ELLIPTICAL COST-SENSITIVE DECISION TREE ALGORITHM - ECSDT

MOHAMMAD ELBADAWI KASSIM

SCHOOL OF COMPUTING, SCIENCE AND ENGINEERING
INFORMATICS RESEARCH CENTRE, UNIVERSITY OF SALFORD, MANCHESTER, UK

Submitted in Fulfilment of the Requirements for the Degree of Doctor of Philosophy, 2018

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Acc	Accuracy
AdaBoost	AdaBoost (Adaptive Boosting)
ADTree	Alternating Decision Tree
BFTree	Best-First Tree
BPNN	Back Propagation Neural Network
C.S.C+J48	CostSensitiveClassifier with J48
C.S.C+NBTree	CostSensitiveClassifier with Naive Bayesian Tree
CART	Classification And Regression Trees
CSC	CostSensitive Classifier
CSDTWMCS	Cost-Sensitive Decision Trees with Multiple Cost Scales
CS-ID3	Cost-Sensitive Iternative Dichotomizer 3
CSNB	Cost-Sensitive Naive Bayes
CSNL	Cost-sensitive Non-linear Decision Tree
CSTree	Cost-Sensitive Tree
CV	Cross-Validation
DAGSVM	Directed Acyclic Graph Support Vector Machine
DB2	Divide-by-2
DDAG	Decision Directed Acyclic Graph
DT	Decision Tree
EAs	Evolutionary algorithms
ECCO	Evolutionary Classifier with Cost Optimization
ECM	Expected Cost of Misclassification
ECOC	Error Correcting Output Coding
ECSDT	Elliptical Cost Sensitive Decision Tree
ECSDT +GA	Elliptical Cost-Sensitive Decision Tree with Genetic Algorithm
ECSDT +PSO	Elliptical Cost-Sensitive Decision Tree with Partial Swarm Optimization
Eq	Equation
FDA	Fisher Discriminant Analysis
FN	False Negative
FP	False Positive
GA	Genetic Algorithm
GBSE	Gradient Boosting with Stochastic Ensembles
GDT-MC	Genetic Decision Tree with Misclassification Costs
GP	Genetic Programming
HKWDA	Heteroscedastic Kernel Weighted Discriminant Analysis
ICET	Inexpensive Classification with Expensive Tests
ICF	Information Cost Function
ID3	Iternative Dichotomizer 3
IG	Information Gain
LADTree	Logical Analysis of Data Tree
LSVM	linear support vector machine

MetaCost+J48	MetaCost with J48
MetaCost+NBTree	MetaCost with Naive Bayesian Tree
ML	Machine Learning
MLSVM	Mixture of Linear Support Vector Machines
MNCS_DT	Non-linear Cost-sensitive Decision Tree for Multiclassification
MOEA	Multi-Objective Evolutionary Algorithms
MOOP	Multi-Objective Optimization Problem
NB	Naive Bayesian
NBT	Naive Bayesian Tree
NSGA-II	Nondominated Sorting Genetic Algorithm II
OMOPSO	Optimum Multi-Objective Particle Swarm Optimization
PPV	Positive Predictive Value
PSO	Particle Swarm Optimization
PSOC4.5	Particle Swarm Optimization with C4.5 algorithm
PSO-SVM	Particle Swarm Optimization with Support Vector Machines
RepTree	Reduced Error Pruning Tree
SOM	Self-Organizing Map
SRM	Structural Risk Minimization
SVM	Support Vector Machine
SVM-BDT	Support Vector Machines utilizing Binary Decision Tree
TCSDT	Test Cost-sensitive Decision Tree
TN	True Negative
TP	True Positive
UCI	University of California, Irvine
WEKA	Waikato Environment for Knowledge Analysis

ACKNOWLEDGEMENTS

First of all, I would like to express my deep thanks and gratitude to Allah Almighty for his help and support in completing this research journey.

I would like also to take this opportunity to express my deep thanks and gratitude to my supervisor Prof. Sunil Vadera who was always ready to help and provide me with a continuous support and guidance throughout the entire period of this research.

Very special thanks to all my family for their continuous love and support, most especially,

My father, for his continuous prays for me.

My wife, for her unlimited support and enduring patience throughout these years.

My daughters, who always make me smile and forget the stress of study.

ABSTRACT

Cost-sensitive multiclass classification problems, in which the task of assessing the impact of the costs associated with different misclassification errors, continues to be one of the major challenging areas for data mining and machine learning.

The literature reviews in this area show that most of the cost-sensitive algorithms that have been developed during the last decade were developed to solve binary classification problems where an example from the dataset will be classified into only one of two available classes.

Much of the research on cost-sensitive learning has focused on inducing decision trees, which are one of the most common and widely used classification methods, due to the simplicity of constructing them, their transparency and comprehensibility.

A review of the literature shows that inducing nonlinear multiclass cost-sensitive decision trees is still in its early stages and further research could result in improvements over the current state of the art. Hence, this research aims to address the following question:

> ***How can non-linear regions be identified for multiclass problems and utilized to construct decision trees so as to maximize the accuracy of classification, and minimize misclassification costs?***

This research addresses this problem by developing a new algorithm called the Elliptical Cost-Sensitive Decision Tree algorithm (ECSDT) that induces cost-sensitive non-linear (elliptical) decision trees for multiclass classification problems using evolutionary optimization methods such as particle swarm optimization (PSO) and Genetic Algorithms (GAs). In this research, ellipses are used as non-linear separators, because of their simplicity and flexibility in drawing non-linear boundaries by modifying and adjusting their size, location and rotation towards achieving optimal results.

The new algorithm was developed, tested, and evaluated in three different settings, each with a different objective function. The first considered maximizing the accuracy of classification only; the second focused on minimizing misclassification costs only, while the third considered both accuracy and misclassification cost together. ECSDT was applied to fourteen different binary-class and multiclass data sets and the results have been compared with those obtained by applying some common algorithms from Weka to the same datasets such as J48, NBTree, MetaCost, and the CostSensitiveClassifier.

The primary contribution of this research is the development of a new algorithm that shows the benefits of utilizing elliptical boundaries for cost-sensitive decision tree learning. The new algorithm is capable of handling multiclass problems and an empirical evaluation shows good results. More specifically, when considering accuracy only, ECSDT performs better in terms of maximizing accuracy on 10 out of the 14 datasets, and when considering minimizing misclassification costs only, ECSDT performs better on 10 out of the 14 datasets, while when considering both accuracy and misclassification costs, ECSDT was able to obtain higher accuracy on 10 out of the 14 datasets and minimize misclassification costs on 5 out of the 14 datasets. The ECSDT also was able to produce smaller trees when compared with J48, LADTree and ADTree.

Chapter 1 : INTRODUCTION

1.1 Background

Nowadays, due to the vast amounts of information available and the rapid development in the use of computers and modern information technologies, the optimal use of these data is a challenge for decision-makers who are struggling when they have to make the right decisions to ensure the best results whilst simultaneously achieving low costs associated with those decisions. Machine Learning is one of the most prominent fields of science that has been employed to assist decision-makers, scientists and researchers in various branches of science and knowledge discovery. Machine learning is the field through which decision-making criteria are learned and developed using the available data and utilizing information and knowledge that has been acquired from previous experiences in the field of study. It has become one of the most widely used subfields of computer science, largely because it is used in a wide variety of applications. Some examples of applications that use machine learning include processing natural language, speech and sound recognition, fraud detection, documents checking, computer visibility, and medical diagnosis. With the expansion of the use of the internet and social networking media, the amount of data exchanged between users has increased significantly. Because of these factors, there has been significant progress in the field of machine learning which provides many tools that can intelligently gather and analyse different types of data and then utilise this experience to produce valuable information.

Classification is one of the vital tasks for machine learning and aims to build prediction models from labelled training data, so they can be used for determining the class or label of new unseen data. The last two decades has seen many studies that aim to develop algorithms that learn to perform accurate classification from data such as the Classification and Regression Tree (CART) algorithm (Breiman et al. 1984), the Iterative Dichotomiser 3 (ID3) algorithm (Quinlan 1986), the C4.5 algorithm (Hormann 1962), the Bagging algorithm (Breiman 1996), the Fuzzy Support Vector Machines algorithm for pattern classification (Abe & Inoue 2001), the Stochastic Gradient Boosting algorithm (Friedman 2002) and the Improved Fuzzy Classifier Function (IFCF) algorithm (Celikyilmaz et al. 2009). But the main

objective of most of these techniques was only to minimize the error rate and ignored costs associated with any type of misclassification as they assume that the costs of all types of misclassification are equal.

Cost-sensitive learning is one of the most challenging recent research topics in data mining and machine learning that strives to develop solutions that help decision-makers make the right decisions at the lowest possible costs. For example, in real-world medical diagnostics, classifying a person with a serious illness as a healthy person is more critical and more costly than classifying a healthy person as a sickly one where the error in the first case may cost the life of the patient, while in the second case the cost will be limited to the cost that is associated to some extra medical tests.

During the last decades, many cost-sensitive learning methods and algorithms have been developed in this field. Lomax and Vadera (2011) present a comprehensive survey of existing studies and algorithms that have been introduced for the purpose of cost-sensitive decision tree learning. Their review identified over 50 algorithms for cost-sensitive learning, most of which focus on binary classification problems. Most authors tackle multiclass problems by converting them into many binary-class sub-problems and then applying the normal binary classifiers on them (Aly 2005) but intuitively the disadvantage of such methods is that they can give fairly unreliable results as if only one of the binary classifiers make a mistake, then it is possible that the entire prediction is wrong.

Vadera (2010) notes that the majority of recent cost-sensitive decision tree induction algorithms, such as in WEKA and R, attempt to deal with classification problems by utilizing linear separators, such as using straight lines, or what is known as axis parallel splits to separate out non-linear regions. Clear visualisation is another challenge facing the current cost-sensitive algorithms. It has been stated in (Ankerst et al. 1999), that the effective visualization of large decision trees particularly in the learning process still requires efficient tools that make the visualization more clear and easy to understand.

1.2 Motivation

The rapid development of powerful computer systems on one hand and the availability of a large amount of labelled or unlabeled data, on the other hand, provide an opportunity to build systems that learn from these data to help make the decisions that are not only accurate and reliable but also least costly.

Although the literature shows a major effort aimed to develop effective cost-sensitive algorithms, it also shows that most of the effort is focused on two class problems. So, the main motivation for this research is to discover the power of non-linear classification methods (ellipses) that utilize optimization methods such as particle swarm optimization (PSO) and Genetic Algorithms (GAs) to induce non-linear cost-sensitive decision trees that consider the costs of different misclassification errors for multiclass problems, as well as producing decision trees that are effective and at the same time small in size.

To understand the motivation for using elliptical boundaries, consider Figures 1.1 and 1.2. It is clear that using linear classifiers as shown in Figure 1.3 and Figure 1.4 to solve such non-linear classification problems are not the suitable and appropriate solutions.

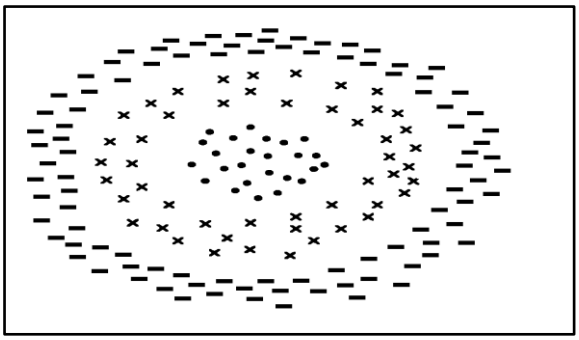

Figure 1.1: Nonlinear classification (example - 1)

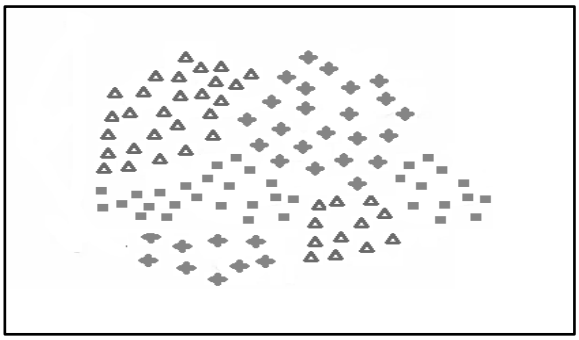

Figure 1.2: Nonlinear classification (example - 2)

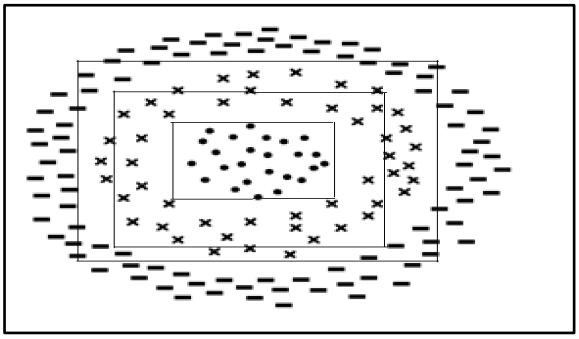

Figure 1.3: Linear classification for (example - 1)

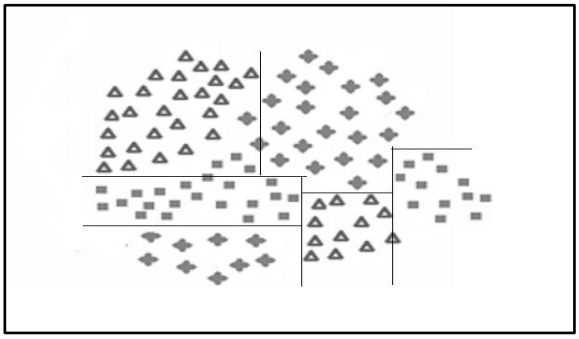

Figure 1.4: Linear classification for (example - 2)

Instead of the above boundaries, Figures 1.5 and 1.6 show the elliptical nonlinear boundaries which are more appropriate visually.

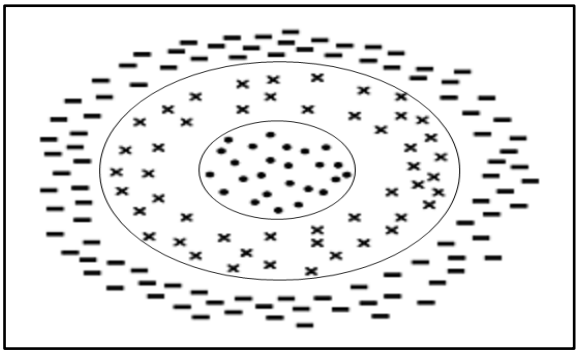

Figure 1.5: Elliptical classification for (example - 1)

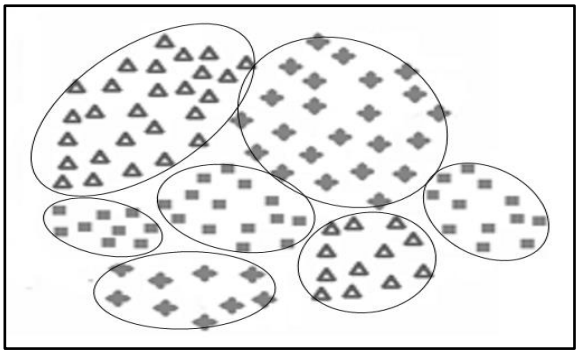

Figure 1.6: Elliptical classification for (example - 2)

1.3 Problem Definition

Given the motivation provided in the previous section, the problem is to develop a cost-sensitive decision tree algorithm with the following properties:

- The algorithm should be suitable for multiclass problems. For example, in an application involving credit assessment, the task could involve constructing a classifier that predicts whether a customer is low, medium, high or very high risk.
- The algorithm should take account of costs of different misclassification cases. For example, when dealing with fraud detection related to granting loans, the costs should not only be based on the true and false predictions (fraud / non-fraud) but also should consider the different amount of cost involved in each misclassification case.
- The algorithm should be able to learn non-linear boundaries that reflect the data.
- The algorithm should reduce the size of decision trees to make them easy to understand and interpret.

1.4 Aims and Objectives

Given the above motivation, the main aim of this research is to develop a novel cost-sensitive classification algorithm that uses elliptical boundaries for classifying multiclass datasets and which improves over the current cost-sensitive classifiers in terms of accuracy, minimizing

misclassification cost and producing smaller decision trees that are easier to interpret and understand.

In order to achieve this aim, the following objectives are developed:

1- To conduct a deep survey of the field of cost-sensitive classification, in order to identify the strengths and weaknesses of existing approaches to cost-sensitive classification and problems faced by researchers addressing similar problems.
2- To develop and implement the proposed new algorithm (ECSDT) that could make a step forward in enhancing the performance of cost-sensitive classifiers.
3- To utilize and explore the performance of different evolutionary optimization methods such as Genetic Algorithms (GAs) and particle swarm optimization (PSO) during the implementation of the new algorithm (ECSDT).
4- To compare the results obtained by the new algorithm (ECSDT) against some common accuracy-based classifiers and some cost-sensitive decision tree methods available in the Weka system (Witten et al. 2016) such as J48 that implements the standard C4.5 algorithm (Hormann 1962), NBTree (Kohavi 1996), MetaCost (Domingos 1999) and CostSensitiveClassifier (Witten et al. 2016).

1.5 Research Methodology

This section describes the rationale for the particular research methodology adopted for this study.

Creswell (2003) defines research methodology as "a strategy or plan of action that links methods to outcomes, and it governs our choice and use of methods (e.g., experimental research, survey research, ethnography, etc.)". Research methodology describes the way that will be followed towards solving the research problem. It shows the various steps that are generally adopted by a researcher in studying his research problem along with the logic behind them (Kothari 2009). Any research should be planned and conducted based on what will best help to answer its research questions. There are three major types of methodologies that can be adopted when conducting research (Barks, 1995; Kraska, 2010):

- Quantitative research, that relies primarily on the collection of quantitative data.
- Qualitative research, that relies primarily on the collection of qualitative data.

- Mixed research, which involves the mixing of quantitative and qualitative methods or other paradigm characteristics.

1.5.1 Quantitative Research Methodology

The literature provides many definitions of a quantitative research methodology and the majority of them share the core principles. The quantitative research methodology can be defined as:

> "Explaining phenomena by collecting numerical data that are analyzed using mathematically based methods (in particular statistics) " (Sibanda 2009).

This definition describes clearly that quantitative methods are used to solve or explain problems by gathering the required numerical data and then applying mathematical and statistical methods to verify claims and hypothesis, such as whether one algorithm performs better than another.

This PhD research, which is in the field of machine learning, adopts an experimental quantitative research methodology because of the need for providing an objective statistical evaluation for the performance of the algorithm and also needs statistical comparisons with other algorithms. Figure 1.7 summarises the main steps of the methodology used in this study. First, a literature survey of the field, which in this research is a cost-sensitive decision tree learning and classification, was carried out. Strengths and weaknesses of existing algorithms were identified and then the research challenges were identified. After that, the main ideas and concepts of the new algorithm were formulated and various alternatives were explored and considered. One of the proposed ideas was then selected and an outline algorithm was developed. A suitable language and toolkit that enable implementation and exploration were then identified. The Java programming language with the Eclipse platform which is widely used was adopted. Given the need for optimization, a framework called MOEA (Multi-Objective Evolutionary Algorithms) was also adopted. Then the algorithm was implemented and provided an experimental environment that enabled the exploration of variants of the main idea.

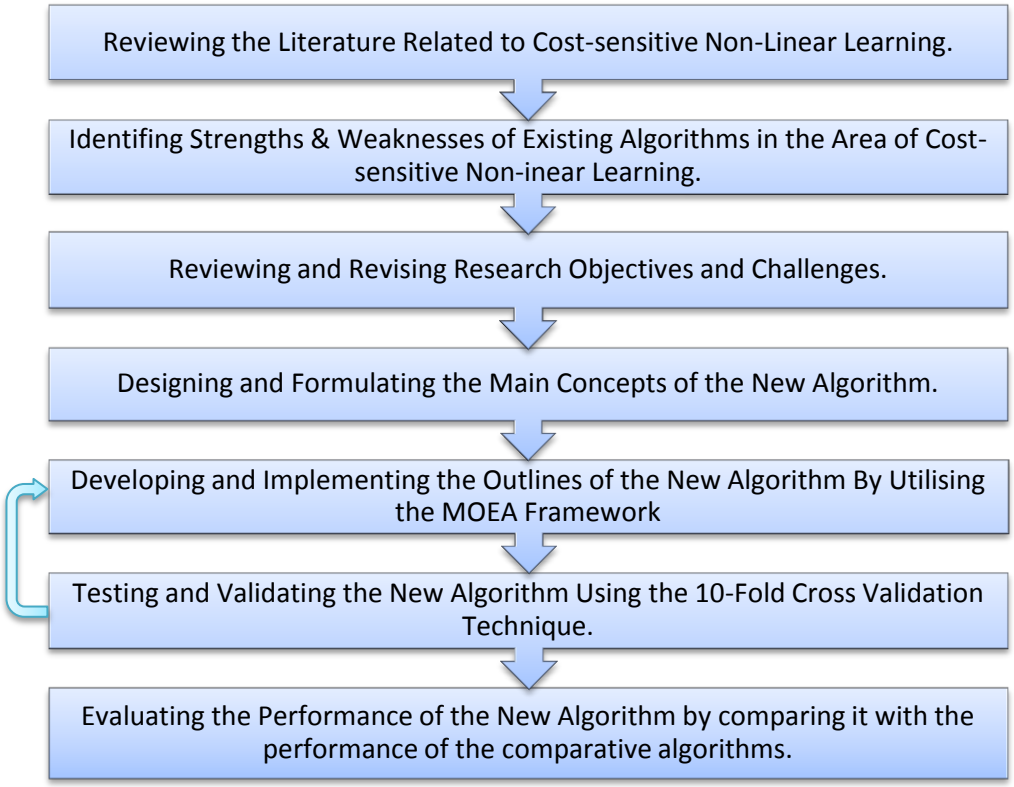

Figure 1.7: Methodology stages

The exploration activity firstly was mainly on handcrafted data where the best results are known and then extended for a number of real datasets from the UCI repository (Lichman 2013). When the exploration was over, the performance of the algorithm was evaluated against some of the well-known algorithms in this area. The empirical evaluation was done using a 10 fold cross validation methodology (Mudry & Tjellström 2011). The measures used are consistent with those widely used in the field, namely accuracy and cost of misclassification (Turney 1995). Since one of the main objectives of developing the new algorithm is to reduce the size of the decision trees, so, the size of the produced decision trees has been adopted as another criterion for the comparison. A wide range of datasets was used from the UCI repository (Lichman 2013) and the results are compared with the current state of the art in the field including algorithms such as those available in WEKA.

1.6 Research Contributions

The main contributions of this research are:

- Identifying weaknesses and strengths of the current algorithms in the field of cost-sensitive classification.
- Developing and introducing a novel non-linear method (ellipses) to separate classes.
- Introducing a direct solution to multiclass classification problems without the need for dividing the multiclass problem into binary sub-problems as is the case with other algorithms when dealing with multiclass problems.
- Exploring the performance of the new algorithm (ECSDT) when two different evolutionary optimization methods (PSO and GA) are applied and then comparing the results of the two methods.

1.7 Outline of Thesis

This PhD research is organized as follows.

- Chapter 1 presents the general introduction to the research. This chapter sets out the motivation, problem definition, aims and objectives, research methodology, main contributions, and ends up with the outline of the thesis.
- Chapter 2 contains two main sub-sections, the first sub-section presents the background for supervised learning and the second section presents the background for decision trees learning as they are both quite relevant to the area of research. The first sub-section presents, the main concepts of supervised learning, and gives some details on cross-validation techniques which are the most widely used methods for validating classification algorithms. The second sub-section provides a background on decision tree learning, gives some details about different feature selection methods that are adopted by different decision tree learning algorithms, and ends up with explanations of some measures that are adopted for evaluating the performance of decision tree algorithms.
- Chapter 3 summarizes a literature review which was conducted on everything related to the research area. The literature review focuses on four areas: Cost-Sensitive Learning, Nonlinear Classification, Multiclass Classification and Evolutionary Optimization Methods. In this chapter, different cost-sensitive theories and strategies

are presented, different attempts and studies for inducing non-linear decision trees are presented, previous methods adopted for multiclass classification problems are illustrated, and ends up with a literature review about using evolutionary optimization methods such as GAs and PSO for inducing decision trees.

- Chapter 4 describes the methodology adopted for developing the new algorithm, explains how ellipses are used to formulate cost-sensitive learning as an optimization problem, shows how ellipses are used to construct the decision tree, gives some explanations about the MOEA framework and how it is utilized for the implementation of the new algorithm and ends up with some illustrative examples of how the new algorithm works.
- Chapter 5 provides some details about the datasets used for the empirical evaluation as well as some details about the comparative algorithms used to evaluate the performance of the new algorithm. This chapter also displays and discusses the results obtained when applying the three different implementations of the new algorithm (Accuracy only, cost only, and accuracy with cost) as well as comparisons of the obtained results with other algorithms.
- Chapter 6 evaluates to what extent the new algorithm achieved its objectives, the primary contributions and conclusions and recommendations for future work.

