallest TMC value at cost ratio r is calculated as follows:

$$m = \frac{1}{0.0001}$$

= 10,000 is the number of classification thresholds.

$O_2 = (m * n)$

= 10,000 * 10,000

= 100,000,000 is the running time required to generate the TMC values for all the classification thresholds.

$O_1 = m$

= 10,000 is the running time required to find the smallest TMC value.

Hence, the total time required to find the classification threshold with the smallest TMC at cost ratio r is:

= 100,000,000 + 10,000

= 100,010,000

However, if we use a bigger classification thresholding step size such as 0.01 instead of 0.0001, then the running time of thresholding through the entire threshold range [0, 1] will be as follows:

$$m = \frac{1}{0.01}$$

= 100 is the number of classification thresholds.

$O_2 = (m * n)$

= 100 * 10,000

= 1,000,000 is the running time required to generate the TMC values for all the classification thresholds.

$O_1 = m$

= 100 is the running time required to find the smallest TMC value.

Hence, the total time required to find the classification threshold with the smallest TMC value at cost ratio r is:

= 1,000,000 + 100

= 1,000,100

We can see that the difference in the running time between the two scenarios is computationally significant. Of course, the smaller the classification threshold step size is the less the total misclassification cost might be. This is because we are searching with a higher accuracy to find a better approximation to a global optimum threshold. On the other hand, that will increase the computational time significantly.

4.3.2 The Optimized Search to Find the Optimal Classification Threshold

We saw in the previous sub-section that we have two choices to find the optimal classification threshold. The first choice is that we use a small classification threshold step size to find a good approximation to a global optimum threshold but at the expense of the computational time. The second choice is that we use a larger classification threshold step size

that requires significantly less computational time but at the expense of the resulting quality. There is a solution to this dilemma through optimizing the search for the optimal classification threshold. To optimize the search for the best threshold with the minimum TMC at a specific cost ratio r, we can use the notion that states the following: When FNcost is more expensive than FPcost, then the optimal classification threshold cutoff will be closer to 1, and vice versa.

Since the FN errors are more expensive than the FP errors, we are going to classify the instances as positive instances even when their positive class membership probability is low, and vice versa. The following formula and rules [3] are used to decrease the length of the search space for the optimal classification threshold at cost ratio r:

$$Initial\ Threshold\ Cutoff = \frac{FNcost}{FNcost + FPcost}$$

- When FNcost is equal or bigger than 0.5, then the optimal classification threshold cutoff is between the initial threshold cutoff and 1.
- When FPcost is bigger than 0.5, then the optimal classification threshold cutoff is between the initial threshold cutoff and 0.

The above rules will shrink the search space length and as a result, will reduce the search time for the optimal classification threshold significantly. Moreover, the bigger the difference between the FPcost and the FNcost is, the smaller the area we should search to find the optimal classification threshold cutoff.

In our experiments (explained in chapter 5), we used the optimized thresholding method instead of the thorough search to find the optimal classification threshold at each cost ratio r. This is because we want to reduce the processing time since we used a small thresholding step size (0.00001) to find the smallest TMC at each cost ratio.

4.4 The Optimal Classification Thresholds Chart

In this chart, a data scientist would be able to determine cost-wise the best classification threshold operating range for the cost ratio range of interest by using the Y-axis. The Y-axis represents the classification thresholds on a range of [0, 1]. We only show the optimal classification thresholds that lead to the MTMCS values at each cost ratio. Furthermore, the X-axis represents the normalized cost ratios, which is helpful to make a connection with the MTMCS chart. For example, let us say that we are interested in the optimal classification threshold operating range of the Random Forest classifier between the cost ratio range [0.3/0.7, 0.45/0.55] in Figure 9. We can easily say that the optimal classification threshold operating range is between [0.72, 0.75].

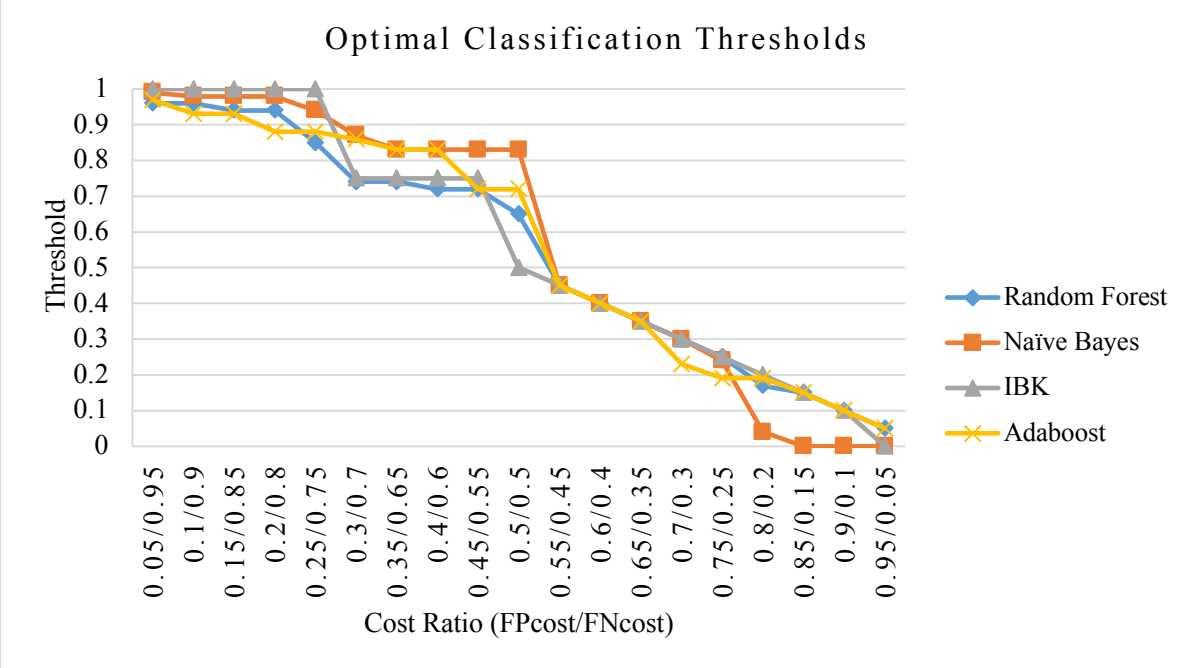

Figure 9. This chart shows the optimal classification thresholds at each cost ratio and for each classifier. The dataset used is German Credit from University of California, Irvine, (UCI) repository.

4.5 The Cost-Sensitive Classification Algorithm

We incorporated the optimized search for the optimal classification thresholds in the cost-sensitive classification algorithm as follows:

Input:

- The posterior probabilities for all the instances
- Cost ratio range [r_1, r_2] /* $r = \frac{FPcost}{FNcost}$ */

Output: List of MTMCS values of each cost ratio r to form a cost curve in the MTMCS chart.

- Calculate the initial classification threshold cutoff

$$Initial\ Threshold\ Cutoff = \frac{FNcost}{FNcost + FPcost}$$

- Classify instances:

If FNcost ≥ FPcost, then

Classification Rule 1: If Probability of (class 1 | given data) ≥ threshold cutoff

Then the instance is classified as class 1

Classification Rule 2: Otherwise the instance is classified as class 2

Else,

Classification Rule 3: If Probability of (class 1 | given data) ≤ threshold cutoff

Then the instance is classified as class 2

Classification Rule 4: Otherwise the instance is classified as class 1

- Generate a confusion matrix for the tested classification threshold
- Compute the TMC value for the tested classification threshold:

$$TMC\ (r) = (FNRate * FNcost) + (FPRate * FPcost)$$

Add point to list L1 (threshold, TMC)

- Using the optimized search for the optimal classification threshold at cost ratio r.

If FNcost ≥ 0.5, then

Increment the initial threshold by the user-defined step size and classify again.

Repeat until classification threshold = 1

Else,

Decrement the initial threshold by the user-defined step size and classify again

Repeat until classification threshold = 0

- If all the classification thresholds are tested, then

Find the minimum TMC value for the selected cost ratio r from L1

Add point to list L2 (cost ratio r, minimum TMC, optimal threshold)

- Select another cost ratio and repeat all the steps to create a cost curve

4.5.1 Applying the Cost-Sensitive Classification Algorithm

According to the cost-sensitive classification algorithm, we are facing two scenarios as follows:

4.5.1.1 Scenario 1: FNcost ≥ FPcost

On the left side of the MTMCS chart, we are showing the scenario where the cost of misclassification of FN errors is more expensive than for FP errors. Suppose we have a problem of 14 instances with the misclassification cost ratio of 1 to 4, where 1 and 4 are the costs of FP and FN errors respectively. The solution would be as follows:

- The first step, we need to normalize the actual cost ratio.

$$Normalized\ FPcost = Actual\ FPcost * \frac{1}{(Actual\ FPcost + Actual\ FNcost)}$$

$$= 1 * \frac{1}{1+4}$$

$$= 0.2$$

$$Normalized\ FNcost = Actual\ FNcost * \frac{1}{(Actual\ FPcost + Actual\ FNcost)}$$

$$= 4 * \frac{1}{1+4}$$

$$= 0.8$$

- The second step, we calculate the initial classification threshold based on the normalized cost ratio (we do not start from 0):

$$Initial\ Threshold\ cutoff = \frac{FNcost}{FNcost + FPcost}$$

$$= \frac{0.8}{0.8 + 0.2}$$

$$= 0.8$$

- The third step is making a decision about the classification assignment sign (smaller than or bigger than)

Since the FNcost is bigger than FPcost, then anything bigger than or equal to the initial classification threshold cutoff will be assigned to the negative class. Therefore, the rule of classification is:

If the posterior probability ≥ threshold, then assign the instance to the negative class.

Otherwise, assign it to the positive class.

- The fourth step is classifying the data.

Table 4 shows the posterior probabilities in the left column of 14 instances, and the assigned classes after applying the cost-sensitive classification algorithm are on the right column.

Table 4. The posterior probabilities and the assigned classes when FNcost ≥ FPcost.

The Posterior Probability for Each Instance to be a Negative Instance	The Assigned Class After Classification
0.01	Positive
0.05	Positive
0.15	Positive
0.18	Positive
0.20	Positive
0.25	Positive
0.35	Positive
0.65	Positive
0.75	Positive
0.80	Negative
0.85	Negative
0.90	Negative
0.95	Negative
0.99	Negative

- The fifth step is searching for the optimal classification threshold that will lead to the smallest TMC value at a specific cost ratio.

We already started with an initial threshold cutoff, but we need to test more classification thresholds to find the best approximation to a global optimal one. When FNcost is more expensive than FPcost, we use this formula:

$$\textit{Threshold cutoff for the negative class is} \geq \frac{FNcost}{FNcost + FPcost}$$

This formula is assuming that the optimal classification threshold is 0.8 or bigger. Thus, the optimal threshold lies between [0.8 and 1]. Therefore, we start increasing the initial threshold 0.8 using our predefined step size. Each time we increase the classification threshold, we classify the data and derive the TMC value for that threshold. We repeat this process until we reach the threshold cutoff 1.

- The sixth step is to select one approximation to a global optimal classification threshold with the smallest TMC value.

Of course, with each classification threshold cutoff between [0.8, 1] we are calculating the TMC values and storing them. Then, it is time to select the optimal classification threshold with the smallest TMC value and relate it to the specified cost ratio.

- The seventh step is to go for the next cost ratio and repeat the above steps for every cost ratio we are interested in.

For example, we used (FNcost = 0.8 and FPcost= 0.2), and so, the next cost ratio can be (FNcost = 0.9 and FPcost= 0.1).

4.5.1.2 Scenario 2: FPcost > FNcost

On the right side of the MTMCS chart, we are showing the second scenario where the cost of misclassification of FP errors is more expensive than for FN errors. We are using the same problem from scenario 1 but with the following misclassification cost ratio: 4 to 1, where 4 and 1 are the costs of FP and FN errors respectively. The solution would be as follows:

- The first step, we need to normalize the actual cost ratio.

$$Normalized\ FPcost = Actual\ FPcost * \frac{1}{(Actual\ FPcost + Actual\ FNcost)}$$

$$= 4 * \frac{1}{4 + 1}$$

$$= 0.8$$

$$Normalized\ FNcost = Actual\ FNcost * \frac{1}{(Actual\ FPcost + Actual\ FNcost)}$$

$$= 1 * \frac{1}{4 + 1}$$

$$= 0.2$$

- The second step, we calculate the initial classification threshold based on the normalized cost ratio (we do not start from 0):

$$Initial\ Threshold\ cutoff = \frac{FNcost}{FNcost + FPcost}$$

$$= \frac{0.2}{0.2 + 0.8}$$

$$= 0.2$$

- The third step is making a decision about the classification assignment sign (smaller than or bigger than).