Chapter 2 : GENERAL BACKGROUNDS

As mentioned in the introduction, this research aims for developing a new algorithm to induce cost-sensitive decision trees. As it is known, Decision tree learning is a supervised learning method, so that, this chapter firstly presents some relevant background on supervised learning which one of the main is fields of machine learning and then gives a general background on decision tree learning. Section 2.1 starts with a general background on the supervised learning field, and then explains the related basic concepts and terminology used in this field, after that lists the general steps for developing any supervised learning algorithm, and finally ends up with some detailed explanation about the Cross-Validation technique which is one of the most important techniques used in the development of the supervised learning algorithms. Section 2.2 gives a general background on decision tree learning and also provides some details on two very important topics related to learning decision trees. The first topic presents some details on the common methods used for feature selection when constructing decision tree algorithms and the second topic covers the common measures used for evaluating the performance of decision tree algorithms. Section 2.3 presents two important concepts that should be considered when building cost-sensitive classifiers, namely: a confusion matrix and a cost matrix.

2.1 Background of Supervised Learning

Machine Learning (ML) can be used in different ways. Learning from existing data is one of the most successful applications of machine learning and data mining. It is difficult for decision makers to find the optimal solutions for problems which are characterized by a very large number of features that signify and differentiate among various data elements. Due to the multiplicity and a large number of features that describe some data sets, there are always chances that decision makers may make errors when performing analyses, or when trying to develop correlations between different features. These issues can be successfully resolved using machine learning and data mining techniques.

During the learning process adopted by machine learning algorithms, the learner interacts with the environment. Hence, the learning process may be explained as a means of using the experience to acquire skills and knowledge. The kind of interactions that are adopted by the machine learning algorithms during the learning process can help in grouping and categorize the learning processes into several categories. The distinction between supervised, semi-supervised and unsupervised learning is observed quite frequently.

- **Supervised Learning:** In this category, the classes in which the data elements were classified accordingly are well-known and clear, and there are well-specified boundaries for each class in a particular (training) data set, and the learning process will be accomplished using these classes (Suthaharan 2016). Every example (element) in supervised learning is a pair that involves an input feature(s) and a required output value. The training data is interpreted and evaluated by a supervised learning algorithm, which tries to determine how the input factors are related to the target factors. After this, an inferred function is created, known as a classifier model or a regression model. It is important that the produced model provides the correct output value for any new, valid, unobserved input element (Kotsiantis 2007). More details about supervised learning are given in section 2.1.

- **Unsupervised Learning:** Unsupervised learning involves deducing and distinguishing different patterns in a particular dataset without knowing the class labels of any of the instances belonging to the dataset. With unsupervised learning, input data is processed by the learner to generate a summary or any targeted form of the data. A common example of this kind of learning is clustering in which a data set is clustered into subsets which have similarities in the targeted characteristics (Shalev-Shwartz & Ben-David 2014). Another example of unsupervised learning is the association rules techniques. Association rule learning is used to find suitable rules that can characterise a large part of the dataset, such as the customers that buy an X extra of goods whenever they aim to buy Y goods (Oellrich et al. 2014).

- **Semi-Supervised Learning:** Semi-supervised learning is a mix between supervised and unsupervised forms of learning where the dataset usually has a large amount of input data and only a few of the data is labelled whereas the majority is not labelled. In this type of learning, the model is first constructed using only the labeled data and then

testing the unlabeled data on the model to set predicted class-labels for them and then a new supervised model is constructed using all the data which consists of the actually labeled data with the unlabeled data that have been given a predicted class-label (Board & Pitt 1989).

2.1.1 Basic Concepts of Supervised Learning

In supervised learning, some data obtained from a particular domain is used by the learning algorithm as an input, and then a model based on the domain's structure is created as an output. That is, a model of the domain is created using a particular set of observations. The observations, on their own, are referred to as "instances" or "examples", while a set of observations is referred to as a "dataset". In every instance, there is a set of values known as the instance's "attributes". For each dataset, the same group of attributes is used to define every instance. It is assumed by the majority of the applications of machine learning algorithms that the attributes may be of type "nominal" or of type "numeric". Nominal attributes involve a set of unordered values, such as a set of colours {red, blue, green, etc.}, or a set of weather conditions {clear, rainy, windy, etc.}. Numeric attributes, in contrast, are either integers or real numbers. The space of every possible mix of attribute values is referred to as "instance space".

Decision trees are part of machine learning models that are known as "classifiers". Classifiers aim to solve classification problems in which each instance of the dataset is labelled with a nominal attribute value known as the "actual class". The instance space is separated by the classifier into several separated sub-spaces, and then one class is allocated to every sub-space. That is, it is presumed that every instance in the instance space is tagged with one nominal attribute value, known as the "predicted class" of the instance.

A learning algorithm aims to automatically induce a classifier from a set of "training" instances, which have been obtained in practice from the search space and allocated class labels by some other procedure, such as by human professionals. A classifier is created by the learning algorithm by dividing the instance space in accordance with the class labels of the

training instances. In a perfect situation, the classifier will help in classifying all the examples correctly.

When an instance is allocated the wrong class, then we refer to it as a "misclassified" instance or example. The predictive performance of a particular classifier is determined by its "accuracy" or by its "error rate", and these should be determined using an independent set of labelled instances which are not used with the learning algorithm when it creates the classifier. This set of instances is referred to as "testing data". The noise that is caused by allocating "incorrect" class labels to training instances may misguide the learning algorithm. This issue arises as the learning algorithm builds a classifier in accordance with the class labels from the training data. Therefore, instances which are not labelled correctly need to be determined and it should be made certain that they have no impact on the construction of the classifier. When there is a close fit between the classifier and the training instances, then noisy instances may also fit with the classifier, making it a less valuable classifier. This problem is referred to as "overfitting" and a usual technique in decision trees is to remove those parts of a classifier that may overfit the training data. This process, known as "pruning" and it helps in increasing the accuracy as well as the comprehensibility of the induced classifier and also decreases the size of the classifier.

2.1.2 Developing Supervised Learning Algorithms

According to (Suthaharan 2016), four key stages should be followed to create a new supervised learning algorithm. These are training, testing, validation and evaluation, and are explained as follows:

- **Training Phase:** Using a labelled dataset known as the training dataset, a systematic approach should be offered in this phase. This approach creates a parameter's model to choose the ideal parameters among others for building the classifier in each particular stage. In the training phase, good quantitative metrics such as the false positive and true positive ratios should be used so that the best parameters that decrease the error between the predicted class labels and the actual class labels can be chosen in the training dataset. The training phase normally includes the sub-processes which are explained in the following steps (Suthaharan 2016):

1. Data domain extraction: This step involves extracting the data domain that has been formed by the feature space from the given data set. This is done to compute the required statistical measures and the variance among the feature space variables.
2. Response set extraction: The training data are provided with class labels, and hence, the labels of every observation can be extracted to develop some response sets that allow each data domains to be assigned to a particular response set. This set helps in computing the accuracy of the prediction and the prediction error by using the predicted responses extracted by the model and by its parameters.
3. Modelling: It refers to creating a relationship between the different variables.
4. Using class labels: The model training is a supervised learning approach; therefore, it is assumed that each instance of the training data has properly labelled with the proper class. If the labelling of the classes is not done properly, the training data considered as an invalid data because the accuracy of the prediction will be calculated as a ratio between the predicted class labels and the actual class labels for the dataset.
5. Optimization: This step includes updating the values of the parameters so that most of the computed responses are closer and corresponding to the actual class labels. This can be done using some quantitative measures, like distance measures and probabilistic measures (entropy and information gain).

➢ **Testing Phase:** Model testing is a process that includes analyzing the efficiency of the model trained by the algorithm. That is, it will be confirmed through the testing phase is that the trained model also works on different labelled dataset called (testing dataset) that is not used during the construction of the model. Majority of the steps of the training phase are also part of the testing phase, with the only distinction being that in the training phase, is that the steps are repeated so that the optimal parameters can be obtained; however, in the testing phase, these steps are performed only once.

➢ **Validation Phase:** In the validation phase, a systematic approach also should be presented so that the model can be trained and tested in various conditions so that it can perform in an efficient manner on unobserved data. This is attained by first developing different mix of training and testing sets from a provided labeled dataset, and then the above training phase is applied on the training sets, after that, testing the parameters obtained in the training phase with the testing sets using a quantitative measure which has a significant role in this validation process. The results are subsequently combined and then an average is obtained. One of the

most common techniques used for the purpose of validation is the cross-validation method (Arlot & Celisse 2010). Through this validation, it is made certain that the overall dataset is completely trained, validated and tested so that the classification error can be minimized, and the accuracy can be maximized for the unseen data.

- **Evaluation Phase:** In the evaluation phase, it is determined whether the trained and cross-validated model is useful when a different data set is used, that had not been used earlier in the training or validation phases. The labelled data set is only used in this phase to test the results given by the final model with respect to some evaluation metrics, like classification accuracy and execution time. For this purpose, there are various measures known as qualitative measures as they can determine the quality of the performance of the trained model. The measures that are often used include accuracy, specificity, sensitivity and precision, as described in (Costa et al. 2007).

2.1.3 Cross-Validation

The purpose of cross-validation was to solve the problem of overoptimistic outcomes given by training an algorithm and evaluating its statistical performance on the same data set (Suthaharan 2016). It was asserted earlier that the cross-validation technique is the most widely used method for validating classification algorithms (Arlot & Celisse 2010). It is shown in machine learning literature that there are different methods used for cross-validation. The most widely used cross-validation methods are presented and discussed in the following sub-sections.

- **K-Fold Cross-Validation:** In this method, the complete data set is initially shuffled and then divided into K disjointed subsets with the same sizes. The first step involves training the model using the first *(K-1)* folds, followed by testing it using the K^{th} fold data (which have not been used in the training). In the second step, the $(K-1)^{th}$ fold is chosen for testing, and the training involves the other folds. This process is then recursively repeated over other combinations of training and testing sets. This will provide K classification models that produce K accuracies (or any other qualitative measures), and this will be averaged as the final testing result (Anguita et al. 2012). When accuracy is considered as the fitness function for the classifier, then the

following equation can be used to obtain the average accuracy for the K testing folds:

$$\text{Average Accuracy} = \frac{1}{K}\sum_{i=1}^{K} \text{Accuracy of Testing Fold[i]} \quad 2.1$$

Here, K refers to the number of testing folds. By applying the K-Fold Cross-Validation it can be made sure that all observations are used for training and validation, and every observation is used just once for validation (Anguita et al. 2012)

- **Leave-One-Out:** This method is a special case of K-fold cross-validation, where K refers to the number of instances in the dataset. In every iteration, the entire data except a single observation is used for training the model, and testing the model takes place on that single eliminated observation (Arlot & Celisse 2010). It is found that the accuracy measure attained using this approach is almost unbiased; however, there is a high variance, which leads to unreliable results (Arlot & Celisse 2010). Despite those shortcomings, this method is still used on a wide scale when the data is quite rare, particularly in bioinformatics that has limited quantities of data samples.

- **Leave-P-Out:** In this approach, the testing set contains a number equals to P instances from a dataset contains N instances, while the remaining (N-P) are used as training, and this procedure is repeated for all possible combinations from the N instances (Celisse & Robin 2008). This cross-validation method is quite expensive to use due to the long computation time it consumes (Arlot & Celisse 2010).

- **Random Subsampling:** In this method, K data splits of the dataset are carried out. In each split, a fixed number of examples are chosen without replacement as a testing set and the remaining examples considered as the training set. In each data split, retraining of the model takes place from the scratch with the training examples, and estimation is computed using the testing examples. Then the average of the different estimates is the final estimate obtained (Steyerberg et al. 2001; Kohavi 1995).

- **Dividing datasets:** In this approach, the dataset is divided into two or three parts using specific ratios. Techniques that divide the dataset into two parts; the first part is used for training, while the second part is used for testing. On the other hand, in those methods that divide the dataset into three parts, a third part is incorporated for validation purposes. The ratio used for dividing the dataset plays a significant role which has a

major and direct impact on the accuracy of the classification, as well as the computational complexity. It was mentioned by (Suthaharan 2016) that it is still quite difficult to divide datasets for training, validation and testing effectively, and no ideal method exists that can provide an optimal ratio for splitting the dataset.

2.2 Background of Decision Tree Learning

The purpose of data mining is to obtain valuable information from large datasets and to present it through illustrations that are easy to understand. One of the most successful methods of data mining is the decision tree that was initially presented in the 1960s. There is widespread use of decision trees in various fields (Song & Lu 2015). The main motivation to use a decision tree as the most widely used classifiers in the machine learning applications is that it can be used and understood easily.

According to the description given in (Song & Lu 2015; Olivas 2007; Rokach & Maimon 2005), a decision tree is a classifier that is presented as a recursive separation of the instance space. There are three kinds of nodes in the decision tree. The first node is the "root node", which is an independent node that is not generated by any other node. All other nodes of the decision tree (except the root node), they should be branched off from another node. A node which has outgoing branches is referred to as a test or decision node. The other nodes which are not split any more are all known as leaves or end nodes.

In (Mitchell 1999; Song & Lu 2015; Olivas 2007; Tan et al. 2006)**,** the way DT algorithms like ID3 (Quinlan 1986), C4.5 (Hormann 1962) and CART (Steinberg 2009) construct classifier models is explained, where every test node, including the root node, divides the instance space into two or greater sub-spaces consistent with a particular discrete function. Every leaf node is allocated to one class, which signifies the most widely occurring value. The new unseen instances are then labelled by passing them downwards from the root of the tree to a leaf, in accordance to the outcome of the tests carried out over the path, and considering the leaf's class label as the prediction class for the new unseen instance.

In the simplest and most widely occurring cases, the instance space is divided in accordance with a particular feature's value. In the practical world, the domain of the instance space is shown in a better manner by including several candidate features. However, most of these are

partially or entirely unnecessary or repetitive with respect to the target concept. It has been found in (Tang et al. 2014; Tan et al. 2006) that the efficiency, as well as the accuracy of the constructed decision trees, is influenced by the usage of some techniques that select only the most important features among others when developing DTs. This is because the aim of these methods is to select a small subset of the most influential features and removing the unnecessary ones in accordance with a particular evaluation criterion, leading to an improved learning performance (e.g., higher learning accuracy for classification, lower computational cost, lower running time for learning the algorithms, and better model interpretability).

Four main steps were presented by (Dash, Manoranjan and Liu 1997; Kumar 2014) to be followed in a standard feature selection method. These include:

- Choosing the feature subset by making use of one the appropriate techniques, like the complete approach, the random approach or the heuristic approach (Muni et al. 2006);
- Analyzing and evaluating the subset being chosen to determine and assess the discriminating capability of the selected feature or subset of features so that the various class labels can be identified and distinguished;
- Defining the stopping criterion so that it can be determined when the feature selection process should be stopped from operating continuously in the space of subsets. Some of the common stopping criteria are given in (Rokach & Maimon 2005), and are listed as the following conditions:
    1. When all instances in the training set are related to a single class.
    2. When the predefined highest tree depth has been attained.
    3. If the node were split, then the number of instances in one or more child nodes would be lower than the minimum number of instances permitted for child nodes.
    4. When the value of applying the splitting criteria is not better than the specific predefined threshold.
- Validating the outcomes so that the validity of the chosen subset can be tested by performing various tests, and then contrasting the findings with the results obtained previously.

The following sub-section gives some highlights for the most well-known methods used for the feature selection.

2.2.1 Feature Selection Methods for Decision Trees

As demonstrated earlier, a decision tree is developed top-down from a root node and consists of dividing the data into appropriate subsets. Therefore, a significant query that arises is that "which measure is the most appropriate for choosing the feature or sub-features that are most useful for dividing or categorizing a specified dataset". Several algorithms have been described in the past studies that make use of various techniques for choosing the best features for dividing the dataset in every decision node when building decision trees. Brief descriptions are presented in the following paragraphs which highlight some feature selection techniques that have been used in the most common decision tree algorithms, i.e. CART, ID3 and C4.5.

- **GINI Index - CART**

The CART algorithm was developed by Breiman in 1984. CART is an abbreviation that stands for Classification And Regression Trees.

Decision trees constructed using the CART algorithm are usually binary decision trees, that is, every split relies on the value of a single predictor variable, and there will be only two child nodes of every parent node. The splitting selection criterion that is used by CART is based on what is known as the GINI index. For any training set S, the following equation is used to compute the GINI index of S (Gupta & Ghose 2015):

$$Gini\ (S) = 1 - \sum_{i=1}^{k} p_i^2 \qquad 2.2$$

Where (k) refers to the number of class labels, P_i refers to the probability that an instance in S is part of the class (i).

The GINI index for choosing the attributes A in the training dataset (S) is subsequently described as follows:

$$\begin{aligned} \text{Gini}\ (A, S) &= \text{Gini}\ (S) - \text{Gini}\ (A, S) \\ &= \text{Gini}\ (S) - \sum_{i=1}^{n} \frac{|S_i|}{|S|} Gini\ (S_i) \end{aligned} \qquad 2.3$$

Here, S_i refer to the partition of S that has been induced by the value of attribute A, and the attribute having the largest GINI value is taken to be the ideal splitting attribute.

- **Entropy and Information Gain - ID3**

ID3 is an abbreviation of (**I**ternative **D**ichotomizer **3**), which is an entropy-based algorithm, where the split criterion is developed such that it seeks an improvement in the purity of leaves (Quinlan 1986). The basis of this criterion is the information theory, and the feature having the largest information gain is given preference. Information gain seeks to measure how well a specific feature sets apart the training examples on the basis of their target classification (Ballester & Mitchell 2010). The entropy that the ID3 algorithm adopts initially was based on the entropy concept discovered by Claude E. Shannon in (Shannon 1948) that computes the sample's homogeneity. When the sample is fully homogeneous, then the entropy is zero, and when the sample is equally divided, then there is entropy of one. Hence, the ideal choice for splitting is the feature that has the higher decrease in entropy (Peng et al. 2009).
Shannon's entropy (Shannon 1948) for an attribute A is described as follows:

$$E(A) = -\sum_{i=1}^{n}(p_i \log_2 p_i) \qquad 2.4$$

Here, (n) refers to the number of classes, and (p_i) is the probability of examples that are part of the class (i) in the set.

Information gain is an add-on to the entropy concept and explains the significance of a specified attribute A of the feature vectors. It is also used to determine the way the more informative attributes are ordered in the nodes of a decision tree. The information gain for an attribute A in a given dataset S determines the variation in entropy from prior to and following the splitting of the training set S on the attribute A. The following equation can be used to calculate this information gain:

$$IG(S, A) = E(S) - E(A) \qquad 2.5$$

Here, E(S) refers to the entropy of the present training set prior to the splitting. The concept of developing a decision tree is based on determining the attribute that offers the largest information gain.

- **Gain Ratio - C4.5**

The C4.5 algorithm is an extension to the ID3 algorithm, both of which have been designed by Quinlan.
The novel features for C4.5 in contrast to ID3 are:

- Continuous and discrete features are both approved.
- Incomplete data points are managed.
- The issue of over-fitting is resolved by adopting the bottom-up method, which is referred to as "pruning".

One of the two given criteria can be adopted by C4.5. The First criterion is the information gain explained in the preceding section. The second criterion is an entropy-based criterion that is also based on the information gain. Information gain was found to show a preference for attributes that had a high degree of cardinality (having a large number of potential values). The ideal variable for splitting is one that is unique for every record. C4.5 computes the possible information from each partition (highest possible information) and contrasts it with the actual information. This is referred to as the Gain Ratio. Normalization to information gain is applied by the Gain Ratio, where a value defined as follows is used:

$$\text{SplitInfo}_{\text{A}}(\text{S}) = -\sum_{i=1}^{v}(|S_i| \ / \ |S|)\ \log_2(|S_i| \ / \ |S|) \qquad 2.6$$

This value is indicative of the information produced by splitting the training dataset S into v partitions according to v results of a test on the attribute A. The gain ratio is subsequently explained as follows:

$$\text{Gain Ratio (A)} \ = \ \text{Information Gain (A)} \ / \ \text{SplitInfo}_{\text{A}}(\text{S}) \qquad 2.7$$

Here, IG (A) refers to the information gain for choosing attribute A and is computed as depicted in section (2.1.2.2). Once the Gain Ratio for every attribute is determined, the attribute having the largest gain ratio is chosen as the splitting attribute.

2.2.2 DTs Performance Measures

Once a new machine learning algorithm has been developed, it is important to find out how effective is the model based on metrics and datasets. Performance measures (or evaluation measures) play important roles in machine learning. These measures not only serve as the criteria to assess learning algorithms but also play the role of heuristics to develop learning models. The most widely used performance criterion for DTs are statistical measures like accuracy and error rate; however, the most appropriate DT for a certain classification problem can be selected by considering other significant measures that have a significant impact on the performance of DT. These include DT comprehensibility and stability of predictions (Osei-

Bryson 2004). The most widely used performance criterion for assessing machine learning algorithms are presented in the following sub-sections.

2.2.2.1 Statistical Measurements

The literature of machine learning shows that various statistical performance tools were used to statistically analyze the efficiency of machine learning algorithms. The most widely used methods for this purpose are an Error rate, Accuracy Rate, Precision, Recall and F1-Score etc. A brief explanation of these evaluation metrics and how they are calculated are given in the following short paragraphs (Fawcett 2006; Powers 2011; Alvarez 2002; Davis & Goadrich 2006).

Accuracy Rate: It refers to the ratio that is attained by dividing the number of accurate estimations with the overall number of estimations.

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \qquad 2.8$$

Error Rate: It is the ratio of all estimations that are not correct. It basically measures how inaccurate a model is.

$$\text{Error} = \frac{\text{FP+FN}}{\text{FP+FN+TP+TN}} \qquad 2.9$$

Precision Rate: It refers to the number of positive predictions divided by the overall number of positive class values estimated. That is, it is the ratio of True Positives to the overall number of True Positives as well as the number of False Positives. This is also referred to as the Positive Predictive Value (PPV).

$$\text{Precision (PPV)} = \frac{\text{TP}}{\text{TP+FP}} \qquad 2.10$$

Recall Rate: It is the ratio of True Positives to the number of both True Positives and False Negatives. That is, it is the ratio of the positive estimations to the number of positive class values in the testing data. This ratio is also referred to as the Sensitivity or the True Positive Rate.

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \qquad 2.11$$

F1-Score Rate: This ratio is also referred to as the F-Measure and demonstrates the balance between the Precision and the Recall.

$$\text{F1} - \text{Score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad 2.12$$

2.2.2.2 DT Comprehensibility

Decision trees are usually considered to be the most precise and efficient classification methods, however, at times, DTs, particularly the bigger ones, are difficult to comprehend and explain. Therefore, they become incomprehensible to experts (J.R. Quinlan 1987), so it is vital to make sure that DTs are as easy to understand so that they can be interpreted even by non-professionals. The complexity of DT is normally measured using one of the metrics given below (Olivas 2007):

- The numbers of nodes that create and construct the tree.
- The number of leaves that are created by the tree
- The tree depth, i.e. the length of the largest path, from a root to the leaf, and a total number of attributes considered.

Various techniques have been presented in the last few years to simplify trees. These include pruning methods that are possibly the methods used most extensively to decrease the size of DTs. In machine learning, pruning is the process of removing non-predictive subtrees (branches) of a model so that its accuracy can be increased, size can be decreased, and the issue of overfitting can be avoided (Patel & Upadhyay 2012). Pruning is of two kinds, pre-pruning and post-pruning (Fürnkranz 1997). Pre-pruning is also known as forward pruning, which prevents the growth of trees early on, at the beginning of the process of constructing the DTs, so that the generation of unimportant branches can be avoided. Post-pruning is also known as backward pruning. In this method, the tree is first constructed and then, the unimportant branches of the decision trees are reduced.

Another technique that is used to simplify decision trees includes decreasing the size of the original learning set (referred to as data reduction techniques). For this, the unimportant features are eliminated before the tree induction process (Sebban et al. 2000).

2.2.2.3 Stability of the Results

Another important factor to consider when creating and assessing the efficiency of DTs is the stability of the DT prediction findings. The criterion of stability performance is important with respect to the quality of DT, as there should not be much difference in the predictive accuracy rate when a DT is used on various validation data sets (Osei-Bryson 2004). One of the most widely used methods by the developers is the k-fold cross-validation estimator (particularly when there is a limited amount of data) which helps to make sure that the outcomes attained by the various DTs are constant and also help in providing an optimal decrease in variance (Kale et al. 2011). More details on K-fold cross-validation is given before in section 2.1.3.

2.3 Confusion and Cost Matrices

To take account of misclassification costs, there are two important concepts that should be considered when building the classifier, namely: a confusion matrix and a cost matrix, which are introduced below.

A confusion matrix, also known as an error matrix (Stehman 1997) is a table representing information about the actual and predicted classifications done by a classification system (Santra & Christy 2012). Table 2.1 shows the format of an ordinary confusion matrix for a binary classification problem which records the predicted outcomes against the actual outcomes. Table 2.2 shows the format of a cost matrix for a binary classification problem. The misclassification cost associated with classifying an instance from its real class (j) into a predicted class (i) is represented in a cost matrix using the notation C (i,j). The misclassification costs are usually determined and proposed by experts and specialists in the field under study, or they may be obtained and learned through other methods. It is normally presumed in cost-sensitive learning that this kind of cost matrix is specified and known. The binary cost matrix can be extended without any problems when there are multiple classes by including more rows and columns, based on the number of classes.