Since the FNcost is smaller than FPcost, then anything smaller than or equal to the initial classification threshold cutoff will be assigned to the positive class. Therefore, the rule of classification is:

If the posterior probability ≤ threshold, then assign the instance to the positive class.

Otherwise, assign it to the negative class.

- The fourth step is classifying the data.

Table 5 shows the posterior probabilities in the left column of 14 instances, and the assigned classes after applying the cost-sensitive classification algorithm are on the right column.

- The fifth step is searching for the optimal classification threshold that will lead to the smallest TMC value at a specific cost ratio.

We already started with an initial threshold cutoff to classify the data, but we need to test more classification thresholds to find the best approximation to a global optimal one. When FPcost is more expensive than FNcost, we use this formula:

$$Threshold\ cutoff\ for\ the\ positive\ class\ is \leq \frac{FNcost}{FNcost + FPcost}$$

This formula is assuming that the optimal classification threshold is 0.2 or smaller. Thus, the optimal classification threshold lies between [0.2, 0]. Therefore, we start decreasing the initial threshold 0.2 using our predefined step size. Each time we decrease the classification threshold, we classify the data and derive the TMC value for that threshold. We repeat this process until we reach the threshold cutoff 0.

Table 5. The posterior probabilities and the assigned classes when FPcost > FNcost.

The Posterior Probability for Each Instance to be a Negative Instance	The Assigned Class After Classification
0.01	Positive
0.05	Positive
0.15	Positive
0.18	Positive
0.20	Positive
0.25	Negative
0.35	Negative
0.65	Negative
0.75	Negative
0.80	Negative
0.85	Negative
0.90	Negative
0.95	Negative
0.99	Negative

- The sixth step is to select one approximation to a global optimal classification threshold with the smallest TMC value.

Of course, with each classification threshold cutoff between [0.2, 0] we are calculating the TMC values and storing them. Then, it is time to select the optimal classification threshold with the smallest TMC value and relate it to the specified cost ratio.

- The seventh step is to go for the next cost ratio and repeat the above steps for every cost ratio we are interested in.

For example, we used (FNcost = 0.2 and FPcost= 0.8), and so, the next cost ratio can be (FNcost = 0.1 and FPcost= 0.9).

5. RESULTS AND DISCUSSION

In this chapter, we are showing four classifiers' performances over four datasets. We compared their performances using the ROC graph and the MTMCS chart. We used Weka 3-9-0 to produce the ROC graphs and to generate the AUC values for all the ROC curves. However, since Weka produces poor ROC resolution graphs, we used Excel to show its results. Furthermore, the classifiers we used in our experiments are from Weka's open source code, and are as follows:

- Random forest
- Naïve Bayes
- Adaboost
- IBK (K Nearest Neighbor)

We used the cross-validation method with 10 folds to train all the classifiers and produce the posterior probabilities. This method helps to overcome the problem of overfitting. In addition, all the datasets used in our experiments are from the University of California, Irvine, (UCI) repository. We selected datasets with various data distributions. The datasets distribution types are as follows:

- Balanced distribution datasets where usually one of the two classes hold relatively higher cost than the other class.
- Unbalanced distribution datasets where usually the minority class holds extremely higher cost than the majority class.

5.1 Robust vs Optimal Classifier

"Stating that a classifier is robust is stronger than stating that it is optimal for a specific set of conditions. A robust classifier is optimal under all possible conditions." [4]. When data scientists are trying to select one classifier in the training and testing phases using the MTMCS chart, they have two options to consider as follows:

- An optimal classifier which is a classifier that leads to the smallest CST value under a specified cost ratio range [r_1, r_2]. This scenario takes place when cost curves overlap.
- A robust classifier which is a classifier that has a stable and good performance throughout the cost ratio changes. "Under any target cost and class distributions, a robust classifier will perform at least as well as the best classifier for those conditions". [4] This scenario takes place when a cost curve dominates others.

Thus, selecting an optimal or robust classifier from the MTMCS chart requires the following steps:

- Defining the cost ratio range [r_1, r_2] to specify the area of interest in the MTMCS chart. This is done by consulting with the client or professionals to acquire the cost estimations.
- If there is no robust classifier and an overlap cost curves scenario occurs, then we need to use the CST measure to select an optimal classifier.
- Determining the operating range for classification threshold cutoffs from the Optimal Classification Thresholds chart, which will help in the real-time classification.

5.2 The First Experiment: German Credit Dataset

The total number of instances in the German Credit dataset is 1000 instances, with 17 attributes. In Figure 10, we can see that the data distribution is unbalanced.

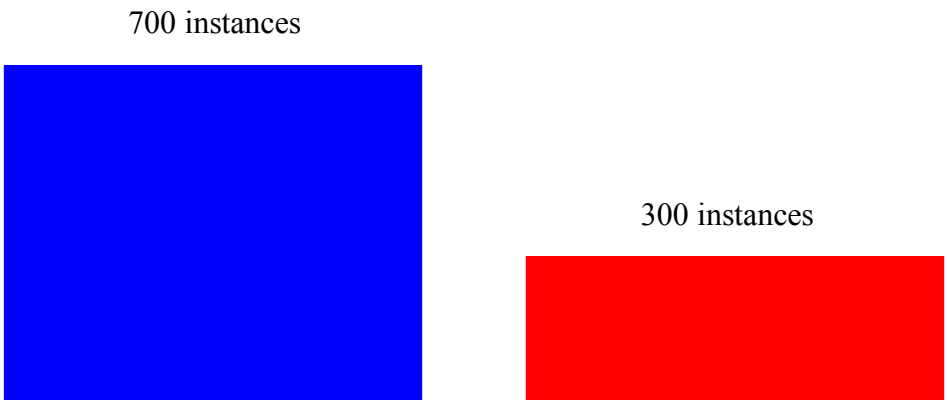

Figure 10. This figure shows the unbalanced data distribution of the German Credit dataset.

From Figure 11, we can see that the Naïve Bayes classifier performs best among other classifiers across all cost ratios. Therefore, we can choose it as the robust classifier for this dataset.

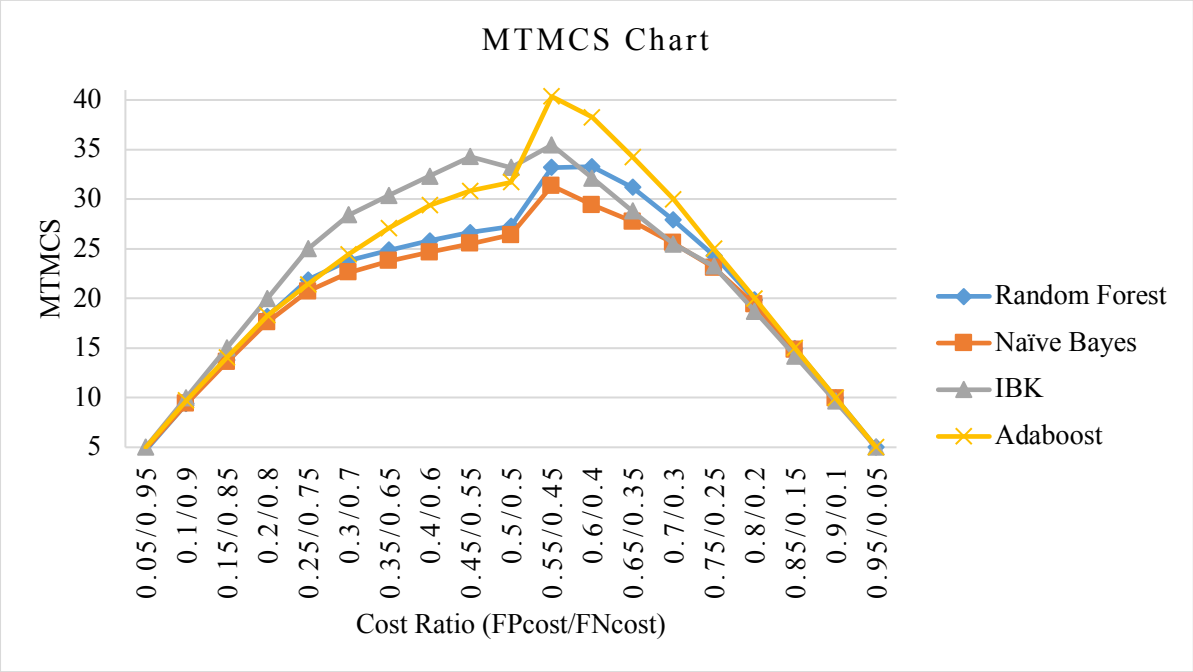

Figure 11. This chart shows the performance of the four classifiers applied to the German Credit dataset, by using the cost of misclassification as an evaluative criterion.

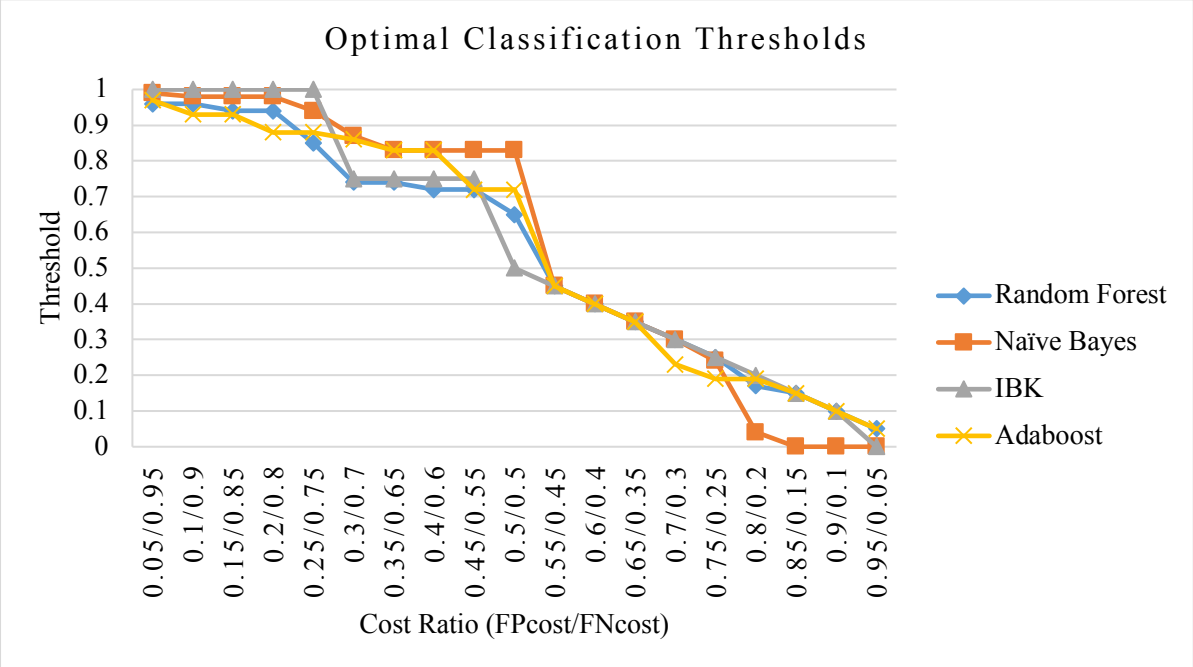

Figure 12. This chart shows the optimal classification thresholds of the four classifiers applied to the German Credit dataset at each cost ratio.

From Figure 13, it is difficult to make a decisive decision about which classifier we should choose. This is because the Random Forest and Adaboost curves overlap each other. The Naïve Bayes is closer to the northwest corner of the ROC graph, while the AUC value of the Random Forest classifier is slightly bigger.

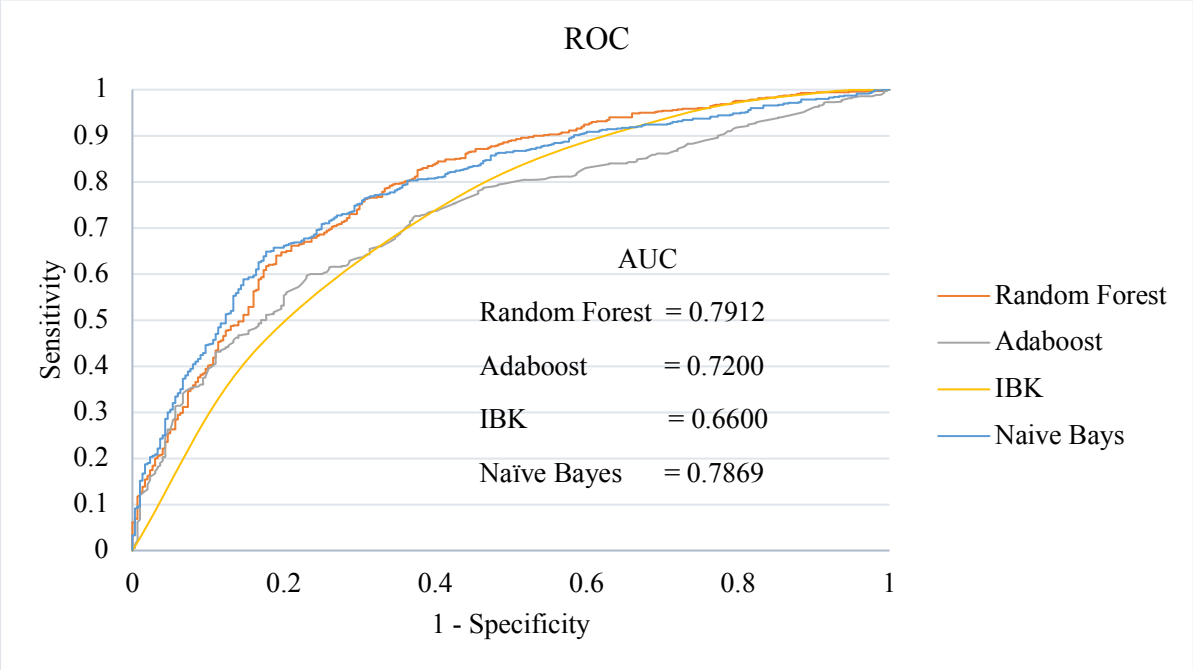

Figure 13. This is the ROC graph. It shows the performance of the four classifiers applied to the German Credit dataset.

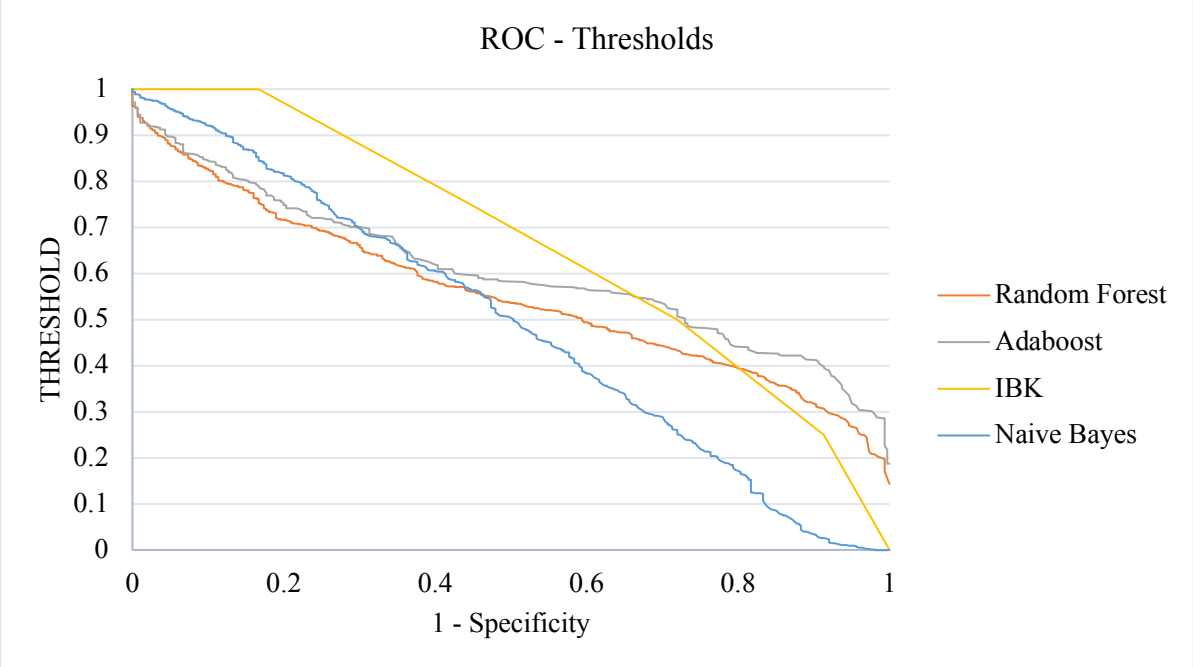

Figure 14. This chart shows the classification thresholds of the four classifiers applied to the German Credit dataset using the ROC graph.

5.3 The Second Experiment: Credit Approval Dataset

The total number of instances in the Credit Approval dataset is 690 instances, with 16 attributes. In Figure 15, we can see that its data distribution is relatively balanced.

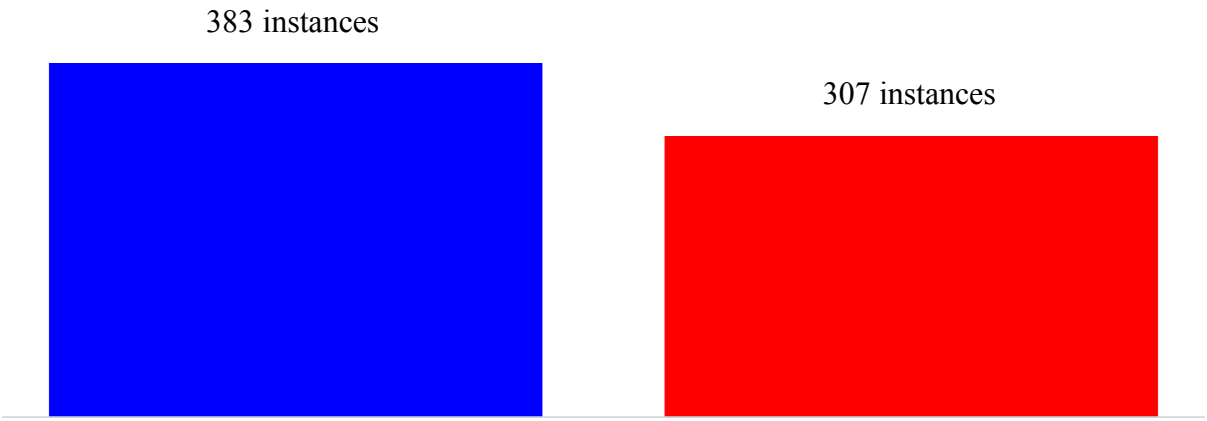

Figure 15. This figure shows the relatively balanced data distribution of the Credit Approval dataset.

In Figure 16, if the FNcost errors are more expensive than the FPcost errors, then the best classifier would be the Adaboost. But if the FPcost errors are more expensive than the FNcost errors, then the Random Forest classifier would be the best classifier to choose for this dataset.

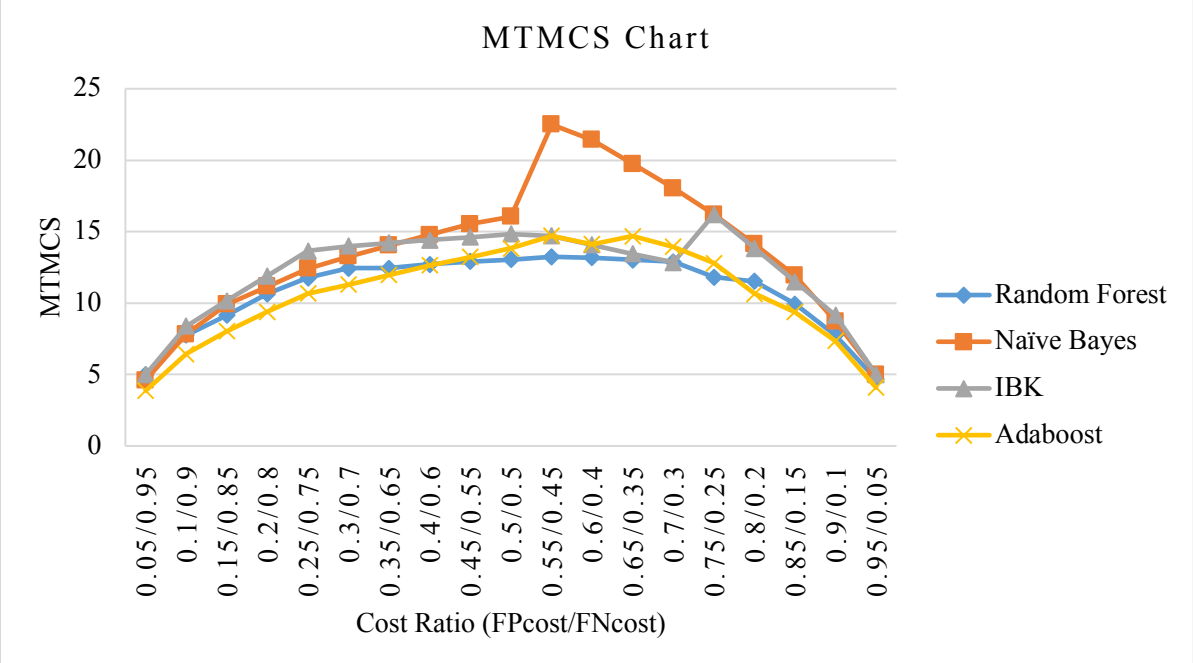

Figure 16. This chart shows the performance of the four classifiers applied to the Credit Approval dataset, by using the cost of misclassification as an evaluative criterion.

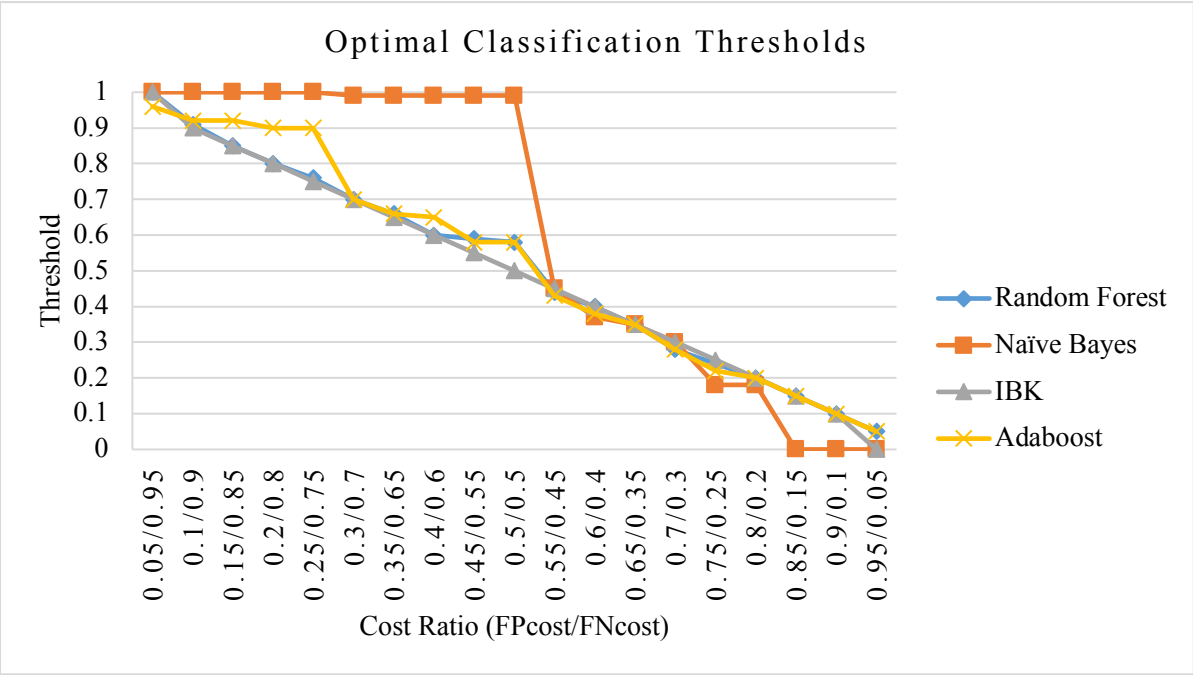

Figure 17. This chart shows the optimal classification thresholds of the four classifiers applied to the Credit Approval dataset at each cost ratio.