Confusion Matrix		Predicted Class	
		Pos (+)	Neg (-)
Actual Class	Pos (+)	TP	FN
	Neg (-)	FP	TN

Table 2.1: The confusing matrix for binary classification problem

Cost Matrix		Predicted Class	
		Pos (+)	Neg (-)
Actual Class	Pos (+)	C (+,+)	C (+,-)
	Neg (-)	C (-,+)	C (-,-)

Table 2.2: The cost matrix for binary classification problem

The ordinary binary confusion and cost matrixes described above can be extended to multiclass problems. For example Tables 2.3 and Table 2.4 respectively give the forms of a confusion matrix and a cost matrix for a problem labelled with three classes (A, B and C) wherein the confusion matrix: AA denotes the number of examples that are actually of class A and predicted as class A. AB denotes the number of examples that are actually of class A and predicted as class B and so on for the rest. And for the cost matrix C(AA) denotes the cost of (AA), C (AB) denotes the cost of misclassifying A as B and so on for the rest.

Confusion matrix		Predicted Class		
		A	B	C
Actual Class	A	AA	AB	AC
	B	BA	BB	BC
	C	CA	CB	CC

Table 2.3: The confusion matrix for a 3-classes classification problem

Cost matrix		Predicted Class		
		A	B	C
Actual Class	A	C(AA)	C(AB)	C(AC)
	B	C(BA)	C(BB)	C(BC)
	C	C(CA)	C(CB)	C(CC)

Table 2.4: The cost matrix for a 3-classes classification problem

Elkan (Elkan 2001) and Ling & Sheng (2008) described how important it is to consider the misclassification costs in various cost-sensitive learning algorithms and mentioned that, when the cost matrix is defined, a new instance x should be classified into the class (i) that leads to the minimum expected cost L(x, i) which is expressed as:

$$L(x,i) = \sum_{j=1}^{n} p(j \mid x) * c(i,j) \qquad 2.13$$

Where p (j | x) is the probability of classifying an instance x into class j, and c (i, j) is the cost of predicting class (i) when the true class is j.

The problem of cost-sensitive learning can be formulated as wanting to learn a classifier that minimises the expected total of misclassification cost, then from the previous two matrixes a set of equations can be derived for multiclass problems as follows:

- ***Accuracy Rate*** = $\frac{\sum_{i=1}^{m} TC_i}{N}$ 2.14

  Where m is the number of class-labels the dataset has, TC_i is the number of examples of Class C that have been classified correctly and N is the total number of examples.

  Then the accuracy rate for the confusion matrix depicted in Table 1.1 can be as the following form:

  Accuracy Rate = (AA +BB+ CC) / N where, N is the total number of examples.

- ***Error Rate*** = $\frac{\sum_{i=1}^{m} FC_i}{N}$ 2.15

  Where m is the number of class-labels the dataset has, FC_i is the number of examples of Class C that have been classified incorrectly and N is the total number of examples.

  Then the error rate for the confusion matrix depicted in Table (1.1) can be as the following form:

  Error Rate = (AB+ AC+ BA+ BC+ CA+ CB) / N where, N is the total number of examples.

  The error rate can also be calculated as a complement to the accuracy rate, then

  Error Rate = 1 - Accuracy Rate 2.16

- Utilising the confusion matrix in Table 1.1 and the cost matrix in Table 1.2, we can calculate the total misclassification cost for the classifier using the Eq 1.1 as follows:

  Total Cost = AA*C(AA) + AB*C(AB) + AC*C(AC) + BA*C(BA) + BB*C(BB) + ……

To illustrate the ideas in an applied manner, suppose that we have a multiclass problem with the following confusion and cost matrixes shown in Tables 2.5 and 2.6 respectively.

Confusion matrix		Predicted Class		
		A	B	C
Actual Class	A	50	0	0
	B	0	45	5
	C	0	5	45

Table 2.5: Confusion matrix (Example-1)

Cost matrix		Predicted Class		
		A	B	C
Actual Class	A	0	5	5
	B	5	-5	50
	C	50	100	0

Table 2.6: Cost matrix (Example-1)

Then,

Accuracy Rate = (50+45+45) / 150 = 0.9333 = 93.33 %

Error Rate = (5+5) / 150 = 0.0666 = 6.66 %

Also the Error Rate can be calculated a complement to the Accuracy Rate as following:

Error Rate = (1 - 0.9333 = 0.0666 = 6.66 %)

The Total Cost = (0* 50) + (5*0) + (5*0) + (5*0) + (-5*45) + (50*5) + (50*0) + (100*5) + (0*45) = 525 units of cost.

As we notice from the cost matrix above, the cost of BB is (-5), and this is allowed as (Turney 1995) mentioned that assigning negative costs for classification can be interpreted as benefits.

As a second example to illustrate the cost vs accuracy issue, using the same cost matrix shown in Table 2.6, consider a situation where another different algorithm is applied and results in the following confusion matrix:

Confusion matrix		Predicted Class		
		A	B	C
Actual Class	A	50	0	0
	B	10	40	0
	C	10	0	40

Table 2.7: Confusion matrix (Example-2)

Then,

Accuracy Rate = (50+40+40) / 150 = 0.8666 = 86.66 %

Error Rate = (1 - 0.8666 = 0.1334 = 13.34 %)

The Total Cost = (0* 50) + (5*0) + (5*0) + (5*10) + (-5*40) + (50*0) + (50*10) + (100*0) + (0*40) = 350 units of cost.

From the results of Example-2 we can observe that cost reduction can sometimes be at the expense of decreasing the accuracy rate, as in cost-sensitive classification problems, the accuracy rate can be sacrificed in order to obtain the lowest possible cost.

Chapter 3 : THE LITERATURE REVIEW

Considering that the main objective of this research is the development of a new nonlinear cost-sensitive decision tree for multiclass problems by utilizing some multi-objective optimization methods, this chapter presents a review of the literature that covers four related areas Cost-Sensitive Decision Trees, Nonlinear Classification, Multiclass Classification and Multi-Objective Optimization. Section 3.1 starts by reviewing the general literature on cost-sensitive learning and then gives some explanations on cost-sensitive strategies, after that, the categories of cost-sensitive algorithms according to the method adopted for implying the cost were presented, and finally, a more detailed literature on cost-sensitive decision trees which is at the core of this research is presented. Section 3.2 covers nonlinear decision tree. Section 3.3 presents some literature on multiclass classification, and Section 3.4 presents the field of multi-objective optimization using PSO and GAs.

3.1 Cost-Sensitive Learning

The literature on classification in data mining and machine learning shows that many approaches have been developed for data classification, including decision trees, Bayesian networks, neural networks, discriminant analysis, and support vector machines, among many others (Freitas et al. 2009).

Prior to the emergence of cost-sensitive classification algorithms, the classification systems in the past only aim to enhance the classification and prediction accuracy for the classification system (Ling & Sheng 2008). Hence, in the previous decades, the cost-sensitive classification was considered to be an important topic to attract the attention of the researchers in the field of data mining and machine learning which led to the development of many methods and algorithms that take into account the costs when performing the classification (Lomax & Vadera 2011; Turney 1995).

The supervised and unsupervised learning methods described in Chapter 2 were first introduced with a view to maximising accuracy (Lomax & Vadera 2011). In contrast, cost-

sensitive learning techniques focus on those applications in which misclassification errors consist of various costs, and where some of the misclassification errors are more costly compared to others. In a medical diagnosis, for instance, an ill patient who is categorized as healthy is considered as a more expensive error (which can cause a loss of life) compared to categorizing a healthy patient as a sick one (which may merely require additional tests). Another example is when a company issues a credit card to a customer where the loss is normally costlier than an error of refusing a credit card to a good client (Jin Li et al. 2005).

A taxonomy of various costs in learning classification problems has been presented by (Turney 2002). These include costs of testing, costs for misclassification errors, costs of interventions, costs of computation, costs of unwanted achievements, costs of instability, and costs of human-computer interaction.

3.1.1 Cost-Sensitive Learning Theories and Strategies

With cost-sensitive learning, the new example that is to be classified should be labelled with the class that leads to the lowest expected cost and the literature shows many attempts introduced by several authors to formalize the cost-sensitive learning problem (Elkan 2001; Turney 1995). These formulations are based on computing the cost of classification using a confusion matrix and a cost matrix.

The differences in cost-sensitive learning algorithms are due to the way costs are included in the learning process. According to the literature, these algorithms can be grouped into two main groups direct and indirect methods (Nashnush & Vadera 2017). The goal of the first category is to construct classifiers that are cost-sensitive on their own, and this is referred to as the direct methods. Examples of direct cost-sensitive learning algorithms include the ICET (Turney 1995) and cost-sensitive decision tree (Ling et al. 2004; Drummond & Holte 2000). In the other group, a "wrapper" is constructed which transmits any existing cost-insensitive classifier into cost-sensitive one. The wrapper methods are also known as meta-learning cost-sensitive approaches, and examples of these methods are MetaCost (Domingos 1999), Costing (Zadrozny et al. 2003) and Weighting (Ting 1998).

Another attempt for categorizing cost-sensitive learning algorithms is that presented at the comprehensive survey carried out by Lomax and Vadera (Lomax & Vadera 2011) which listed several ways to categorize the algorithms intended for inducing cost-sensitive decision trees. They listed some of the algorithms that are distinct in accordance with the kind of cost(s) included. The purpose of some of these algorithms was to decrease just the misclassification costs; others aim to minimize only the cost of obtaining the information, whereas some others aim to decrease both previous expenses. Then, they categorized the cost-sensitive algorithms depending on the approach adopted. It was asserted by them that there are two key techniques that were used in algorithms to develop the cost-sensitive decision tree. The greedy method is used in the foremost category, where a single tree is developed at a time. The most famous algorithms that use the greedy approach are ID3 (Quinlan 1986) and CART (Breiman et a., 1984). The second category adopts the non-greedy approach that intended for inducing multiple trees.

In the following sub-sections, these categories will be discussed in some details. In section 3.1.1.1, Direct Cost-Sensitive Learning approaches will be described, section 3.1.1.2 gives some highlights related to the meta-learning cost-sensitive approaches. Since this research is about developing a new cost-sensitive decision tree algorithm, a separate sub-section 3.1.1.3 has been allocated to give some detailed explanation on some common Cost-Sensitive Decision Tree algorithms.

3.1.1.1 Direct Cost-Sensitive Learning

When constructing direct cost-sensitive learning algorithms, the key concept is to directly include and use misclassification costs or other kinds of costs while developing the learning algorithms. Various algorithms were depicted in the previous studies that adopt direct cost-sensitive learning techniques; one of the adopted methods is by changing cost-insensitive algorithms like (decision trees, Naïve Bayes, Neural Networks and Support Vector Machine) into cost-sensitive ones by modifying and changing the algorithm itself so that it includes functions that include costs while building the classifier .

- **Cost-sensitive naïve Bayes (CSNB):** CSNB (Xiaoyong Chai et al. 2004) is a test cost-sensitive classifier that alters the Naive Bayes Classifier by incorporating a test approach through which it is determined how unknown attributes are chosen to carry

out the test so as to decrease the total of misclassification cost and test cost. The same equation 2.13 given earlier is used in CSNB to classify test examples on the basis of the pre-produced probability created by the naïve Bayes.

- **Modifying Neural Network Algorithm:** In data mining and machine learning, artificial neural networks are quite well-known and valuable tools that play an important part in classifications. A comparative analysis of various strategies for cost-sensitive learning with neural networks was performed by Kukar and Kononenko (Kukar & Kononenko 1998). It was found in their study that neural networks that were trained utilizing the back-propagation version of the neural networks decreases the misclassification cost significantly and it is performing much better compared to the other techniques. The original back-propagation algorithm usually decreases the squared error of the neural network. Hence, the back-propagation learning process in its originality not appropriate for cost-sensitive learning. So, to reduce costs of the errors, the error function should be altered by considering the misclassification costs and that is by including the cost factor C(i, j) where j is the predicted class, and i is the actual class. Therefore, rather than decreasing the squared error, the altered back-propagation learning process decreases the misclassification cost. One of the issues arises when using the back-propagation learning process is the over-fitting of training data. This issue arises because of the resulted over-sized network which leads to a decrease in the generalization capabilities of the network (Kukar & Kononenko 1998).

- **Modifying Support Vector Machine Algorithm:** The basis of the Support Vector Machine (SVM) is the structural risk minimization (SRM) induction principle that was developed from the statistical learning theory. It has been found that SVMs are successful in various practical applications. A cost-sensitive SVM classifier was put forward by Funeral and Roli (Fumera & Roli 2002) by developing a unique form of the training activity as a non-convex optimization issue, and producing a particular learning algorithm to resolve it. This cost-sensitive learning technique offers a higher degree of flexibility to explain the decision boundaries, keeping in view the rejection method employed for typical SVMs. According to the experimental findings found by Funeral and Roli (Fumera & Roli 2002), for the majority of the data sets, the cost-sensitive SVM that includes the reject option attained a better error-reject trade-off compared to the typical cost-insensitive SVMs. In another study (Masnadi-Shirazi &

Vasconcelos 2010), a novel method for learning cost-sensitive SVM classifiers was presented, in this study, the standard SVM is extended and amended by adding some cost-sensitive settings which aimed to the reduction of the related risk. This extension is based on the links between risk minimization and the probability estimation. The cost-sensitive SVM imposes cost sensitivity for separable as well as non-separable training data, that because the cost-sensitive SVM imposes a greater margin for the preferred class.

- **Cost-Sensitive Decision Tree Algorithms:** As the new algorithm developed in this research (ECSDT) is a Cost-Sensitive decision tree method; therefore, a separate more detailed sub-section related to this category has been allocated in section (3.1.1.3).

3.1.1.2 Meta-Learning Cost-Sensitive

Cost-insensitive classifiers can be changed into cost-sensitive ones by using cost-sensitive meta-learning algorithms without making any alterations to original cost-insensitive algorithms. Therefore, cost-sensitive meta-learning algorithms can be considered as a middleware element through which the training data is reprocessed, or the output is post-processed from the cost-insensitive learning algorithms.
Various methods are adopted by the cost-sensitive meta-learning algorithms for the induction of cost-sensitive classifiers. It has been shown in the literature that there have been various attempts to group the cost-sensitive meta-learning algorithms into some correlated categories as presented in (Sheng & Ling 2006; Kotsiantis 2007; Song & Lu 2015; Ting 2008; Ling & Sheng 2008; Turney 2002). Some of the widespread categories of cost-sensitive meta-learning algorithms are sampling methods and non-sampling methods, and these are developed based on whether the distribution of the training data is modified or not in accordance with the misclassification costs. The following paragraphs describe these methods.

Non-sampling methods can be further categorized into three sub-classes: relabeling, weighting and threshold adjusting. These are explained as follows:

- **Relabeling:** In these methods, the class-labels of the examples are relabeled in accordance with the least expected cost criterion (Fulkerson et al. 1995). MetaCost

(Domingos 1999) is one of the most well-known cost-sensitive algorithms that adopts the relabelling method that uses the concept of bagging (Breiman 1996) by creating n resamples of the data set (with replacement), then uses the learning process for every sample, and after that accumulates the findings for the n samples which gives more accurate findings. The training examples are then relabeled with their predicted least cost classes. The error-based learner is subsequently applied to the new training set to produce the ultimate cost-sensitive classifier (Domingos 1999; Vadera 2010). The same equation 2.13 given earlier is used to calculate the expected cost.

- **Weighting:** In this method, a weight is allocated to each instance of the training data, which represents the impact of misclassifying an instance with respect to the cost incurred. One of the well-known examples using this method is an algorithm known as C4.5CS that was put forward by (Ting 1998) for inducing cost-sensitive trees through instance weighting by considering the C4.5 algorithm as the base classifier. Therefore, with C4.5CS, high weights are allocated to the examples of the rare class that have a greater misclassification cost. When C (j) is considered as the cost of misclassifying an example belonging to class j, then the following equation is used to calculate the weight of each example belonging to class j:

$$W(j) = C(j) * \frac{N}{\sum C(i)N_I} \qquad 3.1$$

- **AdaBoost (Adaptive Boosting):** is another cost-sensitive classifier which uses the weighting method (Yoav Freund & Schapire 1999). This algorithm is a cost-sensitive boosting method that consists of generating several hypotheses h_t and then integrating them to create a more accurate composite hypothesis having the form given in the equation below (Yoav Freund & Schapire 1999):

$$f(x) = \sum_{t=1}^{T} \propto_t h_t(x) \qquad 3.2$$

Here, $\propto_t$ represents the degree of weight that should be given to $h_t(x)$. AdaBoost creates $h_i(x)$ in sequential trials using a learning system on weighted instances which are indicative of their significance. In the beginning, weights equal to the value of (1/N) are allocated to each example, where N is the number of training examples. The weights are adjusted at the end of each trial so as to increase the weights of misclassified examples; however, there will be a decrease in the weights of correct examples. Once a certain number of cycles have occurred, a series of trees or

hypotheses $h_i s$ are presented and can be integrated to carry out classification. The classification that is eventually chosen is one that relies on choosing the class that provides the highest weighted vote.

- **Gradient Boosting with Stochastic Ensembles (GBSE):** The target of the most boosting based algorithms is to sort out binary classification problems in which the classification algorithms can set the weights for the training examples in ratio to the cost of misclassifying them. Nevertheless, in multiclass classification problems, an instant could be misclassified into more than one class; therefore, the demonstration of the weights is difficult and unobvious. To solve this issue, an approach known as Gradient Boosting with Stochastic Ensembles (GBSE) was introduced in (Abe et al. 2004) that makes use of boosting. GBSE induces a stochastic hypothesis *H (y | x)* for a class (y) that labels an instance (x). This hypothesis is founded on the individual hypotheses $h_t(x)$ that is attained by the following equation:

$$H(y\backslash x) = \frac{1}{T} \sum_{t=1}^{T} I(h_t(x) = y) \qquad 3.3$$

  Where the value of I (E) will be 1 if the expression E is true and will be 0 if the value of the expression E is false.

- **Threshold Adjusting:** These types of techniques utilize certain threshold functions to classify examples as positives or negatives by applying particular probability estimates (Elkan 2001), (Ling et al. 2006). Through thresholding, the ideal probability from the training examples is identified as the threshold, which is used to determine the class label of the test instances. A test example whose estimated probability is more than or the same as this threshold is considered as positive, while it is negative when it is the opposite.

In contrast to the above methods, sampling methods modify the class distribution of the training data with regards to the cost function and then apply cost-insensitive classifiers on the re-sampled data for the classification. The basis of sampling methods is the assumption that, the misclassification costs for the minority-class instances are more than the misclassification costs of the majority-class examples. Therefore, through the sampling methods, the class distribution of the training data is altered so that it becomes more balanced, which makes the minority class more significant (Weiss et al. 2007; Li et al. 2005). With this approach, two

fundamental sampling techniques can be used; over-sampling and under-sampling. In over-sampling, the number of the minority-class examples are increased by replicating some of them, while in under-sampling, the number of majority-class examples is decreased by disregarding some of them (Elkan 2001).

Costing (Zadrozny et al. 2003) is one of the common algorithms that use the sampling method. The Costing algorithm implements a base learner over a sample of the data so that alternative classifiers can be produced. The purpose of every resample is to alter the distribution of the data so that the minimizing of the error rate on the modified distribution becomes equal to the minimizing of the cost of the initial distribution. It is depicted in (Zadrozny et al. 2003) that there was replication of cases in the training data caused by the proportional sampling with replacement, which then produces what is known as the over-fitting problem in the model building, so that it has been suggested in (Zadrozny et al. 2003) that "rejection sampling" should be used to prevent the replication. That by ensuring that, every instance in the initial training set is drawn one time, and it is included in the sample with the accepting probability C (j, i) / Z, where C (j, i) represents the misclassification cost of class i, and Z signifies the maximum cost of misclassifying an example. When the highest cost is given by Z, then this is the same as maintaining all instances of the rare class, and sampling the majority class without replacement.

3.1.1.3 Cost-Sensitive Decision Tree Algorithms

There are usually two stages in a typical decision tree algorithm, decision tree growing and decision tree pruning. The decision tree algorithm can be altered so that it becomes cost-sensitive in both stages. When the decision tree is in the growing stage, the split criterion is usually altered so that costs are considered within the splitting criterion. In this field, researchers introduced various decision tree algorithms which consider the misclassification costs when creating the splits. Some common examples of such algorithms are briefly explained in the following sub-paragraphs.

- **IDX:** IDX (Norton 1989) refers to a decision tree algorithm in which a look-ahead strategy is used that considers n tests ahead. Here, n refers to a parameter that the user may fix. For splitting the data, IDX selects the attribute that maximizes the following function:

$$C = \frac{I_i}{C_i} \quad 3.4$$

Here, I_i is the information gain that is related to the i^{th} attribute at a particular stage in developing the decision tree, and C_i signifies the cost of calculating the i^{th} attribute.

- **EG2:** EG2 (Nuñez 1991) is a decision tree induction algorithm that adopts the Information Cost Function (ICF) for choosing attributes, and it is a modified form of the ID3 (Norton 1989). In EG2, the attributes are chosen by ICF on the basis of their information gain and their costs, and the overall cost can be decreased by choosing attributes with less test cost and higher information gain for the splitting. Different kinds of generalization are implemented by EG2, which simultaneously decreases the classification cost by using the previous knowledge that gives information for carrying out economic induction and presents limitations and directions for creating more general and appropriate decision trees. The following equation is used to describe the ICF for the i^{th} attribute:

$$ICF_i = \frac{2^{I_i}-1}{(C_i+1)^w} \quad 3.5$$

Here, $\boldsymbol{I_i}$ is the information gain related to the i^{th} attribute at a specific stage in the development of the decision tree, and $\boldsymbol{C_i}$ is the cost of measuring the i^{th} attribute, while w is a modifiable parameter, having a value between 0 and 1.

- **Cost-Sensitive ID3 (CS-ID3):** CS-ID3 signifies a cost-sensitive technique that is formulated on the basis of the ID3 decision tree algorithm (Quinlan 1986). It was explained earlier that ID3 chooses the attribute that increases information gain to be used for the splitting. ID3 is altered by CS-ID3 so that the feature selection measure includes the cost in addition to the information gain, choosing the split attribute that gives the highest value for the function given below:

$$C = \frac{I_i^2}{C_i} \quad 3.6$$

Here, $\boldsymbol{I_i}$ is the information gain related to the i^{th} attribute at a particular stage in the formulation of the decision tree and $\boldsymbol{C_i}$ is the cost of measuring the i^{th} attribute.

- **ICET:** ICET (Turney 1995) is an abbreviation of Inexpensive Classification with Expensive Tests. ICET utilises a genetic algorithm known as GENESIS (Grefenstette 1986) with an extension of the C4.5 algorithm (Hormann 1962) to include the misclassification costs into the fitness function obtained from the Information Cost Function (ICF) that is used with the EG2 system (Nuñez 1991) as a substitute measure of the information gain. Attributes are chosen by the ICET on the basis of their information gain and their cost, and the form given below is adopted by the ICF for the i^{th} attribute:

$$\mathrm{ICF_I} = \frac{2^{\Delta_i}-1}{(\mathrm{C_i}+1)^{\mathrm{w}}} \qquad 3.7$$

  Here, Δ_i is the information gain related to the i^{th} attribute, C_i is the cost of determining the i^{th} attribute and w signifies a bias parameter that regulate the degree of weight that should be awarded to the costs. Its value is between 0 and 1.

  The ICET constructs various trees, and these trees are then examined on an internal holdout data. Once various trials have been carried out, the ideal set of test cost determined by the genetic search is used to develop the ultimate decision tree for the complete training data set.

- **Test Cost-sensitive Decision Tree – TCSDT:** TCSDT (Ling et al. 2004) is a method formed on the basis of the C4.5 algorithm and for the purpose of developing and testing decision trees. Misclassification costs and tests costs are both directly used by the TCSDT in the process of developing the decision tree. That is, with TCSDT, the best attribute by the expected total cost reduction is selected instead of minimizing the entropy in the attribute as in C4.5. Therefore, the best attribute that decreases the total sum of misclassification and test costs for each possible split will be chosen as the root of the tree or sub-tree.

- **CSTree:** There is a subsequent adaption of TCSDT (Ling et al. 2004) into CSTree (Ling et al. 2006) that considers just the misclassification costs while disregarding the test costs. CSTree aimed for solving and addressing two-class classification problems, and the probability of the positive class is calculated by considering the relative cost of the two classes. This is then used to obtain a value for the expected cost.

- **Cost-Sensitive Decision Trees with Multiple Cost Scales (CSDTWMCS):** (Qin et al. 2004) established a cost-sensitive algorithm, known as Cost-Sensitive Decision Trees with Multiple Cost Scales on the basis of the TCSDT algorithm presented in (Ling et al. 2004). The authors of CSDTWMCS (Qin et al. 2004) supposed that there are two types of cost included in the test and the misclassification costs i.e. target costs and resource costs. These two types of cost (target, and resource) were specified on the basis of two dissimilar scales; for example, as a dollar cost and as a time cost associated with a medical diagnosis. The authors also made an assumption that the resources have maximum bounds for each, these bounds known as resource budgets. The target of the approach is to reduce one type of cost whilst manage the other one within the prescribed budget. Each of the attributes has two parameters, that are, test cost and constraint possessed by, similarly, each kind of misclassification owe a cost and a constrained value. These two values are utilized in the splitting criteria to develop a target-resource cost decision tree and are also incorporated in some functions that include target cost reduction (test cost) and the consumption of resources used for attaining missing data. The total cost of choosing an attribute (a) for splitting is calculated using the following equations.

$$\mathrm{ICF_a} = (\mathrm{Cost_0} - \mathrm{Cost_a})/\mathrm{Constrain_a} \quad 3.8$$

$$\mathrm{Constrain_a} = (\mathrm{N} - \mathrm{m}) * \mathrm{r_a} + \mathrm{p} * \mathrm{C_{ij}}(\mathrm{r}) + \mathrm{n} * \mathrm{C_{ji}}(\mathrm{r}) + \mathrm{m} * \mathrm{C_{ji}}(\mathrm{r}) \quad 3.9$$

In which ($\mathrm{Cost_0}$) is defined as the misclassification cost prior to the splitting, ($\mathrm{Cost_a}$) is the predicted cost in case attribute (a) is selected, $\mathrm{r_a}$, Cij(r) and Cji(r) are the resource costs for FN and FP respectively, (p) is the number of positives, (n) the number of negatives and (m) the number of instances that have missing attribute values.

- **CSNL:** Vadera in (2010) presented a new non-linear decision tree learning algorithm called CSNL (Cost-sensitive Non-linear Decision Tree) which uses the misclassification cost for inducing the decision tree. The basis of the algorithm is the hypothesis that non-linear decision nodes give better outcomes for cost-sensitive decision tree induction compared to the axis-parallel decision nodes. In this algorithm, the discriminant analysis put forward by (Fisher 1936) is used to establish non-linear

classification boundaries that consider misclassification costs. CSNL algorithm is aimed at solving the only binary class problem, and it tries to determine a split that reduces ECM, the Expected Cost of Misclassification with the help of the equations given below:

$$\mathrm{ECM} = \mathrm{C_{j,i}} * \mathrm{P(j|i)} * \mathrm{P_i} + \mathrm{C_{i,j}} * \mathrm{P(i|j)} * \mathrm{P_j} \qquad 3.10$$

Here, $C_{i,j}$ represents the cost of misclassifying an example in class i, when it is really in class j; $P(i \mid j)$ represents the probability of categorizing an example in class i considering that it is in class j; and P_i represents the probability of an example in class i. The algorithm was evaluated on 17 datasets and its performance was contrasted to the most popular algorithms in this domain, i.e. ICET and MetaCost. It was depicted in the findings that the performance of CSNL is the same, if not better, than that of the other algorithms for the majority of the datasets used (Vadera 2010). The shortcoming of this algorithm is that it is used for just binary classification problems and concentrates on just the misclassification costs.

- **MNCS_DT:** A modified version of the CSNL algorithm that has been presented by Vadera in (2010) was put forward by (Duan & Ding 2016), this algorithm known as MNCS_DT, which refers to Non-linear Cost-sensitive Decision Tree for Multi-classification, which extends the binary classification problem in CSNL into multi-classification problem by developing non-linear split nodes using the latest discriminant analysis in decision tree for multi-classification problems. MNCS_DT uses altered information gain ratio that combines costs to choose best attributes whose altered information gain ratios are greater than the average one. The main aim of MNCS_DT is to reduce the total ECM, the Expected Cost of Misclassification using the equation 3.1.

$$\text{Total EMC} = \sum_{\mathrm{i,j,i\neq j}}^{\mathrm{n}} \mathrm{EMC_{i,j}} \qquad 3.11$$

Where, $EMC_{i,j}$ is the ECM related to each combination of two classes except when i=j, and is calculated utilizing the same Eq 3.12 introduced with the CSNL (Vadera 2010). The main drawback of this method is that multivariate normal assumption is not always valid, as If the dataset comes with only a few of variables that may not follow multivariate normal distribution then the MNCS_DT algorithm cannot perform better (Duan & Ding 2016).

- **ECCO:** Is a cost-sensitive evolutionary algorithm that was developed by (Omielan & Vadera 2012) in which the trees are directly encoded. ECCO is a shortcut for (Evolutionary Classifier with Cost Optimization) that incorporates certain features of the genetic algorithms, like crossover and mutation operators when developing the cost-sensitive decision trees. Unlike, ICET, ECCO uses binary strings to represent the decision trees, and then the typical crossover and mutation operators are applied on the binary strings. The Expected Cost of Misclassification ECM (as defined in Eq 3.12) is used as the fitness function.

3.2 Nonlinear Decision Trees

The main concern of the majority of the decision tree algorithms is to solve univariate classification problems in which axis-parallel tests are conducted at each decision node of the tree. When the classification problem is nonlinearly separable, the majority of the recent algorithms; for instance, classification algorithms in WEKA and R, tried to solve these nonlinear classification problems through incorporating linear separators to split the nonlinear search space (Vadera 2010). It had been recommended by many researchers in this area that this is not adequately expressive and usually results in more complex than multivariate splits (Lomax & Vadera 2011). The following sections present and highlight some related work and attempts for solving nonlinear classification problems.

Recently, a number of optimization methods have been developed to construct non-linear classifiers for some large-scale applications, but instead of adopting direct non-linear classification, most utilize methods that in some way show the non-linear distribution of the classes to be represented as a new linearly separable distribution. Among those methods are

the algorithms that adopt Kernel Tricks (Passerini 2013) that make linear models work in nonlinear settings by mapping each instance x to a higher dimensional vector y(x) where it can exhibit linear patterns and then apply the linear model for the new classification space (Mika et al. 1999). To illustrate the idea of how Kernel Tricks work, let us consider the hypotheses binary classification problem illustrated in Figure 3.1. In this problem each example is defined by a two features **x** = {x1, x2} and it is obvious that linear separator is not an adequate solution for this data, so by using (Keren Tricks) each example in the feature space can be mapped into a higher dimension space Z that may take one of the forms shown below:

$$x = \{x1, x2\} ==> z = \{X_1^2, \sqrt{2X_1X_2}, X_2^2\} \quad 3.12$$

Then the results of the new feature space can be similar to the one illustrated in Figure 3.2 which is a linearly separable space.

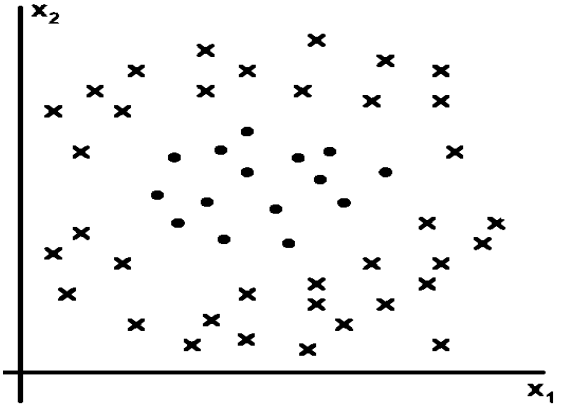

Figure 3.1: Classification problem before applying Kernel tricks

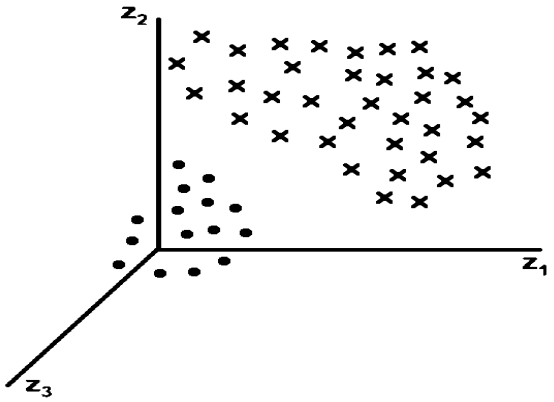

Figure 3.2: Classification problem after applying Kernel tricks

Kernel Tricks have been exploited by many classification algorithms, including Support Vector Machines - SVM (Fu et al. 2010; Suykens 2001), and Fisher Discriminant Analysis - FDA (Mika et al. 1999; Dai et al. 2006) which are briefly described below.

The study presented in (Fu et al. 2010) introduced a new linear support vector machine (LSVM) combination pattern known as Mixture of Linear Support Vector Machines (MLSVM) that uses a divide and conquer policy for separating the feature space domain into sub-domains of linearly separable data points and then applies a SVM for each of these sub-domains. A generative pattern is attained by their approach through the utilization of the joint data and class-label allocations.

Dai et al. (2006) propose a Fisher discriminant analysis algorithm, known as Heteroscedastic Kernel Weighted Discriminant Analysis (HKWDA) for implementing nonlinear features extraction for classification frameworks. HKWDA can cater for non-homogenous data that are commonly found in real-world applications and pays high attention to the classes that are characterized by multiclass classification problems through the integration of a suitably selected weighting function into the discriminant function. HKWDA proposes that classes that are closer to each other in the feature space which may cause a possible reduction in the classification performance should be given more weights in the input space.

3.3 Multiclass Classification

With Multiclass classification, each example belongs to one of three or more available classes. There are many applications of multiclass classification involving text script classification, speech recognition, objects recognition, etc. (Ou & Murphey 2007). Some examples of multiclass problems from the UCI repository (Lichman 2013) that are commonly used in the area of machine learning are described below:

- The Soybeans classification problem in which each example of the dataset represents different characteristics of a crop of soybeans and the classification task is to make a prediction to which one of nineteen possible diseases affecting soybeans crop the new example should be classified.
- The IRIS classification problem in which a prediction for a new example is made to know to which one of three types of the IRIS flowers (Setosa, Versicolour and Virginica) the new example should be classified.
- The Glass identification problem in which the instances represent the chemical combination of some types of glasses and the classification task is to make a prediction for one type of glass among seven known types of glasses.

The literature in this field describes various techniques and algorithms that have been developed with the aim of solving multiclass classification problems. Mehra and Gupta (2013) and Aly (2005) survey and categorize these algorithms into different approaches. Some approaches extend and adjust the ordinary binary classification methods adopted by some algorithms such as Neural Networks (Ou & Murphey 2007), Decision Trees (Vens et al. 2008), Support Vector Machines (Hsu & Lin 2002) and K-nearest Neighbours (Min-Ling

Zhang & Zhou 2005) to solve the multiclass problem. Other approaches implement the theory of partitioning the multiclass problem into several binary classification problems, after that applying one of the ordinary binary classifiers for classifying each binary problem and then the results of all the binary classifications are combined together to obtain a final result for the multiclass problem.

Price et al. (1994) introduced a method based on neural networks for solving multiclass classification problems through separating the actual multi-classification problem into sub-binary-classification problems that consider only two classes, for each pair of classes, a (possibly small) neural network is learned by utilizing only the data of these two classes. The motivation for this method is to depict the ways in which the proposed approach unites the results of the two-class neural networks to attain the subsequent probabilities related to the class decisions. The findings of applying the method to some contemporary datasets show that the obtained results are comparable to the results obtained by applying other neural network techniques.

Friedman (1996) recommended another method for multiclass classification problems. In line with the Friedman's technique, initially, the problem is split into various binary classification problems and afterwards, sorts out each of the binary problems separately. Later, the overall pairwise decisions are merged to formulate a multiclass decision that is used for testing the test observations. There is no doubt that Friedman's combination rule is pretty much intuitive, it allocates the new unseen observation to the class that gets the most pairwise comparisons.

Mehra and Gupta (2013) and Aly (2005) both mentioned that Support Vector Machines (SVMs) are the most extensively utilized binary classifiers for solving multiclass classification problems. Originally, Support Vector Machines (SVMs) were developed by (Vapnik 2000) and aimed to solve only binary classification problems. Afterwards, many techniques were developed to offer solutions for solving multiclass problems by dividing multiclass problems into many binary classification problems (Mehra & Gupta 2013; Aly 2005). Some of these approaches include:

- **One-Against-All Multiclass Classification Technique:** With this approach classifying a problem with (K>2) classes is accomplished through separating the search domain into K binary problems. To attain the N^{th} classification model, examples related to the N^{th} class of the training set are considered as positives and the

rest of the examples are considered as negatives. Then the model yielding the highest classification results is treated as the winner.

- **One-Against-One Multiclass Classification Technique:** In this approach, if the classification problem comprises of K classes then a total of (K * (K-1) / 2) classifiers are required. At each time of building a different binary classifier, the algorithm will consider data of one class as positives and data of another class as negatives. This procedure is recursively repeated for each possible pair of a combination of the overall classes; whereas, the remaining classes are ignored. To assess and predict a new instance, voting is carried out between the classifiers and the model with the highest votes becomes the winner.

- **Directed Acyclic Graph SVM (DAGSVM):** Platt et al. (2000) proposed this method and it is established on the idea of Decision Directed Acyclic Graph (DDAG) structure, which is arranged in a tree-like structure. DAGSVM is similar to the one against one technique in its training modules. For a K-class classification problem, the number of obtained binary classifiers is equal to k (k-1)/2 and each classifier is learned for classifying two classes of interest. In the graph structure, at each node, a binary-class SVM is used. Nodes in DDAG are in a form of a triangle possessing a single root node that is at the topmost and is constantly elevating with an increase of one in each layer till the last layer contains k nodes. An input vector x that begins from the root node and moves towards the next layer on the basis of output values, is assessed by the DDAG. The motivation for the DDAG technique is to discard one class from the list that involves all the classes as it progresses down the tree. The first class against the last class within the list is assessed by each node. In case the assessment leads to one class out of the two classes then it is preferable to discard the other one. Then the same procedure repeated for the first and the last class in the new list and so on. The procedure ends when only one class exist in the list. The class label related to the input data will be treated as the class label of the node in the final layer of the assessment path or either the class left on the list.

- **Error Correcting Output Coding (ECOC):** To resolve the multiclass classification problems, Dietterich and Bakiri (1995) implemented a new method called Error Correcting Output Coding (ECOC) on binary (two-class) classifiers. This technique is

applied by decomposing a multiclass problem into multiple binary classification problems. Firstly, an exclusive code-word that consists of a unique n-bit binary string is assigned to each class rather than assigning a class label, then a matrix called code-matrix that consists of k rows and n columns is generated, each row in the matrix holds the unique code-word for a particular class whereas each column is a binary string that represents one of the classifiers. When a new unseen example to be tested it is applied to n binary classifiers and is given an output-binary-code. Then a particular distance called Hamming distances which are the number of unmatched bit locations between the output-binary-code of the new example with all code-words is calculated, and then the class label for that example is the code-word with the lowest Hamming distance.

- **Hierarchical Classification:** Another approach for multiclass classification problem is called Hierarchical Classification (Mehra & Gupta 2013; Aly 2005). In this technique, a tree structure is used for the organization of the classes. The tree is designed in such a way that the parent node falls into two groups and both of them are child nodes. The process stops when a single class is determined at the leaf node. Every node of the tree contains a classifier mostly a binary classifier that distinguishes between the different child class groups. Similar examples of such approach are the BHS (Binary Hierarchical Classifier) approach proposed by Kumar et al. (2002), the BTS (Binary tree of SVM) method proposed by Fei and Liu (2006), the SVM-BDT (Support Vector Machines utilizing Binary Decision Tree) approach presented in (Madzarov et al. 2009) and the framework known as the DB2 (Divide-by-2) introduced in (Vural & Dy 2004). All these approaches use SVMs in the form of a binary tree structure in which a K-class problem is decomposed into K-1 binary-class problems through recursion. These methods begin constructing the tree from the root-node and moving from top to down. At each node, the number of available classes is divided into two groups according to the Fisher Discriminant (Fisher 1936) which leads to constructing a binary tree whose leaf nodes depict the actual K classes.

3.4 Evolutionary Optimization Methods

The literature shows that a massive number of intelligent optimization methods and algorithms are developed for solving and optimizing problems (Bhuvaneswari 2015; Zitzler & Thiele

1999). In the recent years the exploitation of evolutionary algorithms for inducing decision trees has witnessed a great interest by researchers in the field of data mining and knowledge discovery especially the use of Genetic Algorithms (GA), Genetic Programming (GP) and the Particle Swarm Optimization Algorithm (PSO) (Barros et al. 2012). With these algorithms instead of using the greedy top-down recursive partitioning strategy adopted by most decision tree induction algorithms, they perform a direct search in the space of candidate solutions (Barros et al. 2012). Evolutionary algorithms can make an evaluation for the solutions in a single run and the optimization process can be performed without any interventions from the decision makers to provide information regarding the preference of the objective (Ngatchou et al. 2005). The optimization will be achieved by utilizing particular mechanisms that nominate and store the best solutions for the problem under consideration (Carlos A Coello et al. 2010). Recent research works have demonstrated that genetic algorithm (GA) and particle swarm optimization (PSO) algorithms are amongst the most common distinguished algorithms that have powerful capabilities for optimizing problems (Von Lücken et al. 2014; Šeděnka & Raida 2010; Hassan & Cohanim 2005), and more details about these algorithms are explained in the following sections.

3.4.2.1 Optimization Methods Using Genetic Algorithms

The evolutionary calculation can be stated as a heuristic exploration technique, which promotes Darwin's principle of natural selection. For such sort of simulation of evolution, Genetic Algorithms (GAs) and Genetic Programming (GP) are among many other approaches (Zhao 2007).

The main idea of Genetic Algorithms was introduced by John Holland and his colleagues in the 1960s and mid of 1970s (Konak et al. 2006), where they illustrated that the way genetic algorithms work are influenced by the principles of genetics and evolution as they try to simulate the cloning behavior of biological populations towards solving the problem. The GAs utilize the concept of "survival of the fittest" where only strong individuals have greater chance to pass their features to the new reproduced generations. GAs adopted an approach that is carried out through a number of iterations (Manda et al, 2012; Bandyopadhyay & Saha, 2013), starting by generating and initializing random population, thereafter two genetic processes (crossover and mutation) are utilized for producing new future generations.

Crossover works by choosing from the population two individuals called parents with a preference towards fitness and swaps some features of them to produce two new instances called offsprings. Mutation plays a vital role to bring back and maintain the genetic diversity for the population to prevent premature convergence to local optima, and that is done by applying random changes into the characteristics of the individuals by turning over the characteristics of them randomly.

The literature shows several attempts that utilise genetic algorithms to induce cost-effective decision trees. A survey conducted by (Lomax & Vadera 2011) listed some of these algorithms. ICET (Turney 1995) (Inexpensive Classification with Expensive Test) is an example of the most well-known algorithms in this field. ICET is a cost-sensitive algorithm that utilizes the genetic mechanisms for the purpose of cost-sensitive classification. ICET considers both the cost of tests as well the misclassification costs and it approaches a combination of the greedy search with a genetic search algorithm. ICET utilises C4.5 with EG2's cost function to induce cost-sensitive decision trees, more details about ICET are given before in section (3.1.1.2).

Another example of a genetic algorithm is the ECCO algorithm (Evolutionary Classifier With Cost Optimizer) that is developed by (Omielan & Vadera, 2012), this algorithm uses a direct genetic implementation for inducing decision trees and has been shown empirically to produce decision trees that are more cost-sensitive than ICET and more details about ECCO are given before in section (3.1.1.2).

(Chen et al. 2011) presented a genetic algorithm-based technique to combine the connected weight optimization, parameter determination and feature selection in an evolutionary process. Moreover, for performance analysis, the cost preference is directly involved with the fitness function of the genetic algorithm. By using some contemporary data, the evaluation of the performance of the proposed algorithm is examined and the accomplished outcomes are further determined that the reduction of the number of features enhances the prediction ability.

Another study (Kr & Grze's 2007) in the same field presented an evolutionary approach to decision tree learning intended for cost-sensitive classification known as GDT-MC (Genetic Decision Tree with Misclassification Costs). The classical top-down approach produces the individuals in the initial population; however, the tests are selected by utilising particular

criteria known as the dipolar criteria (Bobrowski & Kretowski 2000). Amongst the feature vectors existing in the examined node, two objects from various available classes are randomly chosen. A functional test, that divides two objects into sub-trees, is created randomly by taking into consideration only attributes with various feature values. When stopping condition (that is based on the least number of learning vectors in a node or homogeneity of a node) does not occur, then the recursive splitting will continue. Lastly, on the basis of the fitness function, the resultant tree is post-pruned. In case the fitness of the best individual in the population does not improve during the fixed number of generations, the algorithm terminates. Moreover, there is a restriction on the maximum number of generations that bounds the computation time if the convergence is slow. The objective of the fitness function utilized in GDT-MC is to consider the anticipated misclassification cost with the size of trees. In reference to the principle, the genetic operators in GDT-MC are as the same as the cross-over and mutation operators but there is one difference i.e. they function on trees. There are numerous possible modifications of nodes that are permitted by the adopted mutation operators, involving replacing a test with an alternative dipolar test, swapping of a test with a descendent node's test, replacement of a non-leaf node by a leaf node, and development of leaf node into a subtree. The upcoming generation is accomplished through a linear ranking scheme, together with an elitist assortment policy.

In recent decades the role of genetic algorithms is extended to participate efficiently in solving the problems that have multiple objectives, as the literature review has revealed that there has been a massive effort directed to the development of a broad range of Multi-Objective Evolutionary Algorithms (MOEA's) that utilize the principles of Genetic Algorithms (Konak et al. 2006; Godínez et al. 2010). A wide range of these algorithms is surveyed and presented in (Bhuvaneswari 2015; Coello et al. 2007; and Von Lücken et al. 2014).

One of the most common multi-objective optimization algorithms is an algorithm called NSGA-II (Nondominated Sorting Genetic Algorithm II). NSGA-II - which was first introduced by (Deb et al. 2005), and perhaps it is the most popularly used in the current literature (Coello et al. 2010). NSGA-II aims to find multiple Pareto-optimal solutions in one single run. For optimizing a particular problem this algorithm mainly utilizes two mechanisms (elitism and crowded selection) to validate the quality of a given solution. The procedure followed by NSGA-II as stated in (Bandyopadhyay & Saha 2013; Coello et al. 2010) is

illustrated in the following points and depicted in the flowchart shown in Figure 3.4 (Bhuvaneswari 2015).

- The population is to be classified based on the fitness of the individuals into several nondomination fronts using a nondominated sorting, then elitism is recorded by reserving the best solutions that enhance convergence.
- A selection process is to be done based on crowding distance in which the density of solutions around a particular objective in the population is to be estimated.
- A solution is said to be better than another one if it is classified in a better rank, but if solutions classified in the same rank, then the one with better crowding distance will be selected.
- Then crossover and mutation operations will be accomplished by the selected solutions to produce the offspring population.

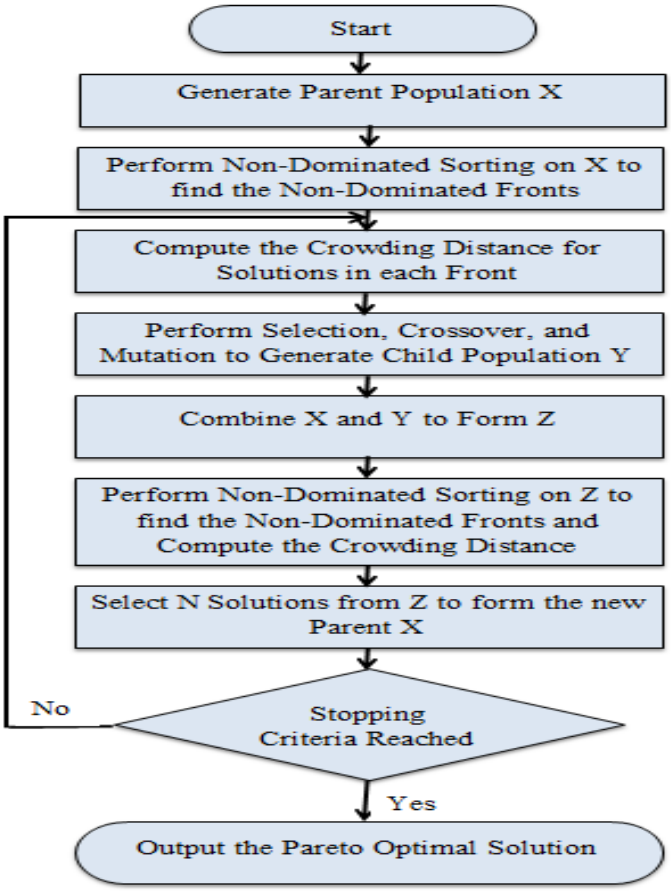

Figure 3.3: The flowchart of NSGA-II general procedure

Bhuvaneswari in (2015) pointed out some of the main advantages of applying NSGA-II for multi-objective optimization as presented below:

i. NSGA-II depends on non-dominated ranking mechanisms to provide solutions that are most close to the Pareto-optimal solution.
ii. Utilizes crowding distance mechanisms which are a measure of how close an individual is to its neighbours to ensure diversity in the solution.
iii. Employs elitist techniques in order to maintain and keep the best solution for the current population to be inherited by the next generation.

3.4.2.2 Optimization Methods Using Particle Swarm Algorithms

Particle Swarm Optimization (PSO) is another successful example of the evolutionary approaches for solving optimization problems. PSO adopts the idea of simulating the movement behaviour of flocks of birds or schools of fishes while searching for food, travelling or escaping from predators. That idea of PSO was first introduced in (Eberhart & Kennedy 1995). PSO represents an optimization solution in which an individual from the population is a potential solution to the collective goal (minimizing or maximizing a function (f)), and therefore to reach the collective goal, each particle collaborates with others in the swarm by exchanging information (Paquet & Engelbrecht 2003). Each particle x_i always remembers its best solution (pBest) achieved so far towards the optimization function y_{ou}, and the particle moves through the search space with a velocity v_i. In every iteration, the position of the particle and its velocity are adjusted according to its (pBest) and according to the best global solution (gBest) achieved so far by all the swarm towards the optimization function (Bai 2010).