In Figure 18, it is difficult to make a decisive decision about which classifier we should choose. This is because the AUC values of Random Forest and Adaboost classifiers are almost the same, and both classifiers are close to the northwest corner of the ROC graph.

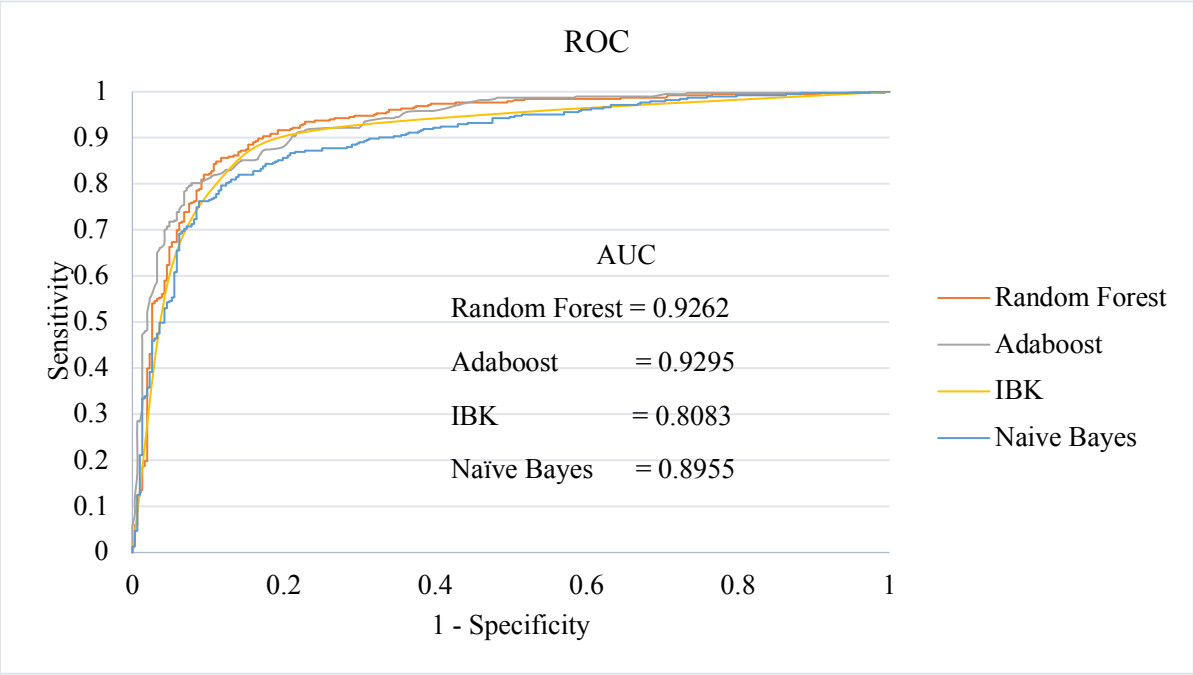

Figure 18. This is the ROC graph. It shows the performance of the four classifiers applied to the Credit Approval dataset.

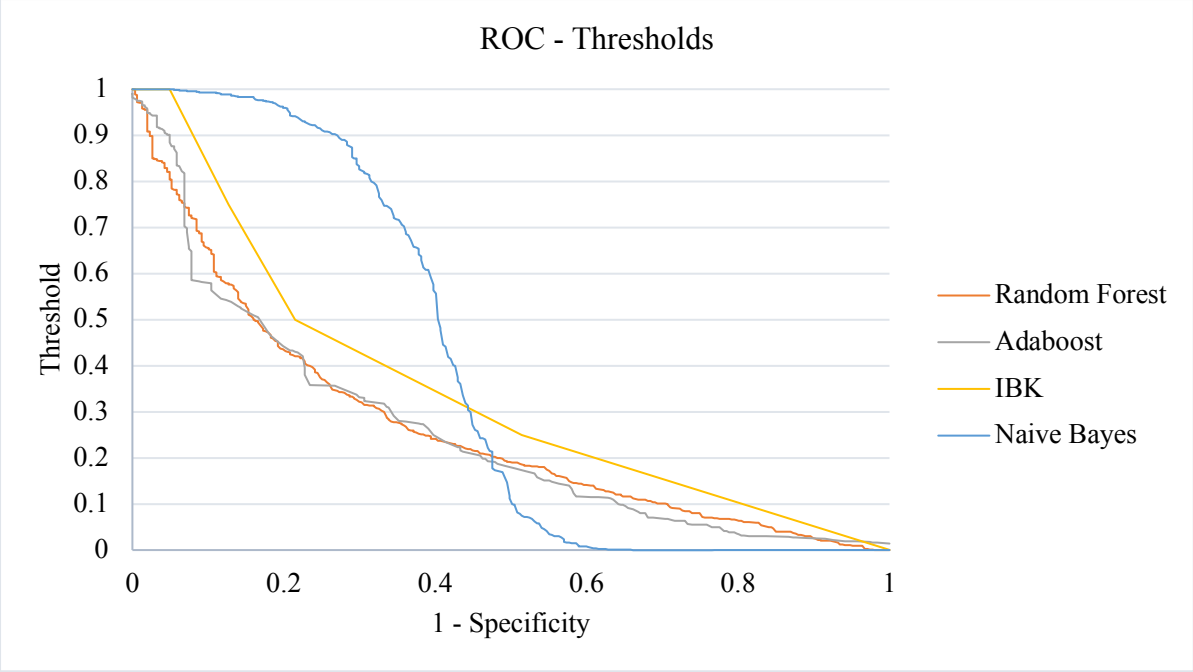

Figure 19. This chart shows the classification thresholds of the four classifiers applied to the Credit Approval dataset using the ROC graph.

5.4 The Third Experiment: Messidor Features (Diabetic Retinopathy Debrecen) Dataset

This dataset contains features extracted from the Messidor image set to predict whether an image contains signs of diabetic retinopathy or not. The total number of instances in the Messidor Features dataset is 1151 instances, with 20 attributes. In Figure 20, we can see that its data distribution is relatively balanced.

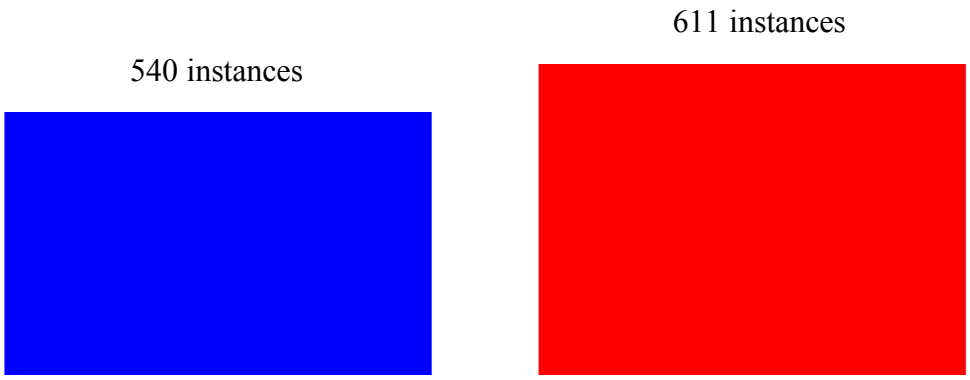

Figure 20. This figure shows the relatively balanced data distribution of the Messidor Features dataset.

In Figure 21, the Random Forest classifier dominates all other classifiers across all cost ratios. Therefore, we would choose it as the robust classifier for this dataset.

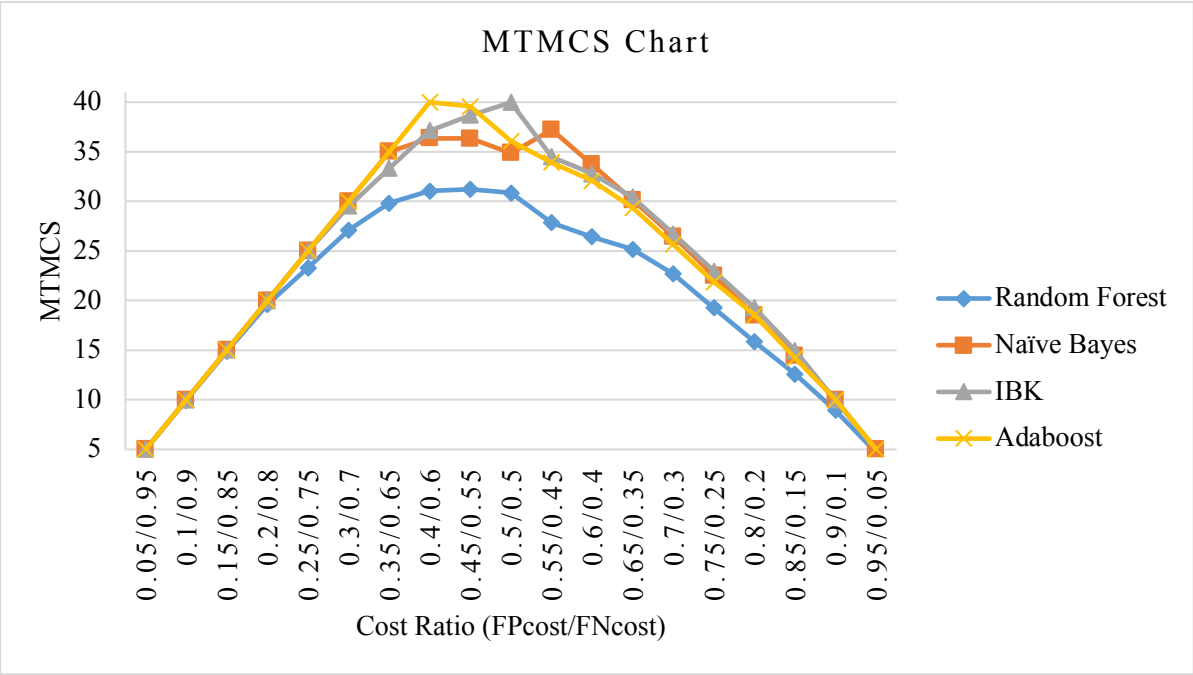

Figure 21. This chart shows the performance of the four classifiers applied to the Messidor Features dataset, by using the cost of misclassification as an evaluative criterion.

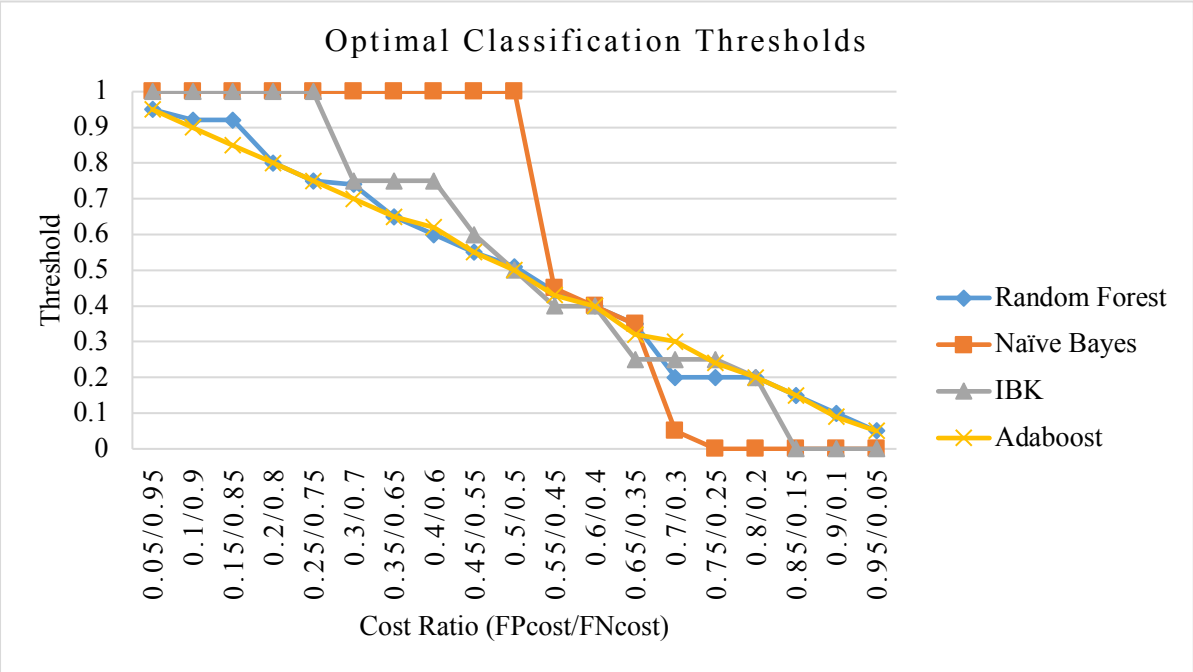

Figure 22. This chart shows the optimal classification thresholds of the four classifiers applied to the Messidor Features dataset at each cost ratio.

Similarly, in Figure 23, the Random Forest classifier dominates all other classifiers across all conditions. Therefore, we can make a decisive decision for this dataset and choose the Random Forest classifier as the robust classifier.

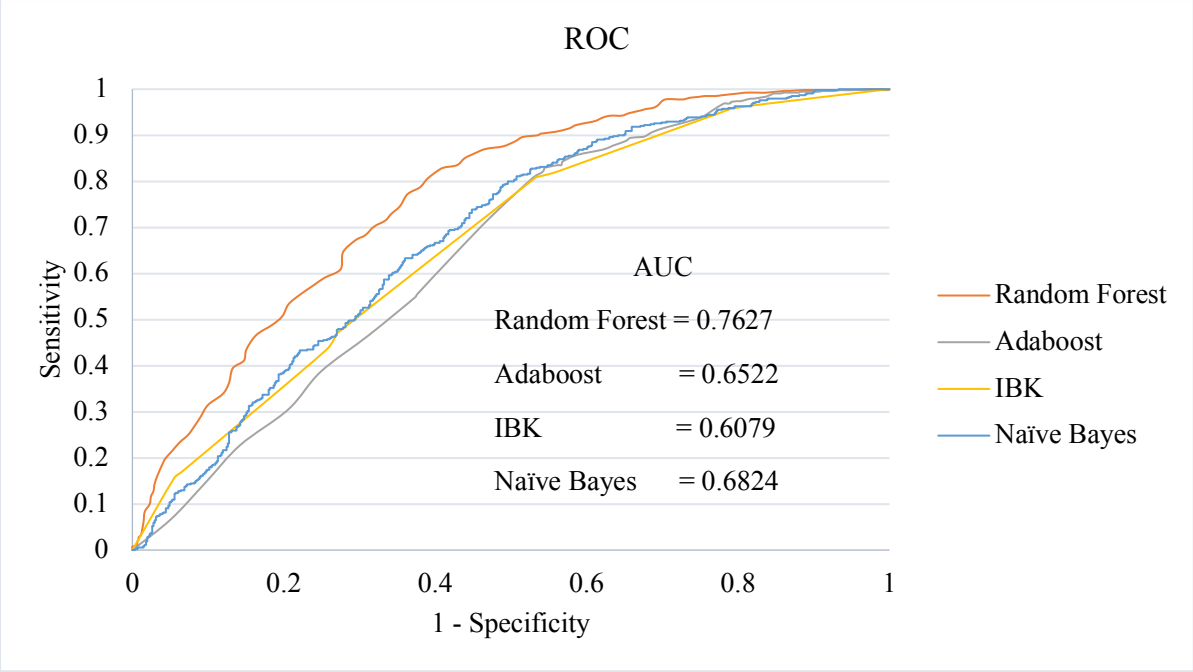

Figure 23. This is the ROC graph. It shows the performance of the four classifiers applied to the Messidor Features dataset.

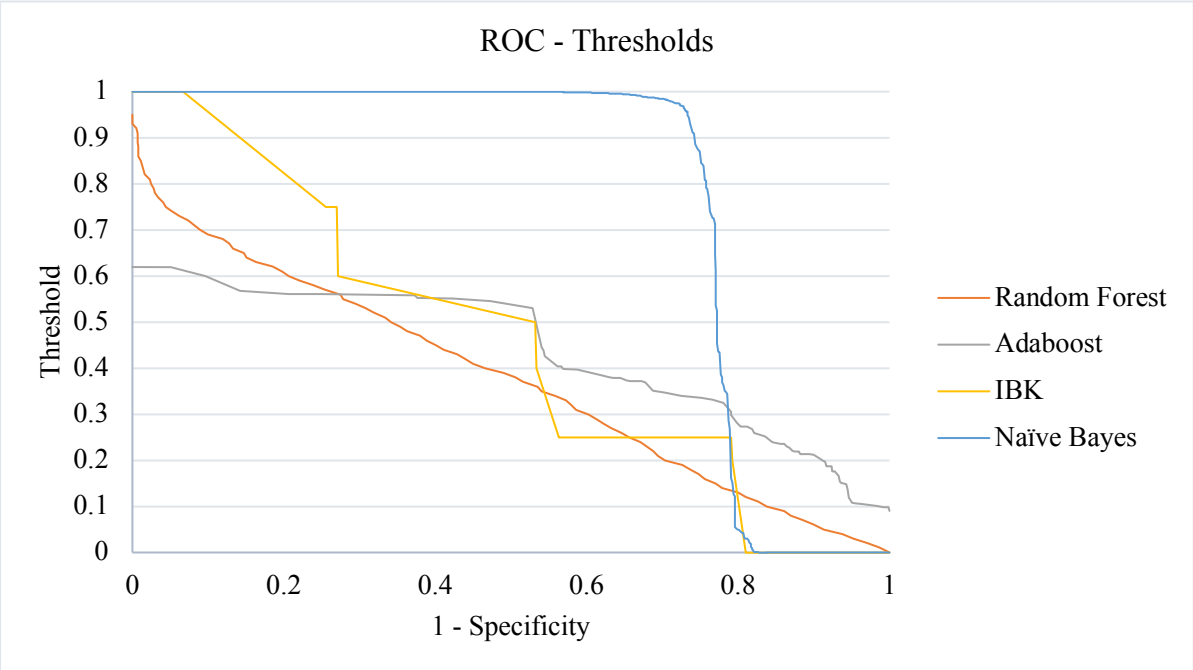

Figure 24. This chart shows the classification thresholds of the four classifiers applied to the Messidor Features dataset using the ROC graph.

5.5 The Fourth Experiment: SPECTF Heart Dataset

The data were derived from the cardiac Single Proton Emission Computed Tomography (SPECT) images. Each patient is classified into two categories: normal and abnormal. The total number of instances in the SPECTF dataset is 267 instances, with 44 attributes. In Figure 25, we can see that its data distribution is skewed.

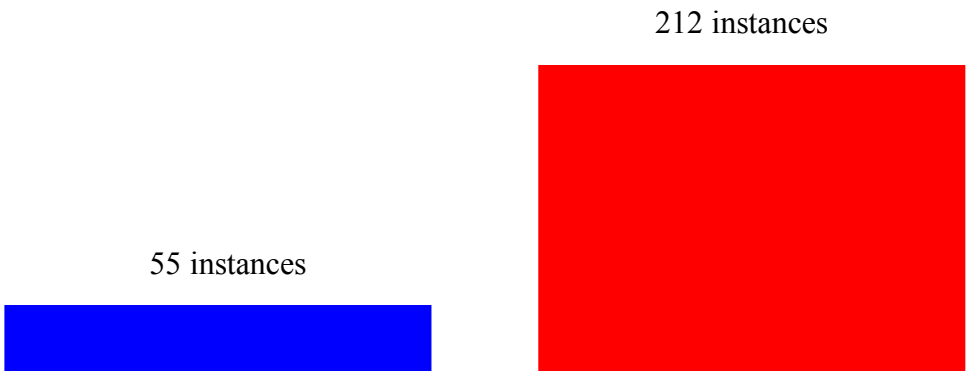

Figure 25. This figure shows the skewed data distribution of the SPECTF Heart dataset.

In Figure 26, if the FNcost errors are more expensive than the FPcost errors, then the best classifier would be Naïve Bayes. But if the FPcost errors are more expensive than the FNcost errors, then the Random Forest classifier would be the best classifier to choose.

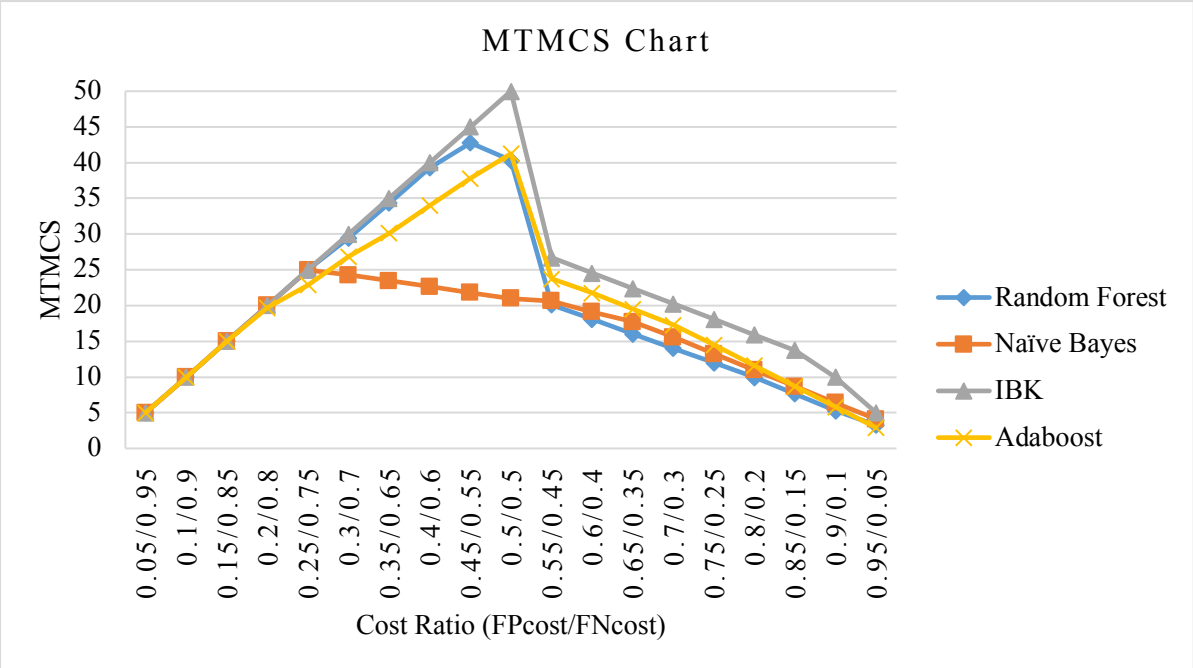

Figure 26. This chart shows the performance of the four classifiers applied to the SPECTF Heart dataset, by using the cost of misclassification as an evaluative criterion.