The velocity and positions of the particle are updated using the following equations (Kennedy & Eberhart 1995).

v[ ] = w * v[ ]+[c1*rand( )*(pbest[ ] - present[ ])]+[c2*rand( )*(gbest[ ] - present[ ])] 3.13

Then, present[ ] = present[ ] + v[ ] 3.14

Here, v[ ] is the particle velocity, w indicates the inertia weight factor ,present[ ] is the current particle (solution), pBest[ ] and gBest[ ] are defined as stated before, rand( ) is a random

number between (0,1), c1 and c2 are the cognition and social learning factors and usually c1 = c2 = 2. The pseudocode of a general single-objective PSO is described in the following Figure 3.5.

```
FOR each particle
    Initialize particle
END FOR
DO
    For each particle
        Calculate fitness value
        If the current fitness value is better than the (pBest)
        set current value as the new (pBest)
    End For
    Choose the particle with the best fitness value of all the particles as the (gBest)
    For each particle
        Calculate particle velocity according equation (a)
        Update particle position according equation (b)
    End For
Repeat while maximum iterations or minimum error criteria are not attained
```

Figure 3.4: The pseudo code of general single-objective PSO

Recently, research and studies proved that Particle Swarm Optimisers are able to compete with the most famous methods used in the field of classification. A study was done by (Sousa et al. 2004), where three different particle swarm data mining algorithms were implemented and tested, and the results were compared with another algorithm such as J48. The results obtained at the end of the study show that PSO proved to be a suitable candidate for classification tasks and can obtain competitive results against J48 in the data sets used.

Another study carried out by (Ardjani et al. 2010) implemented a combination of support vector classifier (SVM) with particle swarm optimization (PSO) to optimize the performance of classifying some benchmark datasets. Therefore, the study illustrated that the new combination approach PSO-SVM resulted in better classification accuracy, even though the execution time is increased.

The literature review revealed that very little research was done in terms of the exploitation of PSO for inducing decision trees. One of the research studies is the paper presented by (Cho et

al. 2011), where it proposed a framework that uses an adaptive PSO to simultaneously determine the optimal thresholds of selecting variables for a decision tree (splitting criteria should be searched simultaneously); after that, the algorithm is validated using some artificial and real-world data sets. The results of the study show that, by using PSO, the computing process can be reduced and the optimal decision tree can be obtained with an improvement in the classification performance. More recently, the power of swarm intelligence has been introduced for effectively solving multi-objective problems (Ngatchou et al. 2005). As indicated in the survey work reported in (Coello & Reyes-Sierra 2006), all the present MOPSOs are featured with some new characteristics listed below:

- The non-dominated solutions (leaders) found during the search are archived externally to keep them for future process.
- Using voting mechanisms and some quality measures for choosing the appropriate leaders of the swarm.
- Using the neighbourhood topologies for calculating the crowding distance factors.
- An optional mutation operator can be applied.

(Chen et al. 2014) offered an approach using the PSO algorithm for the optimisation of the classification accuracy accomplished using the C4.5 classifier, this approach is called PSOC4.5. The motive of this approach is to assemble the PSO method due to its tremendous search ability with the C4.5 for its powerful knowledge discovery and interpretation characteristic. To analyse the performance of the PSOC4.5 approach, it experimented on 11 microarray datasets, and for validating the PSOC4.5, a five-fold cross-validation technique has been utilized. Besides, a comparison is carried between the performance of the approach with the other well-known algorithms in this area, i.e., SVM, Self-Organizing Map (SOM), Back Propagation Neural Network (BPNN), and C4.5. The attained statistical outcomes of the proposed algorithm have overtaken the results offered by the other techniques in terms of classification accuracy.

Another particle swarm optimization method for cost-sensitive attribute reduction problem is the one introduced in (Dai et al. 2016) which is inspired by the PSO powerful search ability. In perspective of the main goal of the proposed algorithm, that is finding any one of the lower test cost models, each particle in the PSO is considered as a binary string that depicts a subset of attribute set and also specifies the fitness function taking into account positive region of rough set and the weights of the testing costs. The suggested method in (Dai et al. 2016) is

examined with three standard test cost distributions. The outcome of applying the new method pointed out that the proposed algorithm is effective and adequate for cost-sensitive attribute reduction problems.

(Carlos A Coello et al. 2010) describe the pseudocode of a general multi-objective PSO as described in the following Figure 3.6 (Coello & Reyes-Sierra 2006).

```
Initialize the Swarm
Initialize the Leaders-Archive
Determine the Leaders-Quality
While termination-condition not achieved Do
    For each particle Do
        Select Leaders
        Update Positions
        Do Mutation
        Do Evaluation
        Update Pbest
    End for
Update the Leaders-Archive
Determine the Leaders-Quality
End while
Return the final Archive
```

Figure 3.5: The pseudo code of general multi-objective PSO

Optimized MOPSO (OMOPSO) is one of the well-known examples of multi-objective optimization algorithms that utilize the principles of PSO. This algorithm was firstly suggested by (Sierra & Coello 2005). OMOPSO uses the external archive based on the crowding distance for the purpose of selecting the leaders and the mechanism of Pareto dominance to limit the number of solutions stored in the archive as well as using the mutation operators to accelerate the convergence of the swarm. This algorithm employs two external archives: the first one for archiving the current leaders for proceeding the flight and the other one for archiving the final best solutions found during the entire search.

3.5 Summary of the Literature Review

As the main objective of this research is the development of a new nonlinear cost-sensitive decision tree algorithm for multiclass problems by utilizing some multi-objective optimization methods, this chapter was divided into four main parts that presented a review of Cost-Sensitive Decision Trees, Nonlinear Classification, Multiclass Classification and Multi-Objective Optimization.

The first part presented some details about cost-sensitive learning through which various techniques and strategies adopted by the previous work were reviewed and analysed. This section explained how important it is to consider different types of costs when building classifiers. Also in this section, one of the common ways for categorising cost-sensitive algorithms into direct and indirect methods was presented. The literature review for cost-sensitive learning ended up with a detailed sub-section that presents cost-sensitive decision tree learning methods.

For the second part of the literature review, the topic of nonlinear decision tree learning was covered which showed some different attempts and studies adopted by different authors towards solving nonlinear classification problems. One of the main concepts presented in this part is the concept of kernel tricks that can sometimes enable the use of linear models for nonlinear problems.

The third part of the literature reviewed the previous work related to multiclass classification and various techniques and algorithms that have been developed with the aim of solving multiclass classification problems were described.

The fourth part explored and presented some evolutionary algorithms for inducing decision trees and focused mainly on two well-known evolutionary algorithms namely: Genetic Algorithms (GAs) and the Particle Swarm Optimization Algorithms (PSO).

Based on the review, the key observations are:

- Considering cost when solving classification problems is very important and plays a vital role in getting optimum solutions for cost-sensitive classifications problems. There are different types of costs, such as misclassification costs, test costs and costs

due to delays in decision making. Misclassification costs are the most obvious and significant costs, and hence the focus of this thesis

- The majority of current classification algorithms use axis parallel univariate divisions to separate the data, which may cause some problems when the problem to be solved is nonlinearly separable. The literature in this field shows that there is a need to introduce and develop new nonlinear classification methods that introduce nonlinear boundaries to solve nonlinear classification problems.
- To classify multiclass problems, most of the current algorithms do not introduce a direct solution for the multiclass problems, but instead, use some indirect tricks and methods through which the multiclass problem is divided into several sub-binary problems. The solution of the binary classifiers is then combined to establish the final solution for multiclass problems. Hence, there is a need to introduce and develop direct methods for solving multiclass classification problems without dividing them into sub-binary problems.

In the next chapter, the different stages of developing the new algorithm (ECSDT) for nonlinear multiclass cost-sensitive classification are presented.

Chapter 4 : DEVELOPMENT OF THE NEW ALGORITHM

4.1 Introduction

As mentioned in the literature review chapter, most of the current cost-sensitive algorithms aim to solve the multiclass problem by adopting solutions that divide the multiclass problem into sub-binary problems and then classifying them by the binary classifiers (Mehra & Gupta 2013; Aly 2005). In addition to that, as presented in (Vadera 2010), the majority of cost-sensitive algorithms, such as decision tree learning methods, deal with non-linear classification problems using linear separators which are not adequate for solving such problems. As pointed out in Section 3.2, the literature shows a number of methods that produce indirect non-linear classifiers like Kernel methods, Support Vector Machines, and Neural Networks (Fung et al. 2002; Lee & Mangasarian 2001). These methods adopt some tricks to make linear models work in nonlinear settings by mapping each instance x to a higher dimensional vector y(x) where it can exhibit linear patterns and then apply the linear model for the new space (Mika et al. 1999). These methods have shown some success in many cases, but drawbacks remain including:

1. Mapping each instance in large datasets to a higher dimensional vector can generate massive storage problems.
2. Using linear separating methods for non-linear problems can result in very big decision trees.
3. Interpreting the classifier's output can be challenging, so, for example, neural networks are not very transparent and decision trees can become very large when viewed with axis parallel boundaries.

This chapter develops a new algorithm for cost-sensitive classification for multiclass problems based on the use of evolutionary optimisation methods such as GA and PSO. The new algorithm is called Elliptical Cost-Sensitive Decision Tree (ECSDT) and utilises elliptical boundaries in the nodes of decision trees for splitting the data. This chapter is organised as follows. Section 4.2 formulates the problem for optimisation, presents some background, and shows the pseudocode for the new algorithm, Section 4.3 describes how decision trees are constructed using ellipses, Section 4.4 describes the implementation of the algorithm using a

framework known as Multi-Objective Evolutionary Algorithms (MOEA) (Hadka 2014), and Section 4.5 gives some details of how the new algorithm works with illustrative examples.

4.2 Formulation of Cost-Sensitive Learning as an Optimization Problem

Most decision tree algorithms focus on linear axis parallel tests at each internal node of a tree (Ittner & Schlosser 1996; Vadera 2010), but for non-linearly separable classes where there is no good linear separator between the distributions of the classes, and even the best linear classifiers might not perform well, so using nonlinear classifiers for such cases are often more accurate than linear classifiers and may produce DTs that are smaller in size (Vadera 2010). To illustrate the idea of the new non-linear classification algorithm (ECSDT), let us consider the hypothetical 3-class classification problem shown in Figure 4.1 and Figure 4.2. As we see from Figure 4.1, we need 10 linear decision nodes (axis parallel lines) to get the optimal boundaries for splitting the classes, whereas, as Figure 4.2 illustrates, having non-linear regions (ellipses), would seem to be more appropriate for such cases and only 4 non-linear decision nodes (ellipses) are required to get the same classification accuracy.

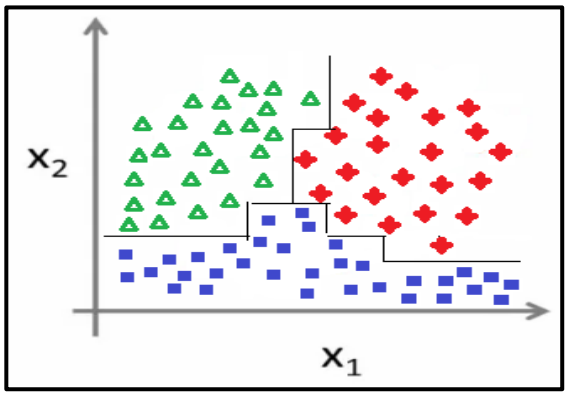

Figure 4.1: Linear classification

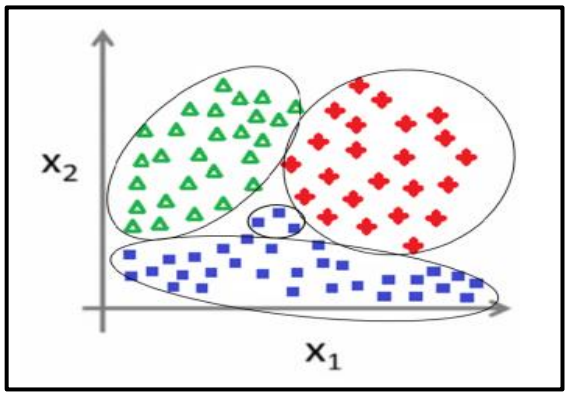

Figure 4.2: Elliptical classification

The new algorithm (ECSDT) developed in this research induces decision trees that utilise elliptical nonlinear decision boundaries, like those illustrated in Figure 4.2, for splitting the dataset instead of the traditional ways that adopt the axis-parallel splitting.

Given the intuition from the example in Figure 4.2, the main task of the new algorithm is to find a suitable number of ellipses and place them so that some measures such as misclassification error and cost are minimized.

As the new algorithm (ECSDT) proposes using ellipses, some important definitions related to geometric ellipse are given below:

A geometric ellipse can be defined as (Korn & Korn 2000):

> ***"The Figure consisting of all points for which the sum of their distances to two fixed points (called the foci) is a constant"***

Ellipses that are centred on the origin and the major and minor axes are parallel to the coordinate system (rotation angle = 0), can be defined as the shape created by all points that satisfy the following equation (Chandrupatla & Osier 2010).

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \qquad 4.1$$

Where x and y are the coordinates of any point on the ellipse; a and b are the radiuses of the ellipse on the x-axes and y-axes respectively.

For ellipses that are not centred at the origin and have no rotation, the general equation can be defined by the following equation:

$$\frac{(x-h)^2}{a^2} + \frac{(y-k)^2}{b^2} = 1 \qquad 4.2$$

Where x and y are the coordinates of any point on the ellipse; h and k are the centre point of the ellipse, (a) is the ellipse radius on the x-axis, and (b) is the ellipse radius on of the y-axis.

For rotated ellipses through an angle = (α), the general equation can be defined by the following equations:

Ellipses centred at the origin

$$\frac{(x\cos\alpha + y\sin\alpha)^2}{a^2} + \frac{(x\sin\alpha - y\cos\alpha)^2}{b^2} = 1 \qquad 4.3$$

Ellipses not centred at the origin

$$\frac{((x-h)\cos\alpha+(y-k)\sin\alpha)^2}{a^2}+\frac{((x-h)\sin\alpha-(y-k)\cos\alpha)^2}{b^2}=1 \qquad 4.4$$

Where x, y, h, k, a and b are as explained above and α is the rotation angle.

With the new algorithm, the above equations especially Eq 4.4 are used for the purpose of checking either a particular instance is located inside an ellipse or not. This is done by applying Eq 4.4 to a point (x,y) and if it results in a value less than or equal to 1, it can be assumed to be within the boundaries of the ellipse.

To formulate the optimisation problem, let us suppose that we focus on L ellipses to produce a solution (S). Suppose that we have two variables x and y that are selected from the available features, then, each ellipse in the solution is represented by five parameters x, y, a, b, and α which are the geometric parameters for the ellipse as described in equation 4.1 and equation 4.4. The general optimisation task for each of the 3 aspects can be formulated as:

For the first Aspect:

- **Maximising overall accuracy of Solution (S).**

  $$\text{Overall Accuracy} = \frac{Number\ of\ correctly\ classified\ examples}{Total\ number\ of\ examples}$$

For the second Aspect:

- **Minimising average cost per example for a solution (S).**

  $$\text{Overall Cost} = \frac{\sum_{i=1}^{n} \text{Conf}(i \mid j) * \text{Cost}\ (i\ ,j)}{Total\ number\ of\ examples}$$

  Where, n is the number of classes, Conf (i | j) is the number of examples from class i that have been misclassified as class j, and Cost (i | j) is the cost of misclassifying an example of class i as class j.

For the third Aspect:

- **Maximising overall accuracy of solution (S) considering cost as a penalty.**
  Where, Objective-Function = Overall Accuracy - Overall Cost

Placing the ellipses can lead to situations where some examples are not covered, hence to achieve the maximum overall accuracy of solution (S), and the minimum overall misclassification cost for a solution(S), two functions should be minimised for each solution(S) as follows:

$$\text{Minimise} \begin{cases} \sum_{i=0}^{N} \textbf{\textit{Misclassified examples}} & 4.5 \\ \sum_{i=0}^{N} \textbf{\textit{Unsclassified examples}} & 4.6 \end{cases}$$

Where N is the total number of examples used for training the classifier.

As mentioned earlier, in theory, different optimization methods, such as genetic algorithms (GAs) and Particle Swarm Optimisation (PSO), can be attempted to solve such multi-objective optimization problem. However, placing ellipses in only one level (there are no inner and outer ellipses) is not adequate in general, so, placing inner ellipses within ellipses as necessary leads to more accurate classification. For example, consider Figure 4.3 below in which only one level of ellipses is used, we can observe that some ellipses have errors related to other classes. In this case, placing inner ellipses within ellipses as shown in Figure 4.4 leads to more accurate classification and gives a clear vision about the construction of a decision tree. In such cases, the outer ellipses are called parent ellipses and the inner ellipses are called child ellipses.

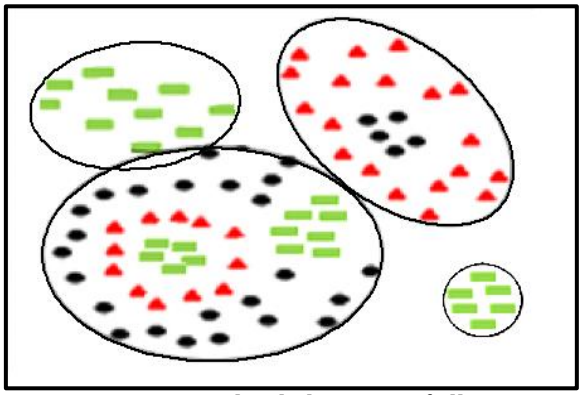

Figure 4.3: One level placement of ellipses

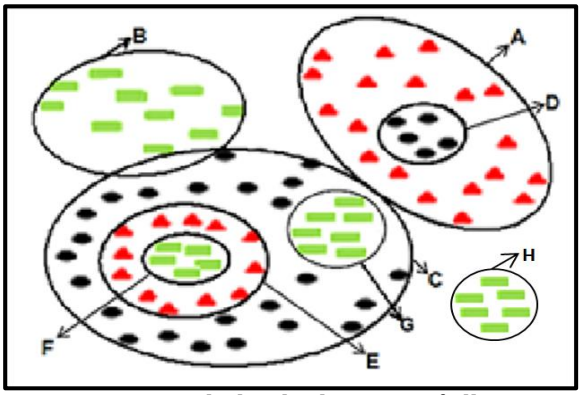

Figure 4.4: Multi-levels placement of ellipses

Based on the ideas and points explained above, an outline of the new algorithm that utilizes elliptical decision trees can be described in the following paragraphs and some illustrative examples of how (ECSDT) works are presented in Section 4.5.

The new algorithm (ECSDT) starts by initialising a pool of S solutions; where each solution consists of L ellipses; where each ellipse is represented by seven variables: the first two variables represent the two selected features used to obtain the geometric information for the ellipse, and the remaining five variables are the geometric parameters of the ellipse as described earlier. After initialising the pool of solutions, the algorithm makes a call to the optimisation method (GA or PSO) to improve the performance of each solution related to the pool and then only the parameters of the ellipses that form the best solution is returned back. Figure 4.5 below; shows the outline for the (ECSDT) algorithm.

1: Input:

A. Dataset = $\{ex_1, ex_2, \dots \dots, ex_n\}$ where each ex_i has features $\{f_1, f_2, \dots \dots, f_m\}$ and a class-label c_i.
B. Cost-Matrix = $\{Cs_1, Cs_2, \dots \dots, Cs_k\}$ where k= number of classes, and Cs_i = the cost of misclassifying $class_i$.
C. L = Number of the ellipses will be used to construct the tree.
D. Z = Population size: The number of solutions that will be optimized to reach the optimal solution.

2: Initialize the Pool with

- P = $\{s_1, s_2, \dots \dots, s_z\}$ where
- s_i is a solution that consists of a set of L ellipses that establish one possible complete tree, $s_i = \{ell_1, ell_2, \dots \dots, ell_e\}$.
- Each ell_i has seven parameters $ell_i = \{x_i, y_i, cx_i, cy_i, rx_i, ry_i, \alpha_i\}$.

3: Let Objective Functions = maximize accuracy and minimize cost.

4: Best Ellipses = Call related Optimizer (GA or PSO) with (Dataset, P, Objective-Functions, No. of cycles).

5: Best Tree = Call Conversion-to-Tree (Best Ellipses, Dataset)

6: Output: DT = Best Tree

Figure 4.5: The outline for the (ECSDT) algorithm

4.3 Constructing Trees from Ellipses

The algorithm presented in Section 4.2 results in a set of ellipses. This section presents how this set is transformed into a decision tree. Based on the general concept of building decision trees, which are constructed a from the top to the down, starting with the root node and then branching down based on relationships between the children-nodes and the parent-nodes. Building the decision tree using the new algorithm (ECSDT) depends on the identification of three types of relationships between the ellipses that establish the best solution. The first type of relationship that should be identified is the so-called "independence relationship", which determines which ellipses are independent and which ones are non-independent. Independent ellipse is an ellipse that is either does not intersect with any other ellipse or the ellipse that is partially intersected with another ellipse but it is not completely contained by the ellipses which intersect with it, while a non-independent ellipse is an ellipse that is completely contained within another ellipse.

The second type of relationship that should be identified and used in the construction of the tree is called the "parental relationship", which determines which of the ellipses are considered as parents, which ones are considered as children and which ones are considered as siblings. A parent ellipse is an ellipse that contains another complete ellipse (s) and a child ellipse as shown above with the child relationship, is an ellipse that is fully contained by another ellipse. Sibling ellipses are those ellipses on the same level and sharing the same parent.

More formally, an ellipse (e_i) is independent relative to a set of ellipses (Se) if:

independent (e_i, Se) = for all e_j in Se, not a child (e_i, e_j)

Where the child relationship is defined by:

child (e_i, e_j) = if e_i is completely contained by e_j

The third relationship that may occur between ellipses called the overlapping relationship, in which a partial intersection between two or more ellipses occurs.

Based on the explanation of the relationships given above, we can derive the following relationships between the ellipses shown in Figure 4.4:

- Ellipses A, B, C and H are independent ellipses.
- Ellipses D, E, F and G are non-independent ellipses.

- Ellipse D is a child of ellipse A, or in other words, ellipse A is a parent of ellipse D.
- Ellipse F is a child of ellipse E and ellipse E is a child of ellipse C, but ellipse F is not considered as a child of ellipse C.
- Ellipses A, B, C and H have a sibling relationship to each other.
- The child ellipses G and E also have a sibling relationship to each other.

ECSDT was developed, tested, and evaluated using three different approaches: (i) in the first approach only accuracy is considered as an objective function, (ii) in the second approach only cost is considers as an objective function, and (iii) the third approach considers accuracy that is influenced by cost as an objective function. Therefore, the tree construction varies slightly from one approach to another according to the main objective function. Figure 4.6 presents the outline for the methodology for constructing a decision tree regardless of the adopted approach.

Conversion-to-Tree (Best Ellipses, Dataset)

A. Let $data_i$ = data inside $ellipse_i$.

B. Let root-node = best independent ellipse.

C. Let decision-nodes = best ellipses according to the following priorities:
   - Independent before non-independent.
   - Ellipses with no children.
   - Ellipses with no intersection.
   - Ellipses with a better result (based on the adopted approach).
   - Parent ellipses come before child ellipses.

D. If a decision-node = an ellipse with no children like ellipses B, D, G and F, then Let:
   Left-Branch = Leaf-node with the majority class of the ellipse.
   Right-Branch = The following options according to order priority:
   - If the ellipse is independent like ellipse B, then
     Right-Branch = Decision-node with the next best independent ellipse (if any), OR, Right-Branch = Leaf-node with the majority class of the nearest independent ellipse.
   - If the ellipse is non-independent like ellipses D, G and F, then
     Right-Branch = Decision-node with the next best sibling ellipse (if any), OR, Right-Branch = Leaf-node with the majority class of the parent ellipse.

E. If decision-node = an ellipse with only one child like ellipses A and E, then Let:

Left-Branch = Decision-node with the child ellipse.
Right-Branch = following the same Right-Branch options listed in point D.

F. If decision-node = an ellipse with more than one child like ellipse C, then Let:
Left-Branch = Decision-node with the best child ellipse.
Right-Branch = following the same Right-Branch options listed in point D.

G. If decision-node = an ellipse that is the last ellipse on the path of one of the tree branches (has no child(ren) and no sibling(s) like ellipses D and F, then Let:
Left-Branch = Leaf-node with the majority class of the ellipse.
Right-Branch = Leaf-node with the majority class of the parent ellipse.

H. When an overlapping occurs between some ellipses, then the ellipse that has the best results according to the priorities shown in point C will be tested first before the others.

I. Repeat the processes in D, E, F, G and H recursively for both (Left-Branches & Right-Branches) for each Decision-node.

J. Return DT.

Figure 4.6: The outline for constructing the DT using (ECSDT)

For further explanation of the methodology used to construct a decision tree, let us assume that, ECSDT is adopting the first approach in which the only accuracy is considered as an objective function and let us also consider the example shown in Figure 4.4, then when applying the concepts that have been presented in Figure 4.6, the decision tree for the example shown in Figure 4.4 is as shown in Figure 4.7 below.