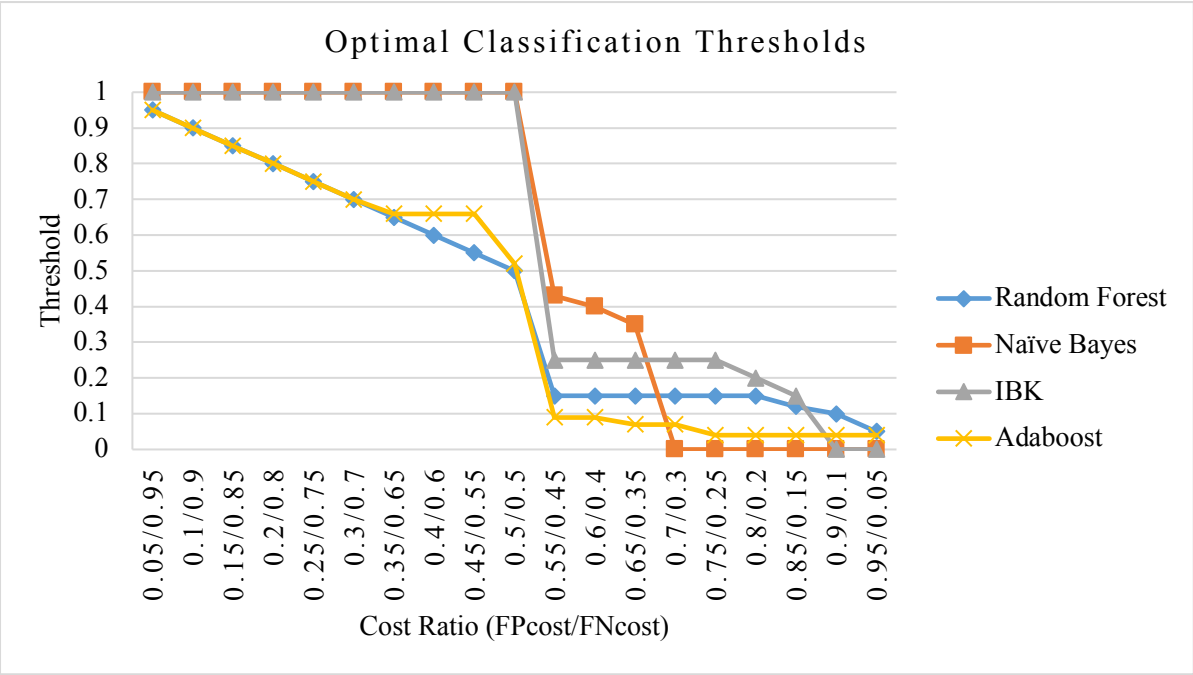

Figure 27. This chart shows the optimal classification thresholds of the four classifiers applied to the SPECTF Heart dataset at each cost ratio.

In Figure 28, the Naïve Bayes classifier has the biggest AUC value. However, the Random Forest classifier is closer to the northwest corner of the ROC graph. Therefore, we are not confident which one to choose as the optimal classifier for this dataset.

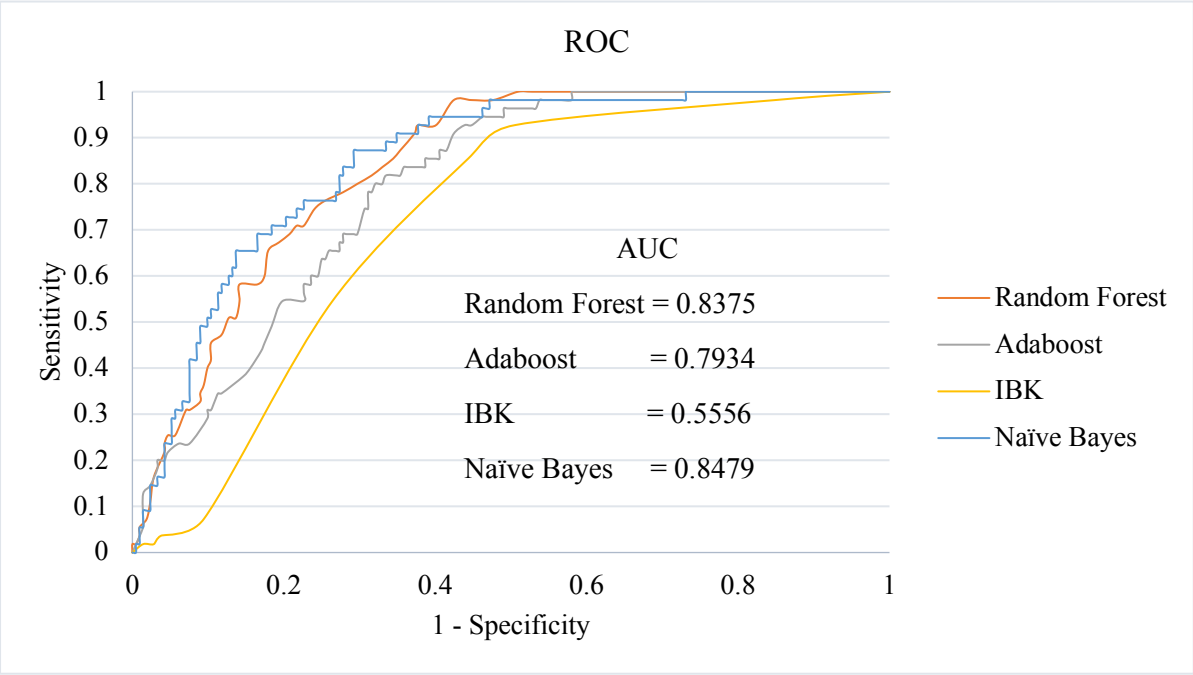

Figure 28. This is the ROC graph. It shows the performance of the four classifiers applied to the SPECTF Heart dataset.

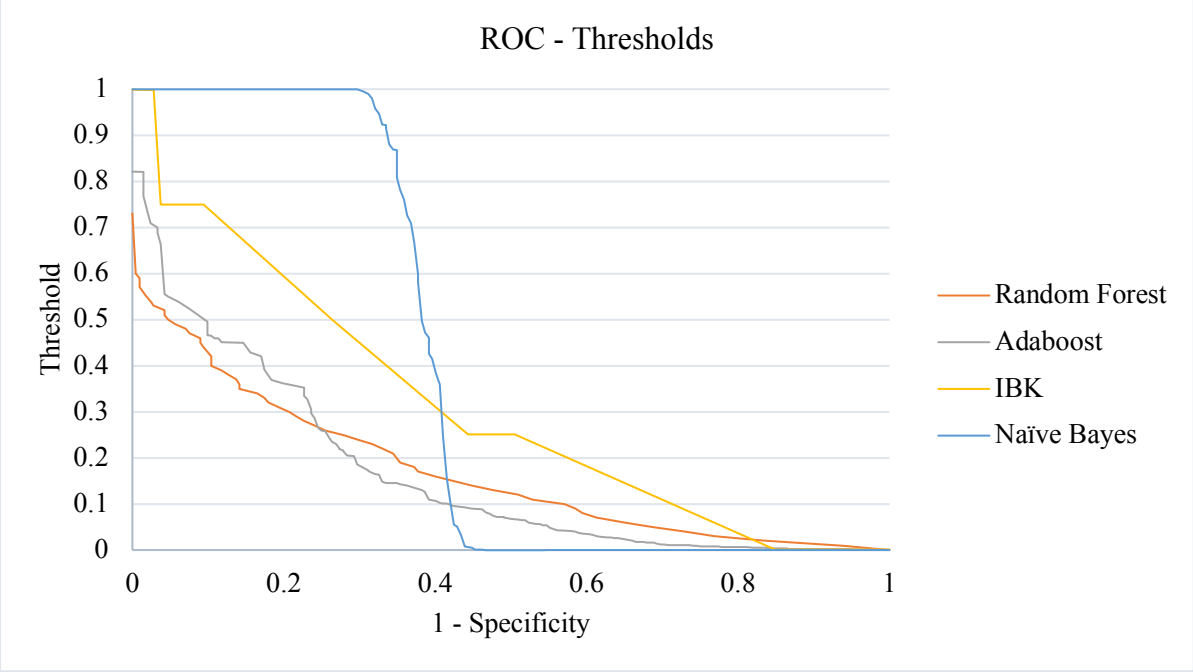

Figure 29. This chart shows the classification thresholds of the four classifiers applied to the SPECTF Heart dataset using the ROC graph.

5.6 The Experiments Summary

In Table 6, we are summarizing the details of the four datasets we used in the four experiments.

Table 6. This table shows a summary of the four datasets used in our experiments.

Exp.	Dataset	No. of Instances	Distribution
1	German Credit card	1000	Unbalanced
2	Credit Approval	690	Relatively balanced
3	Messidor Features	1151	Relatively balanced
4	SPECTF Heart	267	Skewed

Next, we compare the results of classification of four classifiers between the ROC graph and the MTMCS chart. We made four different comparisons with the ROC graph as follows:

- Considering the whole cost curves of the MTMCS chart.
- Considering only the left-hand side of the MTMCS chart where the FNcost is more expensive.

- Considering only the right-hand side of the MTMCS chart where the FPcost is more expensive.
- Considering the middle point of the MTMCS chart where the FPcost is equal to the FNcost.

5.6.1 Comparing the ROC Graph Results with the Whole Cost Curves of the MTMCS Chart

In Table 7, we are considering the whole cost curves in the MTMCS chart without specifying any cost range to compare it with the ROC curves. We can see that the results are the same between the two approaches.

Table 7. This table shows the comparison of the four experiments between the results of the MTMCS chart (by considering the whole cost curves) and the results of the ROC graph. * We selected the Naïve Bayes classifier as the optimal one in the ROC graph of the first experiment because it is closer to the northwest corner.

Exp.	Best Classifiers Candidates				Final Classifier Selection		Same Result
	MTMCS Chart	MTMCR	ROC	AUC	MTMCS Chart	ROC	
1	Random Forest	396.09	Random Forest	0.7912	Naïve Bayes *		Yes
	Naïve Bayes	375.39	Naïve Bayes	0.7869			
2	Adaboost	203.05	Adaboost	0.9295	Adaboost		Yes
	Random Forest	205.9	Random Forest	0.9262			
3	Random Forest	386.2	Random Forest	0.7627	Random Forest		Yes
4	Naïve Bayes	304.65	Naïve Bayes	0.8479	Naïve Bayes		Yes

5.6.2 Comparing the ROC Graph Results with the Left-side of the MTMCS Chart

In Table 8, we are only considering the left-hand side of the MTMCS chart (where the cost of misclassification varies between the range of [0.05/0.95, 0.45/0.55]) to compare it with the ROC curves. Thus, we are considering the cost range where the FNcost is more expensive than the FPcost. Again, we can see that the results are the same between the two approaches.

Table 8. This table shows the comparison of the four experiments between the results of the MTMCS chart (by considering only its left-hand side) and the results of the ROC. * We selected the Naïve Bayes classifier as the optimal one in the ROC graph of the first experiment because it is closer to the northwest corner.

Exp.	Best Classifiers Candidates				Final Classifier Selection		Same Result
	MTMCS Chart	MTMCR	ROC	AUC	MTMCS Chart	ROC	
1	Random Forest	169.12	Random Forest	0.7912	Naïve Bayes *		Yes
	Naïve Bayes	162.72	Naïve Bayes	0.7869			
2	Adaboost	87.52	Adaboost	0.9295	Adaboost		Yes
	Random Forest	94.83	Random Forest	0.9262			
3	Random Forest	191.89	Random Forest	0.7627	Random Forest		Yes
4	Naïve Bayes	167.16	Naïve Bayes	0.8479	Naïve Bayes		Yes

5.6.3 Comparing the ROC Graph Results with the Right-side of the MTMCS Chart

In Table 9, we are only considering the right-hand side of the MTMCS chart (where the cost of misclassification varies between the range of [0.55/0.45, 0.95/0.05]) to compare it with the ROC curves. Thus, we are considering the cost range where the FPcost is more expensive than the FNcost. We can see that the results are different in the 2nd and the 4th experiments. In the 1st and the 3rd experiments, the classifiers selected are dominating in both charts across all conditions; therefore, their results did not change.

5.6.4 Comparing the ROC Graph Results with the Middle Point of the MTMCS Chart

In Table 10, we are only considering the middle point of the MTMCS chart (where the cost of misclassification is equal for both classes) to compare it with the ROC curves. We can see that the result is different in the 2nd experiment between the two approaches.

Table 9. This table shows the comparison of the four experiments between the results of the MTMCS chart (by considering only its right-hand side) and the results of the ROC. * We selected the Naïve Bayes classifier as the optimal one in the ROC graph of the first experiment because it is closer to the northwest corner.

Exp.	Best Classifiers Candidates				Final Classifier Selection		Same Result
	MTMCS Chart	MTMCR	ROC	AUC	MTMCS Chart	ROC	
1	IBK	192.74	Random Forest	0.7912	Naïve Bayes *		Yes
	Naïve Bayes	186.27	Naïve Bayes	0.7869			
2	Adaboost	101.7	Adaboost	0.9295	Random Forest	Adaboost	No
	Random Forest	98.03	Random Forest	0.9262			
3	Random Forest	163.45	Random Forest	0.7627	Random Forest		Yes
4	Random Forest	106.32	Naïve Bayes	0.8479	Random Forest	Naïve Bayes	No

Table 10. This table shows the comparison of the four experiments between the ROC results and the equal cost part of the cost curves in the MTMCS chart. * We selected the Naïve Bayes classifier as the optimal one in the ROC graph of the first experiment because it is closer to the northwest corner.