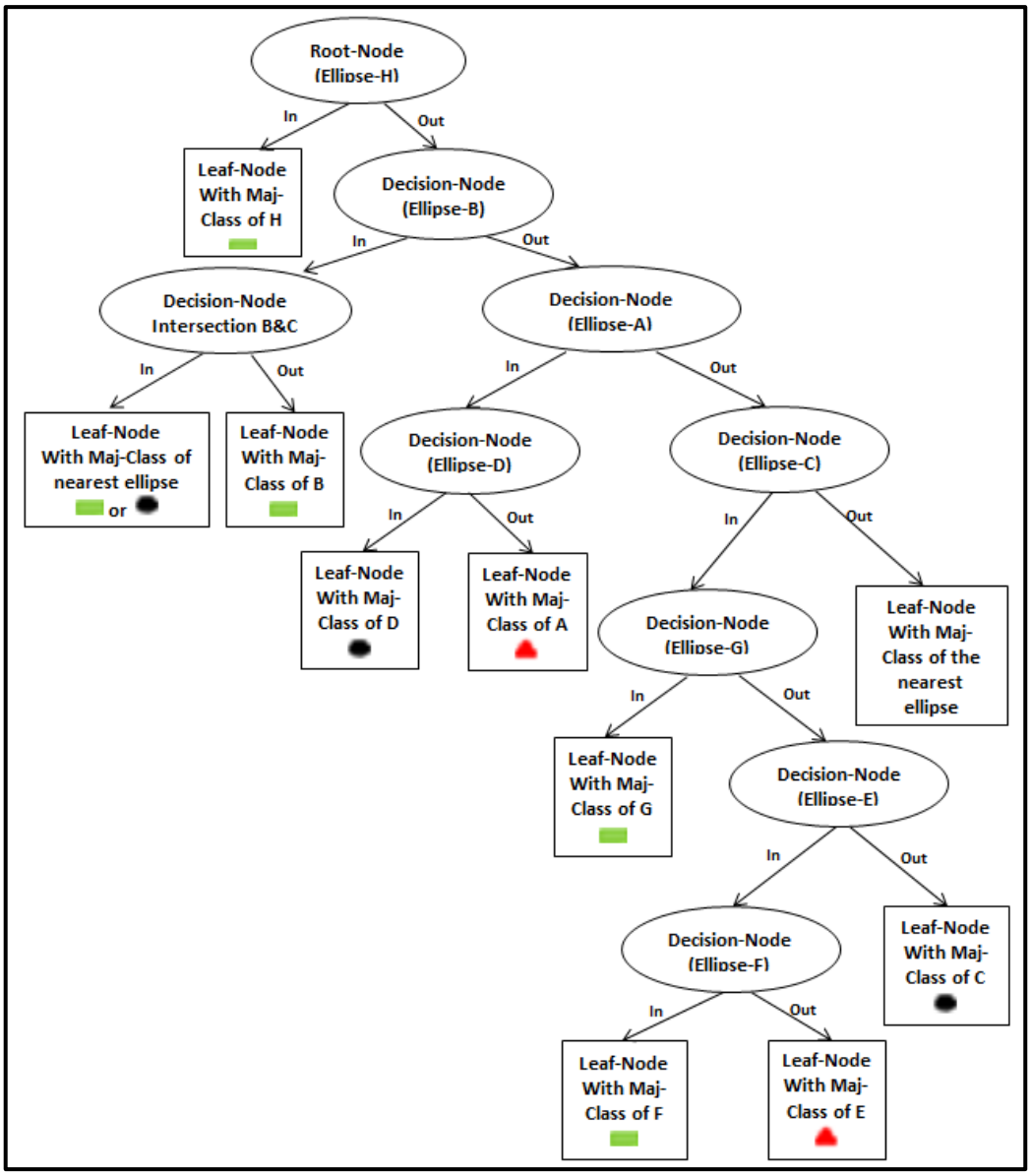

Figure 4.7: The decision tree for the example shown in Fig (4.4)

As mentioned above, the root node of the decision tree will be an independent ellipse with the highest accuracy among others. As we can observe from Figure 4.4, ellipses A, B, C and H are independent ellipses, that, they are not contained by any other ellipses, but ellipse H is the most likely to be the root node for the tree because it is a clean ellipse that has no children, no intersection with other ellipses and has the highest accuracy rate =100%. On the other hand, we can observe that ellipses D, E, F and G cannot be assigned as the root node of the tree because they are considered as non-independent ellipses (children-ellipses) because they are contained by other ellipses.

Given a DT with root (node), the algorithm for classifying a new example (x) is outlined in Figure 4.8. Some examples that illustrate this algorithm are given below and illustrated in Figure 4.9.

```
Classification (x, node)

Case-1: node is an ellipse with no-children & no-intersection;
        If x inside the ellipse
          Then {Return class-label as the Maj-class of the ellipse;}
        Else
          { If node ≠ last-node
              Then {Classification (x, right -node) ;}
             Else {Return class-label as the Maj-class of the nearest ellipse;}
          }

Case-2: node is a parent ellipse;
        If x inside the ellipse
          Then { If x inside any child of the ellipse
                   Then{ Classification (x, left-node );
                   Else { If node is an overlapped ellipse;
                            Then { If x inside the intersection
                                     Then { Return class-label as the Maj-class of the nearest
                                                  ellipse;}
                                     Else { Return class-label as the Maj-class of the ellipse;}
                                   }
                            Else { Return class-label as the Maj-class of the ellipse;}
                          }
                 }
        Else { If node ≠ last-node
                 Then {Classification (x, right -node); }
               Else {Return class-label as the Maj-class of the nearest ellipse;}
             }
          }

Case-3: node is an overlapped ellipse and not a parent;
        If x inside the ellipse
          Then { If x inside the intersection
                   Then {Return class-label as the Maj-class of the nearest ellipse;}
                 Else {Return class-label as the Maj-class of the ellipse;}
        Else { If node ≠ last-node
                 Then {Classification (x, right-node);}
               Else {Return class-label as the Maj-class of the nearest ellipse;}
             }
```

Figure 4.8: The outline for classifying new examples using ECSDT

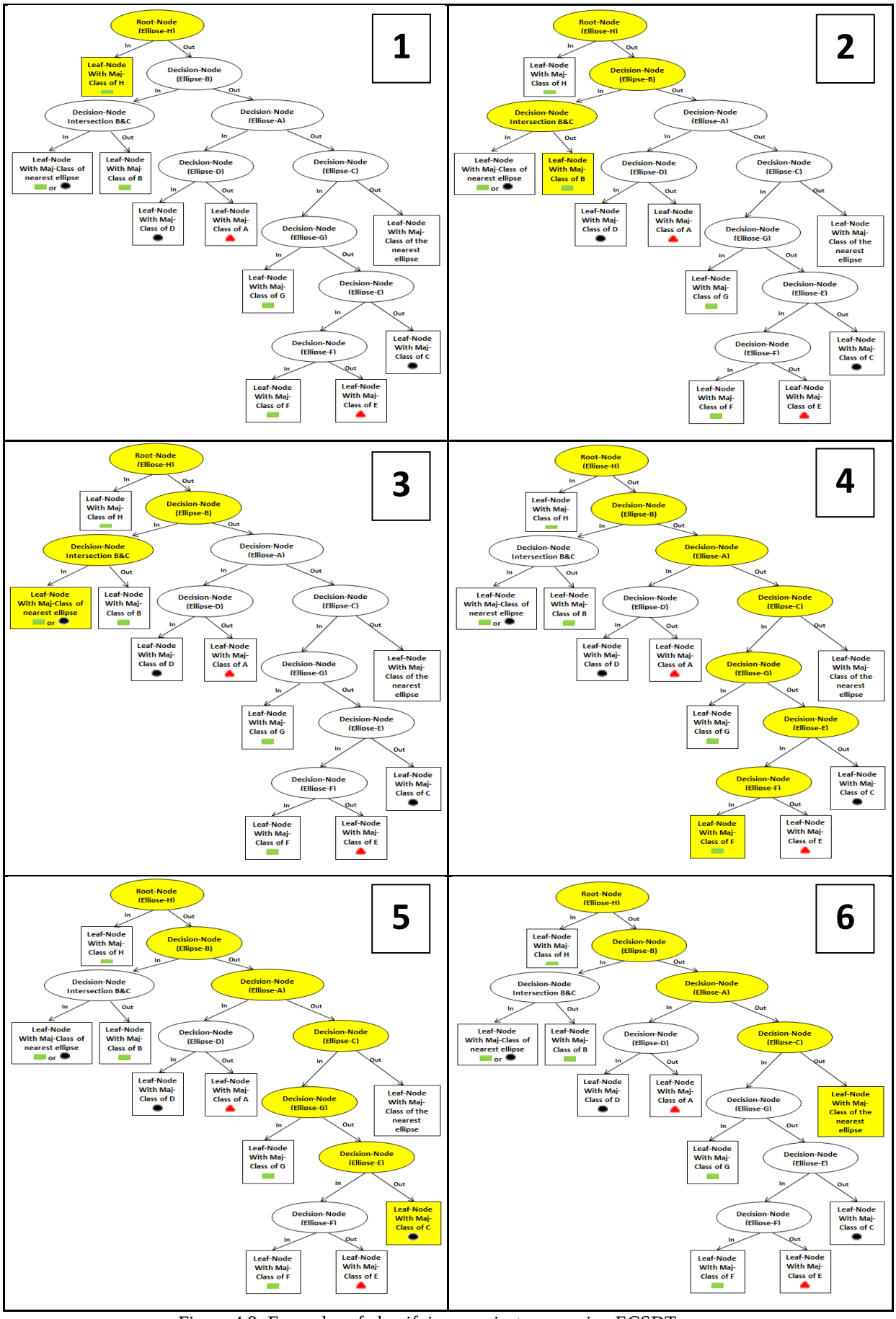

Figure 4.9: Examples of classifying new instances using ECSDT

Based on the outline algorithm in Figure 4.6 for constructing a DT and on the outline algorithm in Figure 4.8 for classifying new examples, Figure 4.9 above; illustrates 6 different examples that show the path followed to make a prediction for 6 unseen examples. The figures marked 1, 2, 4 and 5 show the paths followed to make a prediction for a new example whose feature values fall within the boundaries of ellipses H, B, F, E and C respectively. The figure marked as 3 shows an example whose feature values fall within the overlapped area between ellipse B and ellipse C, and the figure marked as 6 shows the path followed to make a class prediction of any new example whose feature values do not fall within the boundaries of any of the ellipses.

As mentioned before, a multi-objective framework called MOEA was utilised to implement ECSDT. Section 4.4 presents this framework and how it is utilised for optimising the problem and Section 4.5 presents details of the implementation of the new algorithm with illustrative examples.

4.4 Implementing ECSDT in the MOEA Framework

The ECSDT is an evolutionary algorithm because it has been developed by exploiting the evolutionary approach of two evolutionary algorithms, namely NSGAII which is a genetic algorithm (GA) and OMOPSO which is a particle swarm optimisation (PSO) algorithm.

The optimisation part of ECSDT is implemented using a framework called MOEA. MOEA is an abbreviation of (Multi-Objective Evolutionary Algorithms) and is available as an open source Java library developed by Hadka (2014) which can be easily imported into common Java development platforms such as Eclipse or NetBeans. MOEA comes with a vast collection of tools and packages that help developers design, develop, execute and statistically test the performance of common evolutionary optimization algorithms on different optimization problems. One of the powerful features of the MOEA framework is the ability for developers to identify and introduce their new own problems into the MOEA framework to be optimized

and this is the aspect in which this framework has been utilized in the process of developing the new algorithm.

The optimisation process for a particular problem in MOEA is accomplishing in two main steps or phases. The first phase is to define the problem to be optimised, and the second phase is to execute the problem:

- **Defining the problem:**

  To define any problems in MOEA, a class related to the framework called "Problem interface" should be implemented indirectly by extending another class called "AbstractProblem". This class provides the methods for describing and representing the problem and then evaluating solutions to the problem. The **"AbstractProblem"** class includes two main methods required for defining the problem as stated below:

  - The **"newSolution"** method which is responsible for producing new instances of solutions for the problem by defining the bounds for the decision variables used to solve the problem. At the end of this method, a new solution instance will be returned.
  - The **"evaluate"** method which is responsible for evaluating solutions to the problem that have been generated by the optimization algorithm. Thus most of the definitions and methods related to the problem are defined inside this method as well as setting the objective functions to be optimized and the constraints of the problem.

- **Executing the problem:**

  To execute the problem using an optimization algorithm, a class called **"executer"** related to MOEA is used. The executer class requires, at least, three pieces of information:

  - The problem to be optimized (which is the problem defined in the previous steps), and it can be called by the **"executer"** using a direct reference to the problem class using the **"withProblemClass"** method.
  - The algorithm that will be used for optimizing the problem by using its name as an argument for the **"withAlgorithm"** method.
  - The number of evaluation loops required for getting the optimum solution for the problem using the **"withmaxevaluations"** method.

As mentioned earlier with the outline for the (ECSDT) algorithm, the process starts with the initialisation of a pool of solutions, and then the "executer" class makes a connection between the initialised pool of solutions, the Problem class and the algorithm to optimize the solutions and return the best one among them. The pseudo code for the top-level of the new algorithm using the MOEA framework is depicted in Figure 4.10 where the outer for loop creates the initial pool of p solutions and for each of these the inner loop creates the ellipses. Once the population and problem are defined, the optimizer is called (Call Optimization) and then the tree constructed (Call Tree maker).

The top-level for the new algorithm using MOEA:

Let numberOfEllipses = The appropriate number of ellipses for the optimization process
Initialise Pool (populationsize)

```
{
  for (int p = 0; p < populationSize; p ++)
    {
      Solution solution[p] = new Solution (numberOfEllipses)
        {
          for (int ell = 0; ell < numberOfEllipses; ell ++)
            {
              solution.setVariables [ell]= [(x_i, y_i, rx_i, ry_i, alpha_i) ]
            }
        }
    }
}
```

Best solution = **Call Optimization** (Dataset, Problem Class, Optimisation Algorithm
, maximum evaluation, Objective functions, Initialisation of the Pool);
Tree = **CALL Tree-maker** (**Best solution**)

Figure 4.10: The pseudo code of the top-level for the new algorithm

The general pseudocode for optimising a particular solution is shown in Figure 4.11.

The Call Optimisation Method using the "executer" class in MOEA:

```
Let Objective-Function-1 = maximise classification accuracy;
Let Objective-Function-2 = minimise misclassification costs;
For each solution[p] Do
  {
     Let Data[i] be set of data contained in Ellipse[i]
     If Ellipse[i] is a Child of Ellipse[j], Then
        {
          Remove Data[i] from Data[j];
        }
     Else
        {
          If Data[i] is inside an overlapped area between Ellipse[i] and Ellipse[j], Then
            {
              Remove Data[i] from the farthest ellipse;
            }
     Set Predicted-Class Data[i] = Majority-Class of Ellipse[i];
     Calculate Accuracy & Cost for solution[p];
     If solution[p] better that Best-Solution, Then
       {
         Best-Solution = solution[p];
       }
  } End Do;
Return (Best-Solution);
```

Figure 4.11: The pseudo code of the optimisation process

ECSDT calculates the value of the accuracy objective function by subtracting the rate of unclassified examples from the accuracy rate. The accuracy rate and the unclassified rate are calculated using the following equations.

$$\text{Accuracy Rate} = \frac{\sum_{i=1}^{m} TC_i}{N} \quad 4.7$$

$$\text{Unclassified Rate} = \frac{\text{No.Unclassified examples}}{N} \quad 4.8$$

Where, TC_i is the total number of correctly classified examples belonging to the class C_i and N is the total number of examples.

Using the two previous Eq 4.7 and 4.8 we can derive the equation for calculating the value of the accuracy-based objective function as follows:

$$\text{Acc\_Obj\_Function} = \text{Accuracy Rate} - \text{Unclassified Rate} \quad 4.9$$

To calculate the total cost of the classification errors, ECSDT uses a similar formula as shown in equation 2.13 :

$$\text{Total Cost} = \sum_{i=1}^{n} \text{Conf}(i \mid j) * \text{Cost}\,(i, j)$$

Where, Conf (i | j) is the number of examples from class i that have been misclassified as class j, and Cost (i | j) is the cost of misclassifying an example of class i as class j.

4.5 Illustrative Examples of How (ECSDT) Works:

To illustrate the way (ECSDT) calculates accuracy and misclassification costs when building the classifier; let us consider the hypothetical multiclass classification problem shown in Figure 4.12 that depicts a 3-class classification problem that includes a total of 60 examples classified into 3 classes (C1 = triangle = 19, C2 = circle = 30, and C3 = rectangle = 11). Figure 4.13 and Figure 4.15 present two examples of different possible solutions for this problem in which four ellipses (A, B, C and D) are placed for each solution. Based on the steps listed in the pseudo code shown in Figure 4.11, the calculation of accuracy and cost depends mainly on the way of allocating Data[i] to the ellipses. Based on that, when considering the proposed solution example-1 depicted in Figure 4.13, then the ECSDT assigns each example to the appropriate ellipse as shown in Figure 4.14.

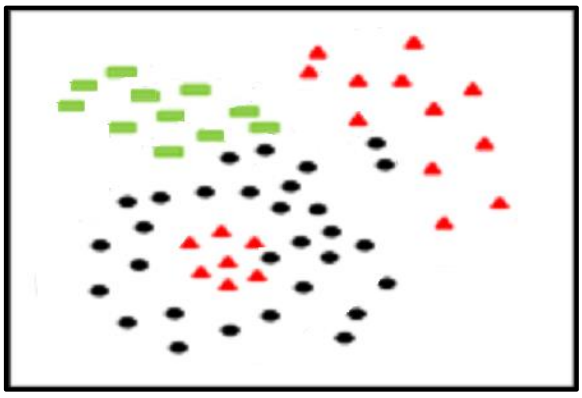

Figure 4.12: Hypothetical multiclass classification problem

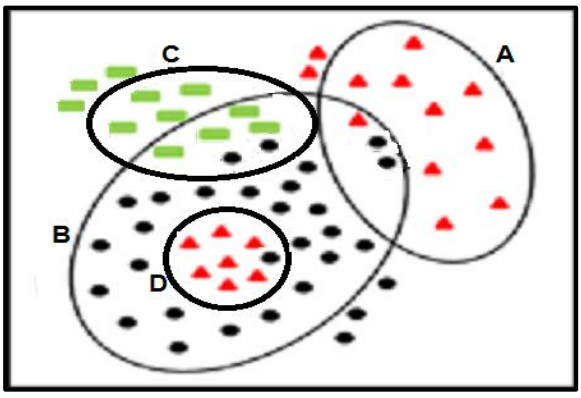

Figure 4.13: Elliptical classification example-1

Figure 4.14 contains 6 different sub-figures 4.14.1, 4.14 .2, 4.14.3, 4.14 .4, 5 and 4.14.6 that show how ECSDT assigns each individual example to only one particular ellipse. Sub-figures 4.14.1, 4.14.2, 4.14.3 and 4.14.4 show examples that fall into only one ellipse, therefore the ECSDT will assign each example to the ellipse that contains it. Sub-figure 4.14.5 shows the

examples that lie in the intersection between ellipses; in this case the ECSDT assigns each example to the ellipse with the nearest center, as we see that the examples located in the intersection between ellipse A and ellipse B were all assigned to ellipse A because the distance between each example and the center of ellipse A is less than the distance between them and the center of ellipse B. The same scenario is repeated to the examples located in the intersection between ellipse B and ellipse C where all examples are assigned to ellipse C because they are closer to ellipse C than ellipse B. Sub-figure 4.14.6 shows the examples that have not been contained in any of the ellipses (Out of all), in such a case, ECSDT does not assign these examples to any ellipse, but instead of that, ECSDT calculates the percentage of these examples over the total number of examples and then imposes this percentage as a penalty on the accuracy rate that decreases the accuracy rate that to stimulate the algorithm to reduce the number of examples that have not been contained in any of the ellipses (Out of all).

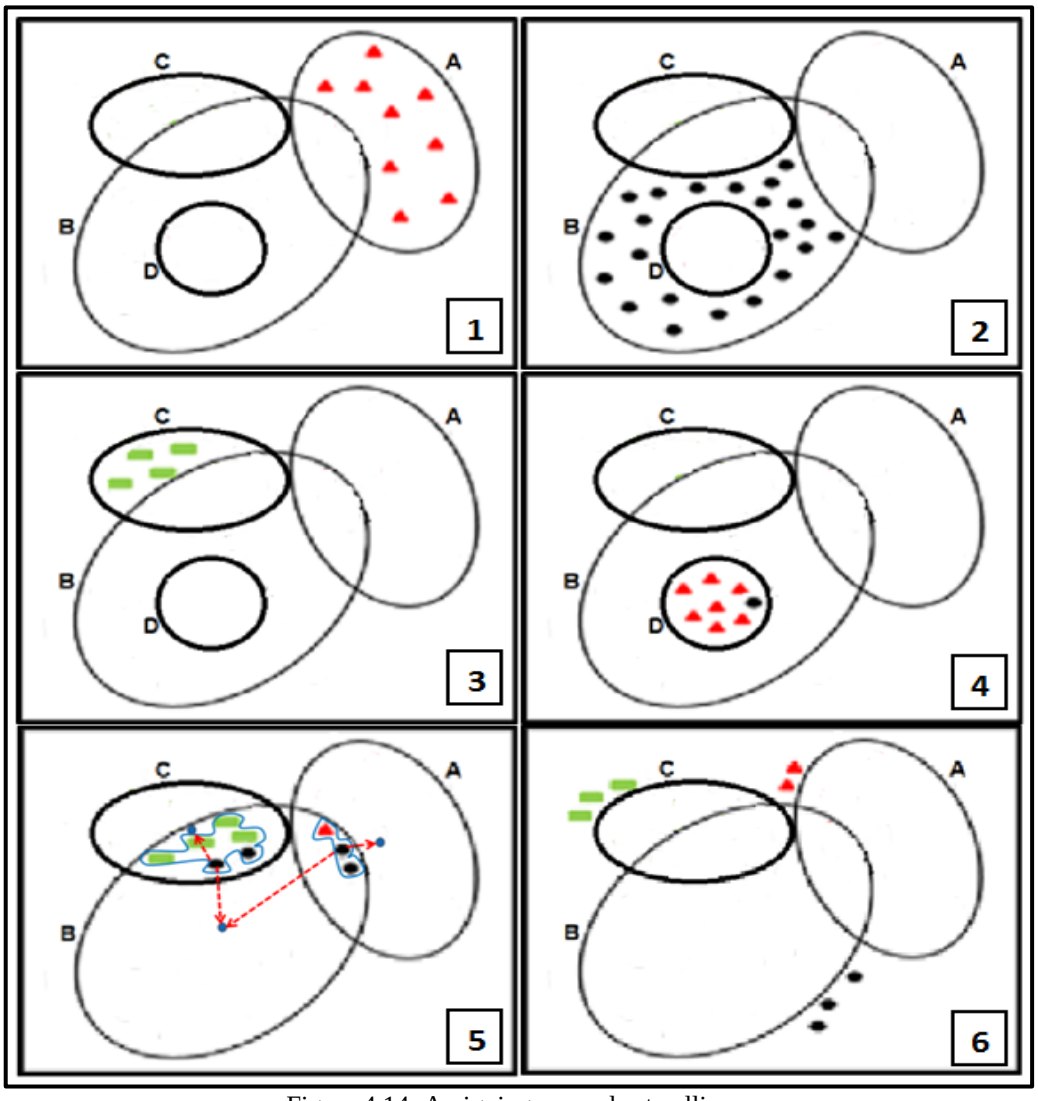

Figure 4.14: Assigning examples to ellipses

From Figure 4.14, the confusion matrix shown in table 4.1 can be observed.

Confusion matrix		Predicted Class			
		C1	C2	C3	Out
Actual Class	C1	17	0	0	2
	C2	3	22	2	3
	C3	0	0	8	3

Table 4.1: The confusion matrix obtained when applying ECSDT for example-1

Using Eq 4.7, Eq 4.8 and Eq 4.9, then the following results can be calculated for example-1:

Accuracy Rate = $(TC_1 + TC_2 + TC_3) / N$
$= (17 + 22 + 8) / 60 \simeq 78.33\ \%$
Unclassified Rate = $(2 + 3 + 3) / 60 \simeq 13.33\ \%$
Acc_Obj_Function = $78.33 - 13.33 = 65\ \%$

As mentioned before, ECSDT keeps trying recursively to improve the overall performance of the classifier by maximising the accuracy and minimising the unclassified rate. Figure 4.15 below depicts another possible solution example-2 for the same problem shown in Figure 4.12 in which the performance is improved over the previous solution shown with example-1 in Figure 4.13.

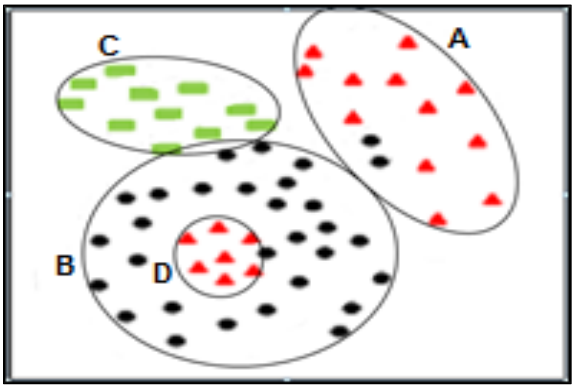

Figure 4.15: Elliptical classification example-2

Confusion matrix		Predicted Class			
		C1	C2	C3	Out
Actual Class	C1	19	0	0	0
	C2	2	28	0	0
	C3	0	0	11	0

Table 4.2: The confusion matrix obtained for example-2

From Figure 4.15, the confusion matrix shown in Table 4.2 can be observed and the following results can be calculated for example-2.

Accuracy Rate = (19 + 28 +11) / 60 ≃ 96.67 %
Unclassified Rate = 0.0 %
Acc_Obj_Function = 96.67 - (0.0) = 96.67 %

From the previous examples and explanations, it can be observed that the perfect optimal solution for any classification problem using ECSDT can be achieved when the Accuracy Rate

= 100% and the Unclassified Rate = 0.0 %, then the value of the accuracy fitness function will be 100%.

To highlight the cost calculation methodology adopted by the ECSDT algorithm, let us again consider both previous examples (example-1 and example-2) along with their associated confusion matrixes are shown in Table 4.1 and Table 4.2 respectively. As mentioned before, to calculate the misclassification costs, a cost matrix should be predefined an expert. Suppose that the cost matrix for both examples was suggested by an expert as given in Table 4.3.

Cost matrix		Predicted Class			
		C1	C2	C3	Out
Actual Class	C1	0.0	5.0	5.0	5.0
	C2	20.0	0.0	20.0	20.0
	C3	10.0	10.0	0.0	10.0

Table 4.3: The cost matrix of (example-1 and example-2)

By using Eq 4.10, then the total misclassification costs associated with example-1 and example-2 can be calculated as follows:

Total cost for (example-1)

= [(17* 0.0) + (0*5.0) + (0*5.0) + (2*5.0)] + [(3*20.0) + (22*0.0) + (2*20.0) + (3*20.0)] + [(0*10.0) + (0*10.0) + (8*0.0) + (3*10.0)]

= [10.0] + [160.0] + [30.0] = 200.0

Average Cost per example (example-1) = 200 / 60 ≃ 3.33

Total cost for (example-2) = (2*20.0) = 40.0

Average Cost per example (example-2) = 40 / 60 ≃ 0.66

Chapter 5 : EMPIRICAL EVALUATION OF THE NEW ALGORITHM

Chapter 4 presented the development of the new algorithm ECSDT and explained how it was implemented using MOEA.

This chapter carries out an empirical evaluation of ECSDT relative to other algorithms. Section 5.1 presents a summary of the data sets used which have been obtained mainly from the UCI Machine Learning Repository (Lichman 2013). As outlined in the literature review chapter, there are many different algorithms for decision tree induction and Section 5.2, therefore, outlines a selection that was used for comparison. Section 5.3 presents and analyses the empirical results from the different algorithms when a 10-fold cross validation methodology is adopted.

5.1 Datasets

For the purposes of this research, common data mining datasets from the UCI Machine Learning Repository (Lichman 2013) have been used in building the classifier and testing its classification accuracy.

Fourteen datasets from the UCI Machine Learning Repository (Lichman 2013) named (Iris, Seeds, Glass, Hepatitis, Bupa, Heart, Diabetes, Haberman, Ecoli, Hayes, Tae, Thyroid, WDBC and WPBC) were used in this research. These datasets are selected to cover several aspects such as the diversity in the areas from which they were derived, the diversity in the number of classes, the diversity in the number of features contained in each dataset, and the diversity in the number of examples contained in each dataset.

The main characteristics of the datasets used in this research are summarized in Table 5.1 and more details about these datasets are provided in Appendix A.

Dataset Name	Area	No. Classes	No. Features	No. Examples
Bupa-Liver Disorders	Medical	2	7	345
Hepatitis	Medical	2	20	155
Statlog Heart Disease	Medical	2	14	270
Haberman's Survival	Medical	2	4	306
Diabetes	Medical	2	20	768
Breast Cancer Wisconsin (Diagnostic) WDBC	Medical	2	32	569
Breast Cancer Wisconsin (Prognostic) WPBC	Medical	2	34	198
IRIS	Medical	3	4	150
Hayes-Roth	Social	3	5	160
Seeds	Medical	3	7	210
Teaching Assistant Evaluation Tae	Education	3	5	151
Thyroid Disease	Medical	3	21	216
Glass Identification	Physical	7	10	214
Ecoli	Medical	8	8	336

Table 5.1: Datasets characteristics

The algorithms were evaluated based on the classification accuracy rate, total misclassification cost and the size of the induced decision trees. The methodology adopted for the empirical evaluation was 10-fold cross-validation which has been widely used by many other studies (Anguita et al. 2012).

In order to get reliable results, the datasets should be filtered and cleaned as much as possible from different types of noises which include, for instance, data entry mistakes, missing values, data instability and other factors that may affect the results (Zhang et al. 2003).

For both stages of developing and evaluating the new algorithm ECSDT, the datasets were manipulated and prepared in a way that works with the new algorithm without problems. That is, the data was prepared to take account of the following points:

1. The algorithm works only with numerical values so that if the dataset contained nominal attributes, they were converted to a numerical format including the nominal class labels which were converted to numerical values as well. The conversion was done by modifying the data in Excel by giving all the similar nominal attribute values a certain numerical value. For example, with IRIS dataset, the 3 types of IRIS flowers Setosa, Versicolour and Virginica were converted to 3 numeric values 1, 2 and 3 respectively. Another example of this conversion is what happened with the Thyroid disease dataset in which the values of some nominal features has been changed to

numeric values such as changing the values of the feature "sex" from M=male and F=female to 1 and 2 respectively, also changing the values of the features that have values of T=true and F=false to 1 and 2 respectively.

2. The algorithm does not deal with datasets with missing values, so only datasets that do not have missing values were used.
3. When using (.csv) dataset files, the dataset heading, which labels the columns of the attributes must be on the first row of the dataset.
4. The column that holds the class labels should be in the last column of the dataset table.

5.2 Comparative Algorithms

The evaluation was made by comparing the results obtained when applying ECSDT to each of the datasets mentioned above along with the results obtained when applying four well-known algorithms to the same datasets considering the same conditions and circumstances. The algorithms selected for assessing and comparing the accuracy of classification are: the J48 algorithm, NBTree algorithm, BFTree, ADTree, LADTree and REPTree; whereas for comparing the cost sensitivity of the ECSDT, two meta-learning algorithms named CostSensitiveClassifier and MetaCost are used. All of these algorithms are provided in the WEKA system which is a collection of a diversity of machine learning algorithms and data mining preprocessing tools (Frank et al. 2016). The algorithms selected for comparison are summarized below:

- **J48:** Is a decision tree learner in Weka (Witten et al. 2016) that implements the standard C4.5 algorithm (Hormann 1962) which is an extension of the ID3 algorithm (Quinlan 1986). More details about ID3 and C4.5 are given in Chapter 2. J48 has more features over the ID3 and C4.5 such as taking into account missing values, pruning the tree, the ability to deal with continuous values, etc.(Kaur & Chhabra 2014).

- **NBTree:** The NBTree algorithm (Kohavi 1996) combines features from both decision trees and Naive Bayes. The decision-making process is based on the structure of the tree that is constructed recursively. However, leaf-nodes are set using Naive-Bayes techniques. For features with continuous values, the algorithm uses a particular threshold to reduce the value of the entropy measure. The nodes are evaluated by

calculating the average of the five-fold cross-validation using Naïve-Bayes at the node. The NBTree algorithm attempts to test if the calculated accuracy using Naive-Bayes at every leaf-node is greater than the result obtained by applying an individual Naive-Bayes classifier at the present node.

- **BFTree:** (Best-First Tree), with BFTree, the best node to split on is the one that has a maximum reduction of the impurity value among others. The produced tree is the same as the one produced by standard algorithms as C4.5 and CART except for the way the decision tree is constructed. BFTree builds binary trees, that is each decision node will have exactly two outgoing branches (Shi 2007).

- **ADTree:** (Alternating Decision Tree), ADTree (Freund & Mason 1999) is a classification method that integrates decision trees with boosting into a collection of classification rules that are small in size and easy to understand. ADTree is mainly aimed at solving binary classification problems and then expanded to include multiclass problems.

- **LADTree:** (Logical Analysis of Data Tree), LADTree (Holmes et al. 2002) is a classification method that was originally introduced for solving binary classification problems. LADTrees are constructed using logical expressions that can differentiate between positive and negative examples. Building LADTrees usually include the creation of a large set of models and then selecting a subset from the models that satisfy particular predefined requirements (S. R. Kalmegh 2015).

- **REPTree:** (Reduced Error Pruning Tree), RepTree (J.R. Quinlan 1987) is a rapid decision tree induction method that is founded on the base of calculating the information-gain with entropy and reducing the errors that emerge from inconsistency. RepTree produces multiple trees in various repetitions that employ the regression tree logic for constructing the trees. Then the best tree is chosen from the others as the representative tree (S. Kalmegh 2015; Devasena & Hyderabad 2015).

- **CostSensitiveClassifier:** Is a meta-classifier that utilise two cost-sensitive methods to construct the model. The first method attempts to reweight the training examples based

on the total cost allocated to each class, whereas, the second method endeavour to make predictions to the class with the lower expected misclassification cost rather than making the prediction to the majority class (Witten et al. 2016).

- **MetaCost:** This meta-classifier is one of the most well-known methods for converting base classifiers to cost-sensitive (Domingos 1999). More details about MetaCost are given in Chapter 3.

5.3 Empirical Evaluation

The evaluation phase of the ECSDT includes three different aspects. The first aspect considers accuracy only as the main objective where the ideal solution is the solution that gives the highest accuracy regardless of the cost, whereas, for the second aspect, misclassification costs are the main target for the classifier, and for the third aspect, the target is to explore the impact of misclassification cost on the accuracy that by subtracting the average cost from the accuracy rate which ensures that the higher the accuracy and the lower the cost, the better the objective function result will be.

All experiments were carried out using 10 fold cross-validations in which the dataset is randomly divided into 10 folds of equal sizes. Of the 10 folds, 1 fold is held for testing the classifier and the remaining folds are used for training the classifier. For experiments in which the cost is taken into account, trials were conducted on different sets of cost ratios in order to assess the cost sensitivity of the new algorithm. More details about the cost ratios used in this research are presented later in Section 5.3.2.

As mentioned before, this algorithm follows a new method which uses ellipses instead of straight lines for data separation and classification. One of the challenges facing this algorithm is how to determine the appropriate number of ellipses that should be used to build the decision tree for each classification problem. For this reason, all experiments were repeated five times with a different number of ellipses to determine the number of ellipses that gives the best results compared to the other alternatives. The number of ellipses used varies from one dataset to another according to the nature of the dataset and according to the number of classes. The methodology used to determine the number of ellipses was based on the number

of classes. The number of ellipses used for each dataset usually starts with a number equal to the number of classes and then this number is gradually increased with a particular increment. For example, with 2-class datasets, the tested numbers of ellipses are 2, 4, 6, 8 and 10 ellipses and with 3-class databases, the tested numbers of ellipses are 3, 6, 9, 12 and 15 ellipses. For the Glass dataset which is a 7-class problem and for the Ecoli dataset which is an 8-class problem, it is obvious that following the same methodology will lead to the production of very large decision trees, so the numbers of ellipses used with both datasets was limited to 6, 8, 10, 12 and 14 ellipses. After determining the optimal number of ellipses for each dataset the results obtained from applying the optimal number of ellipses in each optimization method are compared with the results obtained by applying the other comparative algorithms on the same datasets with the same conditions and circumstances.

The experiments were performed with both optimization methods namely: OMOPSO and NSGA-II that are available in MOEA framework. These optimization methods are controlled by several parameters, ECSDT uses these methods with only the default values. Some examples of these properties are populationSize, mutationProbability, maxEvaluations and archiveSize.

As an example, but not exclusively, Figure 5.1 below depicts the general methodology used in the evaluation process for each dataset.

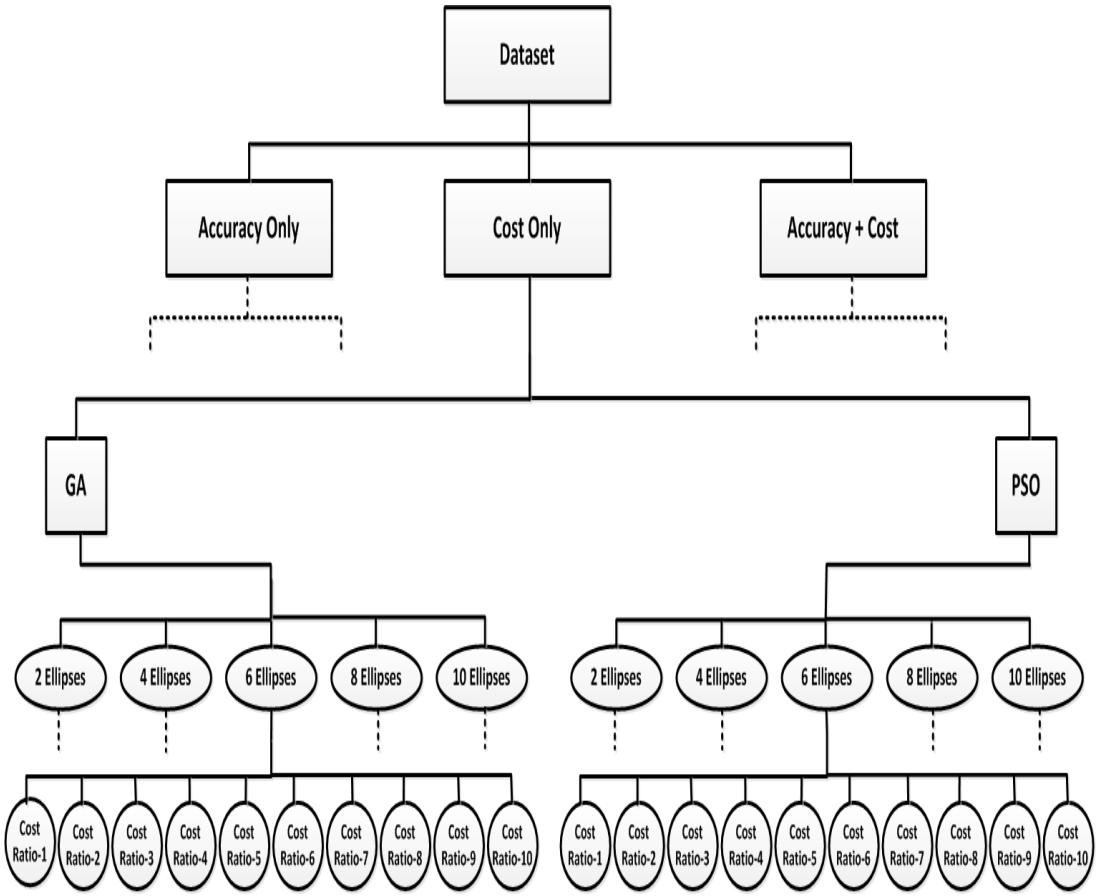

Figure 5.1: General methodology for the evaluation process

The following sub-sections highlight and discuss the evaluation processes and show the comparison results obtained in each of the three evaluation aspects mentioned above, namely: accuracy only, cost only and both accuracy and cost as one objective function. Each sub-section presents an evaluation for the results of all datasets, and then for further clarification, detailed explanations of the results related to three datasets namely: Bupa, IRIS and Ecoli are given in separate sub-sections. These 3 datasets were selected to represent a sample of datasets labelled with different numbers of classes, where Bupa is a 2-class dataset, IRIS is a 3-class dataset and Ecoli is an 8-class dataset.

5.3.1 Empirical Comparison Based on Accuracy

In the first aspect where accuracy is considered as the only objective function for building, testing and evaluating the classifier; the comparative algorithms used to evaluate the performance of the ECSDT algorithm are: J48, NBTree, BFTree, ADTree, LADTree and REPTree.

For accuracy-based experiments, the primary objective is to compare classification accuracy only; so, it has been assumed that all classification errors similarly have the same cost which has been set to only 1 unit of cost during the stage of building the classifier. For each dataset and for each of the optimization methods (GA and PSO) the best accuracy results along with the associated decision tree sizes are compared with those obtained by the other algorithms. The following sub-sections explain and discuss the results obtained from the accuracy-based evaluation. Section 5.3.1.1 gives a general discussion for the accuracy-based results for all datasets in general and as mentioned above, sub-sections 5.3.1.2, 5.3.1.3 and 5.3.1.4 are devoted to a detailed explanation of the results obtained for Bupa, IRIS and Ecoli datasets respectively. All tables and results related to this aspect are provided in Appendix B.

5.3.1.1 Discussion for Accuracy-based results

As mentioned previously, one of the most important steps in the evaluation of the new algorithm ECSDT is to determine the appropriate number of ellipses for each dataset and then to compare the results obtained with those from other algorithms. Table 5.2 shows the classification accuracy obtained when ECDT is applied to each dataset when using a different number of ellipses and when both optimization methods (GAs and PSO) are used.

It is reasonable to think that increasing the number of ellipses will always lead to increased accuracy rates, but we note from Table 5.2 that this is not always true as in some cases although the number of ellipses is increased, the accuracy results remain in the same range, or even reduces slightly. The other thing that can be seen from the table is that the general performance when using the GA optimization method (NSGA-II algorithm) is better than the general performance of using the PSO optimization method (OMOPSO algorithm), as we can observe that applying the NSGA-II algorithm has better results than the OMOPSO algorithm in 44 out of the 70 trials, while applying the OMOPSO algorithm achieved better results than the NSGA-II algorithm in only 20 trials and the performance of both algorithms was equal in only 6 trials.

Table 5.3 presents the results achieved by the comparative algorithms along with the results from ECSDT when it is used with the ideal number of ellipses determined at the previous stage.

Figure 5.2 and Figure 5.3 graphically illustrate the accuracies and decision tree sizes that are shown in Table 5.3. Figure 5.2 shows that the ECSDT algorithm achieved better results for the majority of datasets especially when the NSGA-II algorithm is used as the optimization method where it recorded better accuracy on 10 out of the 14 datasets. The datasets with which the ECSDT algorithm recorded the best accuracy results are: Bupa, Hepatitis, Hart, Huberman, Diabetes, WPBC, IRIS, Seeds, Thyroid, and Ecoli. In addition to the promising accuracy results achieved by the ECSDT algorithm, it was able to produce smaller decision trees than the J48 algorithm on 6 out of the 10 datasets with which the ECSDT algorithm produced the best accuracy results. The datasets with which the ECSDT algorithm recorded the best accuracy with the smallest decision trees are: Bupa, Hart, Diabetes, WPBC, IRIS, Thyroid, and Ecoli. For example, with Bupa, the ECSDT algorithm obtained an accuracy rate of 72.35% with a decision tree size of 17 nodes, while the best accuracy rate achieved by other algorithms was produced by the J48 algorithms with an accuracy value of 68.70% and a decision tree size of 51 nodes. Also, with the Diabetes dataset, the ECSDT algorithm obtained an accuracy rate of 78.28% with a decision tree size of 13 nodes, while the best accuracy rate achieved by other algorithms was produced by the REPTree algorithms with an accuracy value of 75.26% and a decision tree size of 49 nodes.

Dataset	Optimizer	No. Ell	Accuracy	SE	No. Ell	Accuracy	SE	No. Ell	Accuracy	SE	No. Ell	Accuracy	SE	No. Ell	Accuracy	SE
Bupa	OMOPSO	2	49.41	0.51	4	59.71	0.87	6	58.82	0.51	8	65.85	0.43	10	68.52	0.77
	NSGA-II	2	53.53	0.43	4	59.41	0.76	6	60.59	0.62	8	66.47	0.39	10	72.35	0.46
Hepatitis	OMOPSO	2	81.42	0.96	4	82.86	1.12	6	85.71	1.16	8	84.29	1.71	10	84.29	1.05
	NSGA-II	2	80.00	1.90	4	82.86	1.47	6	85.71	0.95	8	88.57	0.60	10	87.14	1.05
Heart	OMOPSO	2	68.15	1.33	4	73.7	0.82	6	77.40	0.59	8	73.70	0.40	10	84.81	0.44
	NSGA-II	2	67.40	0.57	4	71.48	0.67	6	80.37	0.58	8	77.03	0.51	10	83.33	0.80
Haberman	OMOPSO	2	73.33	0.90	4	76.00	0.56	6	76.66	0.38	8	78.00	0.74	10	78.66	0.52
	NSGA-II	2	74.66	0.89	4	75.00	0.65	6	78.00	0.50	8	79.66	0.57	10	80.00	0.41
Diabetes	OMOPSO	2	74.73	0.57	4	75.52	0.35	6	76.97	0.41	8	78.81	0.36	10	75.13	0.46
	NSGA-II	2	73.42	0.52	4	76.44	0.30	6	78.28	0.28	8	77.89	0.43	10	76.84	0.48
WDBC	OMOPSO	2	85.88	0.99	4	92.14	0.24	6	85.35	1.22	8	93.20	0.16	10	87.32	0.54
	NSGA-II	2	85.71	1.07	4	88.92	0.58	6	90.88	0.46	8	93.34	0.23	10	89.63	0.33
WPBC	OMOPSO	2	76.84	0.44	4	77.35	0.89	6	78.42	1.17	8	76.84	0.50	10	77.90	0.88
	NSGA-II	2	76.84	0.75	4	78.42	0.97	6	75.26	1.02	8	76.31	0.93	10	73.16	0.76
IRIS	OMOPSO	3	95.33	0.30	4	98.66	0.28	5	97.33	0.34	6	96.66	0.35	7	95.33	0.32
	NSGA-II	3	94.66	0.52	4	98.66	0.28	5	96.66	0.46	6	96.00	0.34	7	95.33	0.44
Hays	OMOPSO	3	57.5	1.43	6	61.87	0.85	9	65.62	1.03	12	68.12	0.85	15	62.50	0.77
	NSGA-II	3	60.62	0.72	6	68.12	1.08	9	68.12	0.95	12	71.25	0.94	15	63.12	1.33
Seeds	OMOPSO	3	83.32	1.27	6	87.62	0.78	9	87.14	0.87	12	91.90	0.50	15	86.66	0.83
	NSGA-II	3	85.71	0.86	6	89.52	0.66	9	87.14	0.71	12	93.33	0.41	15	89.04	0.59
Tae	OMOPSO	3	44.00	1.26	6	51.33	1.17	9	51.33	1.04	12	54.00	1.23	15	50.00	1.30
	NSGA-II	3	43.33	1.22	6	47.33	1.06	9	49.32	0.95	12	50.00	0.90	15	55.32	0.83
Thyroid	OMOPSO	3	85.70	1.22	6	91.43	0.86	9	87.14	0.45	12	87.61	0.68	15	89.04	0.45
	NSGA-II	3	88.56	0.93	6	94.28	0.49	9	92.37	0.51	12	89.52	0.54	15	91.90	0.55
Glass	OMOPSO	6	79.04	1.63	8	89.04	0.45	10	90.47	0.40	12	92.37	0.58	14	93.32	0.81
	NSGA-II	6	81.89	0.94	8	90.47	0.38	10	93.32	0.33	12	94.28	0.43	14	95.23	0.38
Ecoli	OMOPSO	6	72.72	0.85	8	76.35	0.68	10	76.35	0.46	12	82.78	0.20	14	74.50	0.57
	NSGA-II	6	73.32	0.51	8	77.26	0.55	10	76.96	0.21	12	83.33	0.54	14	76.35	0.46

Table 5.2: Accuracy-based results for all datasets using different number of ellipses when ECSDT is applied

Dataset	J48			NBTree			BFTree			ADTree		
	Size	Acc	SE	Size	Acc	SE	Size	Acc	SE	Size	Acc	SE
Bupa	51	68.70	0.5	11	66.38	0.48	19	64.92	0.50	31	59.71	0.47
Hepatitis	11	85.89	0.172	5	84.61	0.35	5	83.33	0.38	31	87.17	0.31
Heart	39	77.40	0.44	17	80.74	0.38	37	77.40	0.44	31	79.25	0.38
Haberman	5	71.89	0.43	3	72.54	0.42	5	73.52	0.44	31	71.56	0.43
Diabetes	39	73.8	0.44	1	73.56	0.42	5	73.56	0.44	31	72.91	0.41
WDBC	25	92.97	0.26	23	92.79	0.25	17	92.97	0.25	31	94.72	0.18
WPBC	21	73.73	0.49	11	71.21	0.47	1	75.75	0.43	31	73.23	0.43
IRIS	9	96.00	0.15	9	94.66	0.17	11	94.66	0.17			
Hays	23	83.12	0.27	13	64.37	0.38	29	81.25	0.27			
Seeds	15	91.90	0.23	7	90.95	0.22	19	93.33	0.21			
Tae	67	59.60	0.46	7	58.27	0.43	28	57.61	0.45			
Thyroid	17	92.09	0.21	7	93.02	0.20	17	92.09	0.23			
Glass	11	96.72	0.09	7	93.45	0.11	11	98.13	0.07			
Ecoli	41	79.76	0.20	13	80.05	0.20	57	78.86	0.21			
Overall Average	26.71	81.68	0.31	9.57	79.76	0.32	18.64	81.24	0.32	31	76.94	0.37

Dataset	LADTree			REPTree			ECSDT+OMOPSO			ECSDT+NSGA-II		
	Size	Acc	SE	Size	Acc	SE	Size	Acc	SE	Size	Acc	SE
Bupa	31	65.50	0.46	23	64.05	0.49	21	68.52	0.77	21	72.35	0.46
Hepatitis	31	80.76	0.40	11	87.17	0.31	13	85.70	1.16	17	88.57	0.60
Heart	31	80.00	0.36	7	76.66	0.42	21	84.81	0.44	21	83.33	0.80
Haberman	31	73.52	0.44	5	71.24	0.45	21	78.66	0.52	21	80.00	0.41
Diabetes	31	74.08	0.42	49	75.26	0.42	17	78.81	0.36	13	78.28	0.28
WDBC	31	95.60	0.18	9	92.44	0.25	17	93.20	0.16	17	93.34	0.23
WPBC	31	75.25	0.44	7	72.22	0.45	13	78.42	1.17	9	78.42	0.97
IRIS	31	94.00	0.19	5	94.00	0.19	9	98.66	0.56	9	98.66	0.46
Hays	31	82.50	0.26	25	83.75	0.27	25	68.12	0.85	25	71.25	0.94
Seeds	31	91.90	0.20	5	90.00	0.24	25	91.90	0.59	25	93.33	0.46
Tae	31	59.60	0.43	29	53.64	0.46	25	54.00	1.23	31	55.32	0.90
Thyroid	31	93.95	0.18	7	92.09	0.22	13	91.43	0.86	13	94.28	0.49
Glass	28	98.13	0.07	11	98.59	0.06	29	93.32	0.40	29	95.23	0.38
Ecoli	31	82.44	0.18	25	76.78	0.21	25	82.72	0.38	25	83.33	0.26
Overall Average	30.79	81.95	0.3	15.57	80.56	0.32	19.57	82.02	0.68	19.71	83.26	0.55