Exp.	Best Classifiers Candidates				Final Classifier Selection		Same Result
	MTMCS Chart	MTMCS	ROC	AUC	MTMCS Chart	ROC	
1	Random Forest	27.26	Random Forest	0.7912	Naïve Bayes *		Yes
	Naïve Bayes	26.4	Naïve Bayes	0.7869			
2	Adaboost	13.83	Adaboost	0.9295	Random Forest	Adaboost	No
	Random Forest	13.04	Random Forest	0.9262			
3	Random Forest	30.86	Random Forest	0.7627	Random Forest		Yes
4	Naïve Bayes	20.99	Naïve Bayes	0.8479	Naïve Bayes		Yes

5.7 The Use of the CST Measure to Compare the Overlapping Cost Curves

The sensitivity to cost ratio changes can be helpful for data scientists to determine the optimal classifier for a specific problem when there is no robust classifier available. For instance, in the first, third, and fourth experiments, all classifiers are sensitive to cost ratio changes. On the contrary, in the second experiment, Random Forest shows low sensitivity to the cost ratio changes in the cost ratio range of [0.30/0.70, 0.70/0.30], which means it has a stable performance under this range of interest. However, sometimes the tradeoff between the MTMCR and the cost sensitivity values is not clear; the CST measure can help us to find this tradeoff. We used these formulas in the next examples and as follows:

$$MTMCR\ ([r_1, r_2]) = \sum_{r=r1}^{r2} MTMCS\ (r)$$

$$Cost\ Sensitivity = Highest\ MTMCS - Lowest\ MTMCS$$

where both MTMCS values are within the cost ratio range of interest $[r_1, r_2]$.

$$CST = MTMCR\ (1 + \frac{Cost\ Sensitivity}{100})$$

where 100 is the maximum possible value of the MTMCS.

5.7.1 An Example of Two Sensitive Classifiers Scenario

In the third experiment (Messidor Features dataset), what if the client was estimating that the cost ratio range of interest is between [0.40/0.60, 0.60/0.40]? For the sake of explaining the two sensitive classifiers scenario, we will consider from experiment 5.4 only the Naïve Bayes and Adaboost classifiers and disregard the other classifiers. In Figure 30, we can see that we have two classifiers (Naïve Bayes and Adaboost) where their cost curves overlap under the cost ratio range of [0.40/0.60, 0.60/0.40].

Table 12 shows the performance evaluation of the Naïve Bayes and Adaboost classifiers. We used the MTMCS values from Table 11 to derive the MTMCR, cost sensitivity,

and CST values of Table 12. We can see that the Naïve Bayes has the lowest MTMCR, cost sensitivity, and CST values. Hence, the overall performance of the Naïve Bayes classifier in the cost ratio range of interest is clearly better.

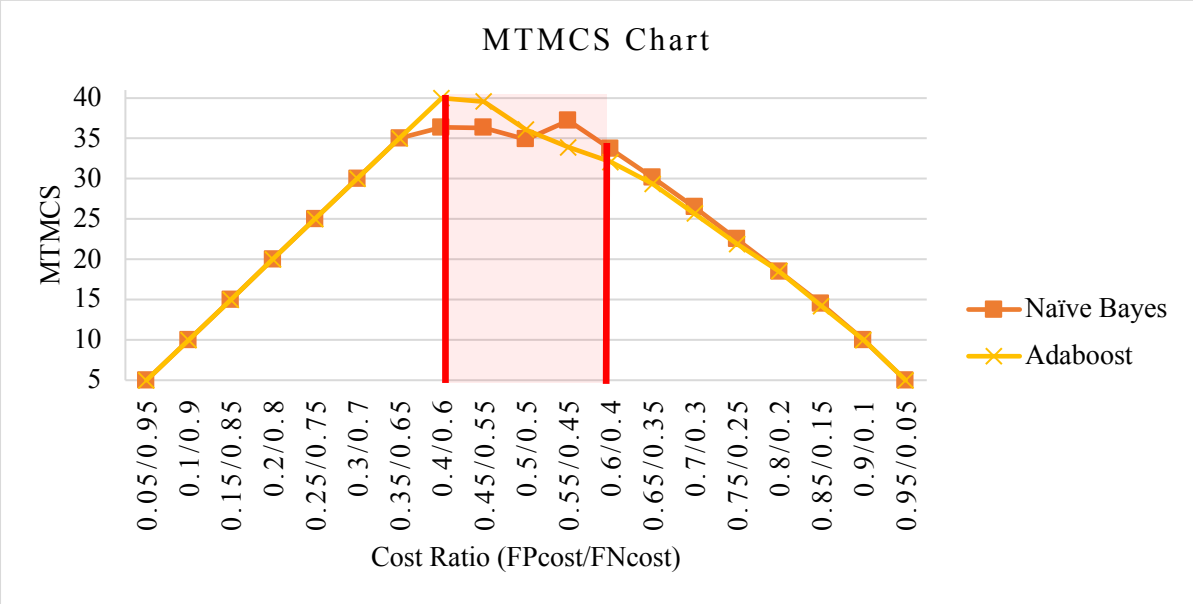

Figure 30. This chart shows the performance of two classifiers applied to the Messidor Features dataset. The highlighted red area is the cost ratio range of interest.

Table 11. This table shows the MTMCS values at each cost ratio for the Naïve Bayes and the Adaboost classifiers from Figure 30.

Cost Ratio	0.40/0.60	0.45/0.55	0.50/0.50	0.55/0.45	0.60/0.40
MTMCS of Adaboost	40	40	36	34	32
MTMCS of Naïve Bayes	37	36	36	37	32

Table 12. This table shows the MTMCR, cost sensitivity, and the CST values for the Naïve Bayes and Adaboost classifiers in Figure 30.

	MTMCR	Cost Sensitivity	CST
Adaboost	182	8	= 196.56
Naïve Bayes	178	5	= 186.90

5.7.2 An Example of the One Sensitive Classifier Scenario

In the second experiment (Credit Approval dataset), what if the client was estimating that the cost ratio range of interest is between [0.25/0.75, 0.50/0.50]? For the sake of explaining the one sensitive classifier scenario, we will consider from experiment 5.3 only the Adaboost and Random Forest classifiers and disregard the other classifiers. In Figure 31, we can see that

we have two classifiers (Adaboost and Random Forest) where their cost curves overlap under the cost ratio range of [0.25/0.75, 0.50/0.50].

Table 14 shows the performance evaluation of the Adaboost and Random Forest classifiers. We used the MTMCS values from Table 13 to derive the MTMCR, cost sensitivity, and CST values of Table 14. We can see that the Adaboost classifier has the lowest MTMCR value, but with a higher cost sensitivity than the Random Forest classifier. Therefore, we need to use the CST measure to find the tradeoff between the MTMCR and the cost sensitivity values. The CST measure shows that even if the Adaboost classifier has a higher cost sensitivity, it still performs better than the Random Forest classifier which has a smaller cost sensitivity but a higher MTMCR value. Hence, the overall performance of the Adaboost classifier in the cost ratio range of interest is clearly better based on the result of the CST measure.

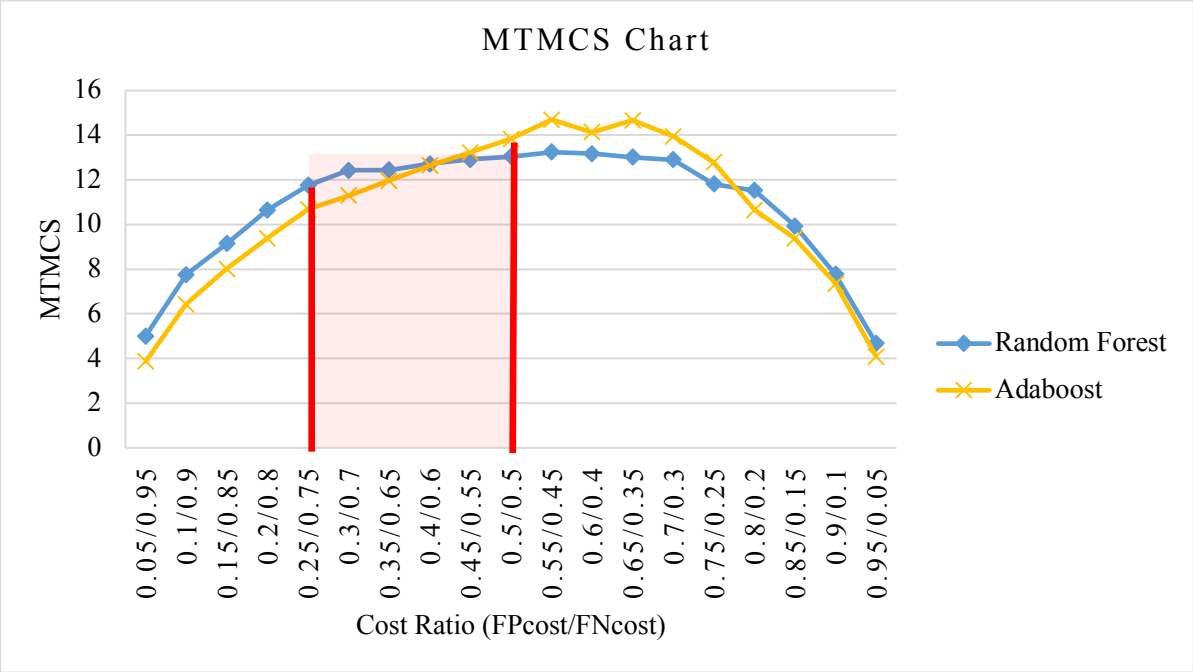

Figure 31. This chart shows the performance of two classifiers applied to the Credit Approval dataset. The highlighted red area is the cost ratio range of interest.

Table 13. This table shows the MTMCS values at each cost ratio for the Adaboost and Random Forest classifiers from Figure 31.

Cost Ratio	0.25/0.75	0.30/0.70	0.35/0.65	0.40/0.60	0.45/0.55	0.50/0.50
MTMCS of Adaboost	10	11	12	13	13.5	14
MTMCS of Random Forest	12	13	13	13	13	13

Table 14. This table shows the MTMCR, cost sensitivity, and the CST values for the Adaboost and Random Forest classifiers in Figure 31.

	MTMCR	Cost Sensitivity	CST
Adaboost	73.5	4	= 76.44
Random Forest	77.0	1	= 77.77

5.7.3 An Example of a Deceiving MTMCR Value Scenario

In the first experiment (German credit dataset), what if the client was estimating that the cost ratio range of interest is between [0.35/0.65, 0.55/0.45]? For the sake of explaining how the MTMCR value can be deceiving, we will consider from experiment 5.2 only the Adaboost and IBK classifiers and disregard the other classifiers. In Figure 32, we can see that we have two classifiers (IBK and Adaboost) where their cost curves overlap under the cost ratio range of [0.35/0.65, 0.55/0.45].

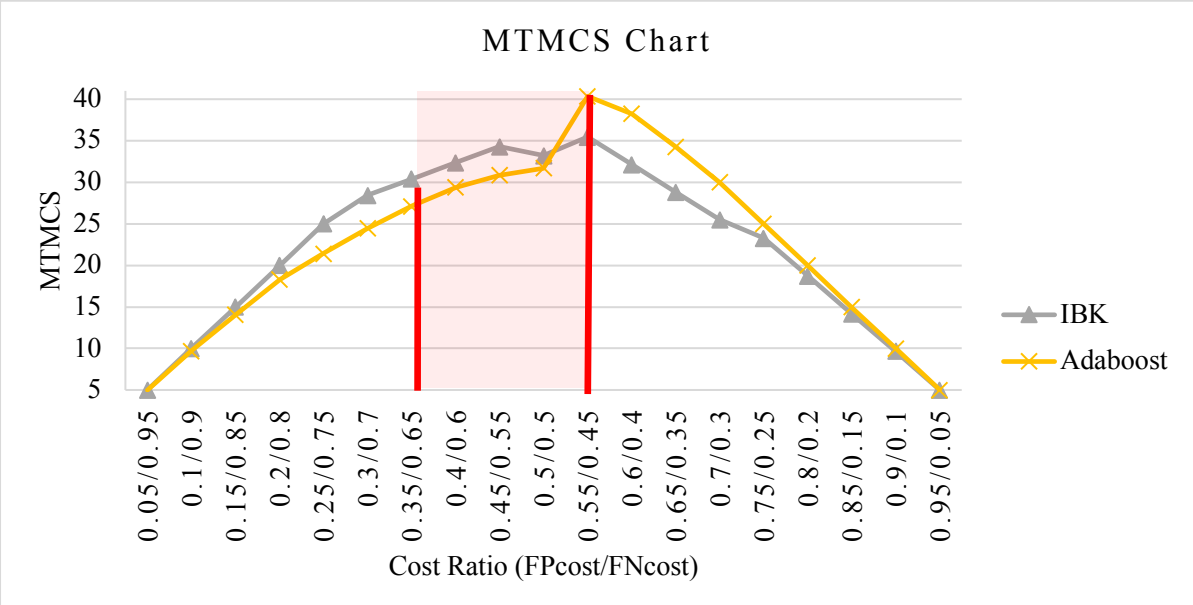

Figure 32. This chart shows the performance of two classifiers applied to the German credit dataset. The highlighted red area is the cost ratio range of interest.

Table 16 shows the performance evaluation of the IBK and Adaboost classifiers. We used the MTMCS values from Table 15 to derive the MTMCR, cost sensitivity, and CST values of Table 16. The overall performance of the Adaboost classifier in the cost ratio range of interest is clearly better because it has the smallest MTMCR value. Therefore, we would select the Adaboost classifier as the optimal one. However, the Adaboost classifier is much more sensitive to cost ratio changes than the IBK classifier. Thus, choosing the IBK as the optimal classifier can be a good choice since its sensitivity is less than the Adaboost's to the cost ratio changes in this cost ratio range of interest. We are not definite it is the best choice, though! The CST measure can resolve this doubt. The CST value of the IBK classifier is the smallest, therefore, we recommend selecting it as the optimal classifier for this problem.

Table 15. This table shows the MTMCS values at each cost ratio for IBK and Adaboost classifiers from Figure 32.

Cost Ratio	0.35/0.65	0.40/0.60	0.45/0.55	0.50/0.50	0.55/0.45
IBK MTMCS	30	32	34	32	35
Adaboost MTMCS	27	29	31	32	40

Table 16. This table shows the MTMCR, cost sensitivity, and the CST values for IBK and Adaboost classifiers in Figure 32.

	MTMCR	Cost Sensitivity	CST
IBK	163	5	= 171.15
Adaboost	159	13	= 179.67

5.8 Discussion

After examining the results from our four experiments and its summary, one can conclude that the proposed approach is better than the ROC to compare the classifiers' performances because of the following reasons:

- The ROC result is embedded and shown in our MTMCS chart at the cost ratio 0.50/0.50. We are not only considering the traditional assumption of equal costs for misclassification, but also both possibilities when FPcost and FNcost are different. As a result, we show a better representation of real-world circumstances.

- In Table 7 where we compare the results of the MTMCS chart (complete curves) with the results of the ROC, we see that they are very similar. Also, we see the same conclusion in Table 8, where we only consider the left-side of the MTMCS chart to compare it with the results of the ROC. However, the difference here is that by using the MTMCS chart, we are confident when we select a classifier as the optimal or the robust choice and under which conditions. Whilst in the ROC, the classifier selection is not clear and depends on the AUC.
- In our approach, we are using the cost sensitivity as a quantified value to help compare the cost curves and to help determine classifiers' performances when the cost curves overlap. Whilst the ROC cannot do this by itself. It needs the ISO performance lines to provide an estimation of the cost sensitivity.
- Usually, cost-sensitive classification is used with unbalanced data distributions. However, as we saw in Table 9, regardless to the data distribution, when we considered the scenario where the FP errors are more expensive than the FN errors, the final decision of which classifier is best has changed in the 2nd and 4th experiments. The 2nd experiment has relatively balanced data distribution and the 4th has a skewed data distribution. Therefore, we believe that cost-sensitive classification must be used most of the time.
- When a classifier dominates across all the conditions, then regardless of which class is more important, the result will not change by changing the cost ratios. We can see this in the 1st and 3rd experiments.
- We are using a new measure (the CST measure) to compare the overlapped cost curves by incorporating both the MTMCR and the cost sensitivity values, instead of the widely used but flawed AUC. The CST measure provides the tradeoff between the MTMCR

and the cost sensitivity values. While in the ROC, we need to make an estimation to select the best classifier if there is no dominant classifier.

- The MTMCS chart is customizable, unlike the ROC graph. The user can define the X-axis' range and its step size to represent the cost ratio range of interest. This is a useful feature to reduce the required processing time. There is no need to calculate the MTMCS values for cost ratios in which the user has no interest.
- Our classification threshold chart is better than the ROC's threshold chart because we show only the thresholds where a minimum TMC (associated with each cost ratio) is reached. Thus, we show the classification thresholds where each classifier performs best.

Based on that, we can say that our approach enables us to firmly make a decision about which classifier is the best classifier for the problem at hand. Unlike the ROC, where we need to make an estimation when two or more classifiers overlap.

6. CONCLUSIONS

One of the main motivations to start working for this thesis was to replace the ROC with a better chart to evaluate classifiers' performances. Even though the ROC is one of the most used graphs to compare classification algorithms performances, it has shortcomings. These weaknesses are embodied by the assumptions of equal costs for all errors, and the use of the AUC to select the best classifier.

First, the ROC does not consider variations of misclassification costs as an evaluation measure and instead assumes a rare scenario of equal costs for all misclassification errors. Furthermore, only improved versions of the ROC that uses the ISO performance lines can give an estimation of classifiers' cost sensitivity. Therefore, we need a readable chart that enables us to make a quantitative comparison among classifiers. A comparison that considers various cost ratios because the performance of each classifier may differ according to the change in the cost ratio. Such chart can provide more realistic and precise results than the one derived from the ROC.

Another shortcoming of the ROC is its reliance on the AUC. The area under the curve result is reliable only when a classifier is dominant throughout all thresholds, which is a rare scenario. The AUC fails when the ROC curves overlap because it is possible for a classifier to perform better under specific conditions and perform worse under others. Therefore, we tried to overcome the problem of overlapped cost curves in our MTMCS chart by proposing a new measure which is called the CST measure. This new measure (CST) combines the MTMCR and the cost sensitivity values. What made this new measure (CST) possible to derive is that our MTMCS chart can provide quantified values for the cost of misclassification and the cost sensitivity. Thus, the CST measure takes the cost sensitivity into consideration along with the MTMCR values to add the extra cost for the cost range of interest when having a cost-sensitive classifier.

Regarding the cost of misclassification, we believe that the classifier that leads to the minimum total misclassification cost must be considered as with superior performance. However, considering the cost of misclassification as an evaluative criterion has its own downside as well. This is because the cost associated with a class is not equal to the cost associated with each instance within a class. In cost-sensitive classification, we are assigning equal costs for all the instances that belong to a specific class which is not a precise treatment. The cost of each misclassified instance could probably be different than the others.

Furthermore, we must mention that determining misclassification cost ratios and even having a cost range is not an easy task to accomplish. Cost estimations are a subjective matter and rely on the problem itself. This notion applies to fields where specifying cost of misclassification is difficult such as the medical field, and even in fields where we may think defining the cost can be an easy task such as the financial services. [9]

On another matter, the classification threshold operating range cannot be elicited easily neither from the ROC graph nor from our MTMCS chart. For both approaches, we need a secondary chart to map the classification thresholds. However, what makes our classification threshold chart better is that we select the thresholds that lead to the MTMCS values associated with each cost ratio. On top of that, determining the classification threshold operating range is very important because it is where a classifier performs best and without it the classifier performance will be weak. [5] In other words, a classifier can be optimal only under specific conditions.

Regarding the required processing time, since we are using many cost ratios in our approach, we must repeat the classification process many times with every cost ratio. This is because we need to find the optimal classification threshold for each cost ratio r, which as a result, will add more CPU time. Unlike the ROC, because it only assumes one cost ratio

scenario (equal FPcost and FNcost). To minimize the required CPU time in our approach, we optimized the search for the optimal classification thresholds. There is a downside for this method though. When FNcost ≥ 0.5 we assume that the optimal classification threshold is closer to 1. Similarly, when FPcost > 0.5 then we assume that the optimal classification threshold is closer to 0. Since we are not searching through all the range [0, 1], the selected classification threshold as the optimal one at cost ratio r is an estimation. It is an estimation because maybe a better optimal classification threshold can be found by using the thorough search method. However, we do not believe that a thorough search to find an optimal classification threshold would lead to a much better result.

Our main contributions in this thesis are as follows:

- Proposing and using easy measures to replace the AUC. The MTMCR value is an estimation of the AUC but it can be calculated for the range of interest instead of the whole area covered by the curve. In addition, the CST measure is very helpful when cost curves overlap because it takes into consideration the cost sensitivity to cost ratio changes along with the MTMCR.
- We selected the best practices from the references we used. For instance, some papers did not use normalized cost ratios, others did not optimize the search of the optimal classification thresholds. Normalized cost ratios are important because it helps compare points on the cost curve from the same classifier to know its cost sensitivity to cost ratio changes. Also, the optimized search for the optimal classification thresholds is used to reduce the required processing time.

REFERENCES

1. R. Kumar and A. Indrayan, "Receiver Operating Characteristic (ROC) Curve for Medical Researchers," *Indian Pediatr.*, vol. 48, no. 4, pp. 277–287, 2011.

2. K. McCarthy, B. Zabar, and G. Weiss, "Does cost-sensitive learning beat sampling for classifying rare classes?," in *Proceedings of the 1st international workshop on Utility-based data mining - UBDM '05*, 2005, pp. 69–77.

3. N. M. Adams and D. J. Hand, "Comparing classifiers when the misallocation costs are uncertain," *Pattern Recognit.*, vol. 32, no. 7, pp. 1139–1147, 1999.

4. F. Provost and T. Fawcett, "Robust classification for imprecise environments," *Mach. Learn.*, vol. 42, no. 3, pp. 203–231, 2001.

5. C. Drummond and R. C. Holte, "What ROC Curves Can't Do (and Cost Curves Can)," *Work. ROC Anal. AI ROCAI*, pp. 19–26, 2004.

6. T. Fawcett, "ROC Graphs : Notes and Practical Considerations for Researchers," *ReCALL*, vol. 31, no. HPL-2003-4, pp. 1–38, 2004.

7. C. Elkan, "The foundations of cost-sensitive learning," in *IJCAI International Joint Conference on Artificial Intelligence*, 2001, pp. 973–978.

8. J. M. Lobo, A. Jiménez-valverde, and R. Real, "AUC: A misleading measure of the performance of predictive distribution models," *Global Ecology and Biogeography*, vol. 17, no. 2. pp. 145–151, 2008.

9. D. J. Hand, "Measuring classifier performance: A coherent alternative to the area under the ROC curve," *Mach. Learn.*, vol. 77, no. 1, pp. 103–123, 2009.

10. Https://En.Wikipedia.Org/Wiki/Confusion_Matrix. Accessed December 20, 2017.

11. S. Halligan, D. G. Altman, and S. Mallett, "Disadvantages of using the area under the receiver operating characteristic curve to assess imaging tests: A discussion and proposal for an alternative approach," *Eur. Radiol.*, vol. 25, no. 4, pp. 932–939, 2015.

VITA

Ramy Al-Saffar is a native of Baghdad, Iraq, received his bachelor's degree in computer science from University of Technology Iraq. He occupied various positions while working in different disciplines such as banking, investment, and oil industry. Afterwards, Ramy was granted the Fulbright scholarship to continue his higher education. He began his studies in the School of Engineering at Louisiana State University majoring in computer science in August of 2015. Ramy anticipates graduating with his master's degree in computer science from Louisiana State University in May 2018. His future technical interest is data analytics.