Table 5.3: Accuracy-based Results for all datasets when applying different algorithms

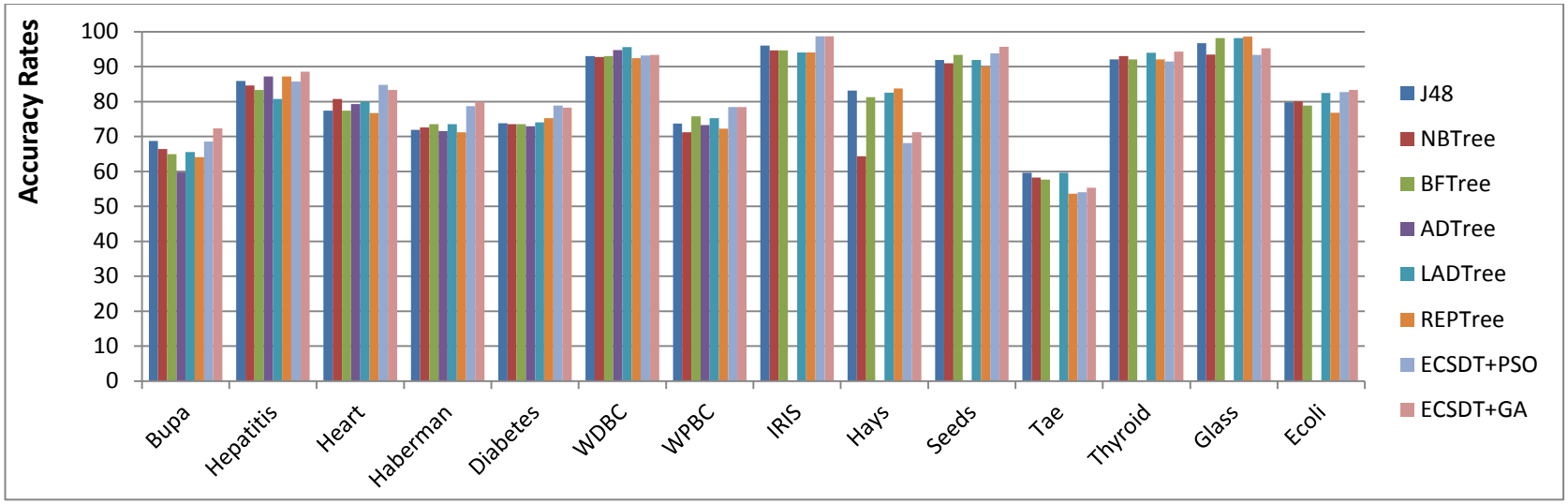

Figure 5.2: Accuracy comparison (Accuracy-based aspect)

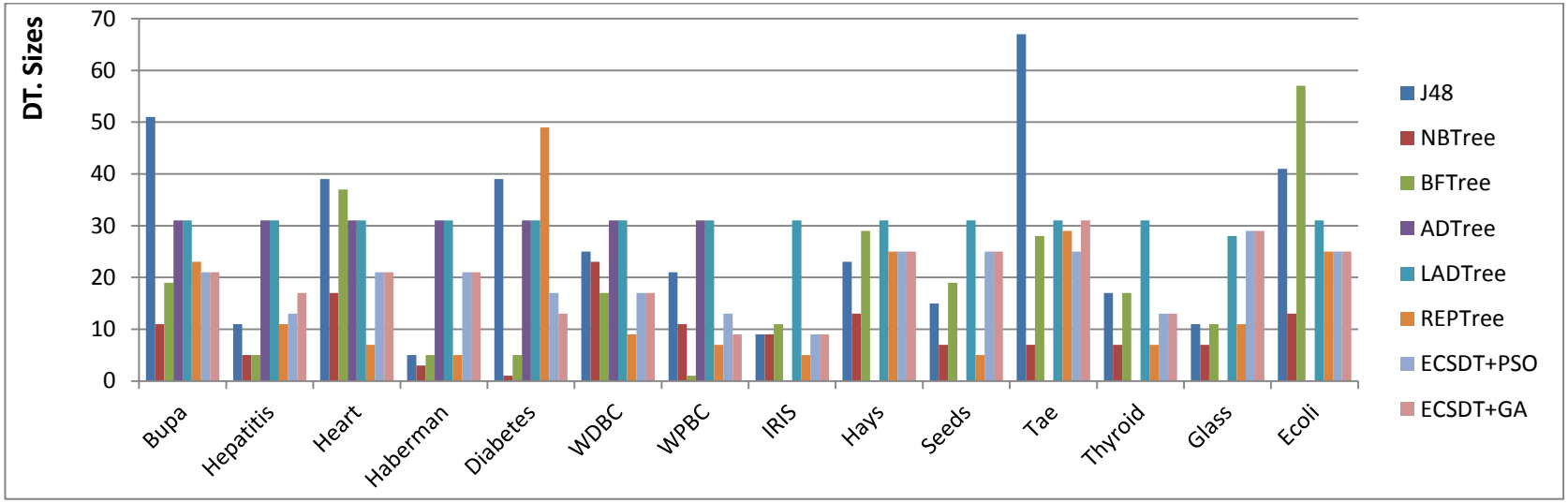

Figure 5.3: DT-size comparison (Accuracy-based aspect)

With the Seeds dataset, the ECSDT algorithm managed to achieve better classification results than the rest of the algorithms which reached a value of 95.71% while the highest accuracy rate obtained by the rest of the algorithms is 93.33% which is achieved by the BFTree algorithm, but getting better accuracy was at the expense of a larger tree: ECSDT resulted in a tree with 25 nodes which exceeds the sizes of most of the trees produced by other algorithms.

The other algorithms performed better than the ECSDT algorithm on only 4 out of 14 datasets, which are: WDBC, Hayes, Tae and Glass.

5.3.1.2 Accuracy-based results for the Bupa dataset

Table 5.4 below presents the results obtained when a different number of ellipses (2, 4, 6, 8, and 10) are used with each of the optimization method (OMOPSO and NSGA-II) along with the results obtained when applying the comparison algorithms.

Algorithm		DT Size	Accuracy	SE
J48		51	68.7	0.50
NBTree		11	66.38	0.48
BFTree		19	64.927	0.50
ADTree		31	59.71	0.47
LADTree		31	65.507	0.46
REPTree		23	64.058	0.49
ECSDT + OMOPSO	**2-ell**	5	49.41	0.51
	4-ell	9	59.71	0.87
	6-ell	13	58.82	0.51
	8-ell	17	65.85	0.43
	10-ell	21	68.52	0.77
ECSDT + NSGA-II	**2-ell**	5	53.53	0.43
	4-ell	9	59.41	0.76
	6-ell	13	60.59	0.62
	8-ell	17	66.47	0.39
	10-ell	21	72.35	0.46

Table 5.4: Accuracy-based results (Bupa)

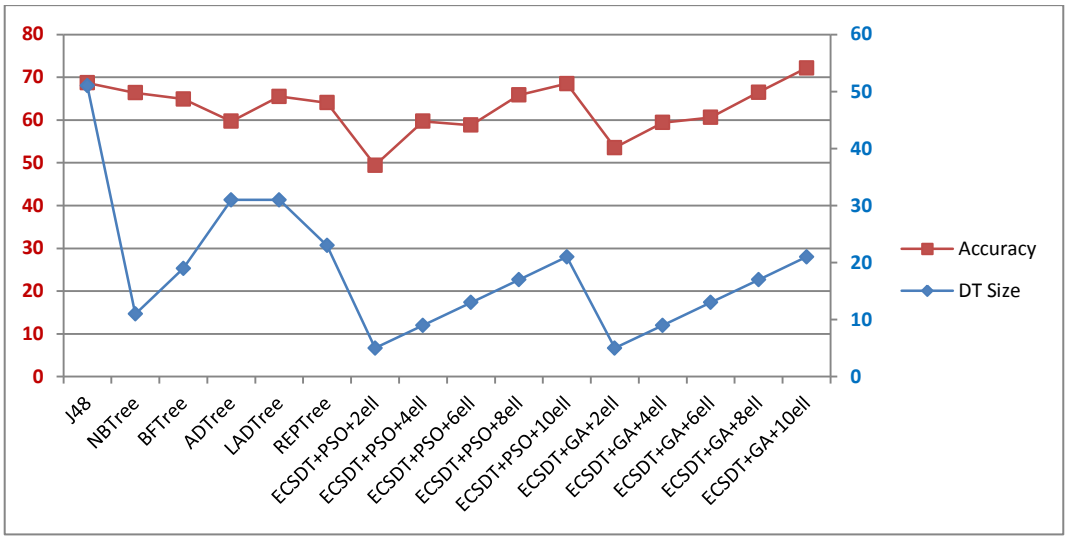

Figure 5.4: Line chart for the Accuracy-based results (Bupa)

From Table 5.4 and Figure 5.4, we can observe that the accuracy obtained with ECSDT using both optimization methods (OMOPSO and NSGA-II) improves by increasing the number of ellipses with the exception of some minor cases that result in a slight decrease. For example, we see that the classification accuracy rate using the OMOPSO optimizer with 4 ellipses reached a value of 59.71%, but after increasing the number of ellipses to 6 ellipses, the value fell a bit to 58.82 %. Intuitively we may think that increasing the number of ellipses for the optimization process should always lead to improved results. However, this may not always be true for two reasons: first, the ellipses used are initially generated randomly with different sizes, locations and angles of rotation, and secondly since increasing the number of ellipses may results in complicated overlapping ellipses, making it difficult for the optimization algorithms to achieve a good classification accuracy. Using 10 ellipses with the utilization of the genetic algorithm NSGA-II as an optimizer, the ECSDT algorithm achieved better accuracy results than all the other algorithms that reached 72.35%. When considering both accuracy rates and decision tree sizes as a double criterion for evaluation, we found that the ECSDT algorithm performs better than the other algorithms (J48, BFTree, ADTree, LADTree, and REPTree). In contrast with (J48, ADTree, LADTree, and REPTree), the ECSDT algorithm was able to achieve better accuracy of 72.35% with a smaller size of tree of only 21 nodes, whereas the best accuracy obtained by the 4 algorithms was 68.7% which was achieved by the J48 algorithm, and the smallest tree size was 23 nodes that achieved by the REPTree algorithm. In contrast with the BFTree algorithm, the ECSDT algorithm was able to achieve better accuracies and smaller trees with both optimization methods. ECSDT was able to achieve an accuracy of 65.85% with the OMOPSO method and an accuracy of 66.47% with

the NSGA-II method and both accuracies achieved with decision trees of size 17 nodes, whereas the BFTree obtained less accuracy of 64.92% with a larger decision tree of size 19 nodes.

5.3.1.3 Accuracy-based results for the IRIS dataset

IRIS is a 3-class dataset and it is one of the most widely used multiclass datasets for evaluating classifiers. This dataset describes some measurements such as petal length, petal width, sepal length and sepal width for 3 types of iris flowers namely: Iris Setosa, Iris Versicolour and Iris Virginica. The goal is to classify a new plant to one of the 3 types of iris plants based on some given measurements of the flower.

Many studies suggest that a small number of boundaries are sufficient for this problem. Hence, for the IRIS data, the proposed five different numbers of ellipses that are tried are (3, 4, 5, 6, and 7). Table 5.5 presents the results obtained for IRIS dataset when using the different number of ellipses with each of the optimization methods along with the results obtained by the comparison algorithms. The ADTree algorithm was excluded from the comparison process since it is intended for only binary classification problems and does not deal with multiclass problems.

Algorithm		DT Size	Accuracy	SE
J48		9	96.00	0.158
NBTree		9	94.66	0.170
BFTree		11	94.66	0.175
ADTree		///	///	///
LADTree		31	94.0	0.193
REPTree		5	94.0	0.193
ECSDT + PSO	**3ell**	7	95.33	0.30
	4ell	9	98.66	0.28
	5ell	11	97.33	0.34
	6ell	13	96.66	0.35
	7ell	15	95.33	0.32
ECSDT + GA	**3ell**	7	94.66	0.52
	4ell	9	98.66	0.28
	5ell	11	96.66	0.46
	6ell	13	96	0.34
	7ell	15	95.33	0.44

Table 5.5: Accuracy-based results (IRIS)

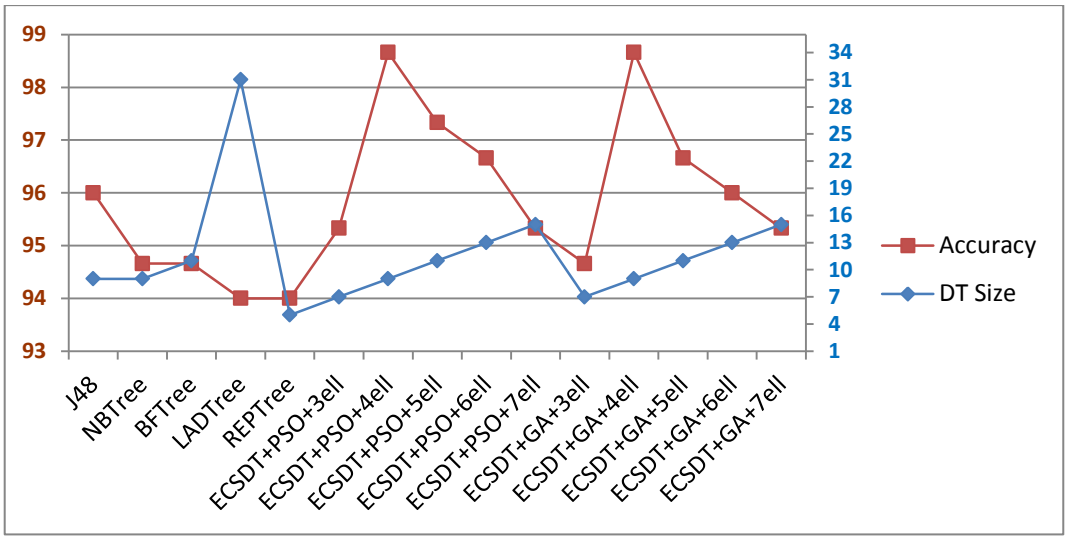

Figure 5.5: Line chart for the Accuracy-based results (IRIS)

From Table 5.5 and Figure 5.5 above, we note that the increase in the number of ellipses does not always lead to improvements in accuracy. The fluctuation in results can be attributed to the same reason that was previously mentioned with Bupa's accuracy-based results which are based primarily on the complex overlapping which could happen between the increased numbers of ellipses, and also having too many ellipses can lead to overtraining in which the produced classifier is too close to the training data.

In general, the new algorithm was able to achieve higher accuracy than the other algorithms with an equal or even smaller decision tree. For the REPTree algorithm we can see that it was able to produce the smallest tree, but at the same time, it could not improve upon other algorithms in terms of accuracy.

5.3.1.4 Accuracy-based results for the Ecoli dataset

The Ecoli dataset is a multiclass classification dataset with 336 instances and 8 classes. The main objective of using this dataset in classification is to make a prediction for the localization area of proteins of the bacteria cell by utilizing some measurements of the cell. Table 5.4 presents the results obtained for the Ecoli dataset when using a different number of ellipses with each of the optimization methods along with the results obtained by the comparison algorithms.

Algorithm		DT Size	Accuracy	SE
J48		41	79.76	0.42
NBTree		13	80.06	0.39
BFTree		29	78.86	0.51
ADTree		N/A	N/A	N/A
LADTree		31	82.44	0.38
REPTree		25	76.79	0.55
ECSDT + PSO	**6ell**	12	72.72	0.37
	8ell	17	76.35	0.48
	10ell	21	76.35	0.44
	12ell	25	82.72	0.55
	14ell	29	74.50	0.52
ECSDT + GA	**6ell**	12	73.32	0.52
	8ell	17	77.26	0.48
	10ell	21	76.96	0.46
	12ell	25	83.33	0.54
	14ell	29	76.35	0.48

Table 5.6: Accuracy-based results (Ecoli)

Table 5.6 presents the results obtained for the Ecoli dataset when implementing the different number of ellipses (6, 8, 10, 12, and 14) with each of the optimization methods along with the results obtained by the comparison algorithms.

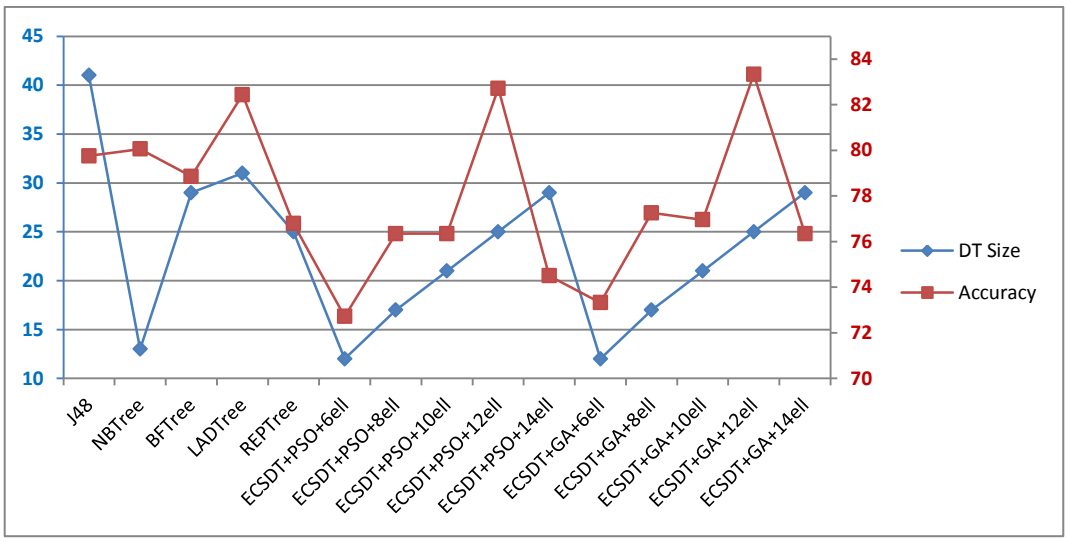

Figure 5.6: Line chart for the Accuracy-based results (Ecoli)

From Table 5.6 and Figure 5.6 above, we note that ECSDT with both optimization methods was able to achieve higher accuracy rates than other algorithms with a decision tree of size 25 nodes which is smaller than the trees produced by other algorithms except for that produced by NBTree which is of size 13 nodes, but with result in less accuracy of 80.06% than that obtained by ECSDT.

When comparing the performance of the two optimization methods, we find that the performance of NSGA-II is slightly better than the performance of OMOPSP with all the numbers of used ellipses.

5.3.2 Empirical Comparison Based on Cost

For the second aspect that considers only cost as the only objective function for building, testing and evaluating the classifier; the performance of ECSDST is compared with that of two meta-classifiers named CostSensitiveClassifier and MetaCost. With these two meta-classifiers, J48 and NBTree were used as the base learners for inducing the decision trees. Like the accuracy-based approach, experiments with a number of ellipses were carried out. Each alternative number of ellipses is examined with a number of cost ratios to assess the cost sensitivity of the ECSDT algorithm. The cost matrices adopted vary from one dataset to another depending on the number of classes, where it is obvious that adopting cost ratios to evaluate the cost sensitivity of the ECSDT algorithm on 2-class problems certainly will not be adequate for evaluating the algorithm on 3-class problems, and the cost ratios used on 3-class problems will not be adequate for 4-class problems and so on. These cost ratios are shown in Table 5.7.

For the cost-based experiments, the primary objective is to compare the misclassification costs. Therefore, for each optimization method (OMOPSO and NSGA-II) only the results related to the number of ellipses that gives the lowest average cost per example for all cost ratios is considered and its results compared with the corresponded results obtained from the other algorithms. The average cost for each alternative number of ellipses is calculated using the following Eq 5.1.

$$\text{Cost\_Average}_{(k)ellipses} = \frac{\sum_{i=1}^{N} \text{Cost}_i}{N} \qquad 5.1$$

Where, k indicates the number of ellipses used, N is the number of cost ratios and Cost_i is the cost obtained when using the i^{th} cost ratio with K ellipses.

And in the same way, the average cost for the results obtained by the comparative algorithms is calculated using the following Eq 5.2.

$$\text{Cost\_Average}_{algorithm(g)} = \frac{\sum_{i=1}^{N} \text{Cost}_i}{N} \qquad 5.2$$

Where N is the number of cost ratios and Cost_i is the cost obtained when using the i^{th} cost ratio with the algorithm (g).

		Cost Ratios									
2 Class Datasets		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
	Class-1	1	1	1	1	1	10	50	100	500	1000
	Class-2	10	50	100	500	1000	1	1	1	1	1

		Cost Ratios								
3 Class Datasets		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
	Class-1	1	100	10	1	10	5	150	250	200
	Class-2	10	1	100	5	1	10	200	150	250
	Class-3	100	10	1	10	5	1	250	200	150

		Cost Ratios					
6 Class Datasets		**1**	**2**	**3**	**4**	**5**	**6**
	Class-1	1	1000	500	100	50	10
	Class-2	10	1	1000	500	100	50
	Class-3	50	10	1	1000	500	100
	Class-4	100	50	10	1	1000	500
	Class-5	500	100	50	10	1	1000
	Class-6	1000	500	100	50	10	1

		Cost Ratios							
8 Class Datasets		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
	Class-1	1	1000	500	250	100	50	10	5
	Class-2	5	1	1000	500	250	100	50	10
	Class-3	10	5	1	1000	500	250	100	50
	Class-4	50	10	5	1	1000	500	250	100
	Class-5	100	50	10	5	1	1000	500	250
	Class-6	250	100	50	10	5	1	1000	500
	Class-7	500	250	100	50	10	5	1	1000
	Class-8	1000	500	250	100	50	10	5	1

Table 5.7: Misclassification cost ratios used in all experiments

The following sub-sections explain and discuss the results obtained from the cost-based evaluation. Section 5.3.2.1 presents the results for all the datasets; Sections 5.3.2.2, 5.3.2.3 and 5.3.2.4 provide some more detailed discussion for three data sets, namely Bupa, IRIS and Ecoli datasets respectively. All tables and results related to this aspect are provided in Appendix C.

5.3.2.1 Discussion for Cost-based results

The methodology adopted to compare the algorithms in terms of cost sensitivity was by calculating the average costs obtained when applying all cost ratios for each number of ellipses and then dividing the results by the total number of examples to get the average cost for each example.

Table 5.8 below depicts the average misclassification cost per example along with the associated accuracy obtained for each of the cost-based experiments for all datasets when using different numbers of ellipses for both optimization methods. From the table we can see that: (i) NSGA-II is able to achieve lower costs than OMOPSO on 8 out of 14 datasets, (ii) OMOPSO is better than NSGA-II on 4 datasets, and (iii) their performance is equal on only 2 datasets.

More specifically, NSGA-II is better than OMOPSO in 47 out of the 70 trials, OMOPSO achieved better results in only 17 trials and the performance of both algorithms was equal in 6 trials. The table also shows that increasing the number of ellipses has a positive effect in reducing costs in about 92.86% of the total number of trials and the best results recorded for the majority of datasets are achieved when using the largest proposed number of ellipses.

To compare the cost-based results of ECSDT with those obtained by the comparative algorithms, the costs obtained using all cost ratios for each algorithm is averaged. For ECSDT, only the results associated with the numbers of ellipses that gives the lowest average cost are selected for comparison.

Table 5.9 presents these results in terms of costs, associated accuracies and decision tree sizes and the comparisons are graphically illustrated in Figures 5.7, 5.8, 5.9 and 5.10. Figure 5.7 and Figure 5.8 presents the cost comparison; Figure 5.9 presents the accuracy comparison, and Figure 5.10 presents a comparison of the size of decision trees.

From Figure 5.7 and Figure 5.8, we can notice that the ECSDT algorithm was able to achieve lower costs on 10 out of the 14 datasets, and in addition to that, it was also able to achieve higher accuracy with 5 of the 10 datasets. The datasets with which the ECSDT algorithm recorded both, the lowest cost and the highest accuracy are: Bupa, Hart, Haberman, Diabetes and Thyroid, whereas the datasets with which the ECSDT algorithm succeeded in recording lowest costs but failed to record a higher accuracy rates are: WDBC, IRIS, Tay, Glass, Ecoli.

Haberman was one of the datasets with which ECSDT achieved good results in terms of cost reduction and increased accuracy. ECSDT with NSGA-II was able to achieve lower expected cost than the other algorithms at an average cost of 0.43 units per example, and also was able to achieve higher accuracy than the other algorithms of 56.21%, while the lowest average cost and the highest accuracy rate recorded by the other algorithms were 0.50 and 50.33% respectively.

When comparing the general performance of the two optimization methods in this aspect, we find that their performance is fairly close where the overall cost average per example recorded by each method for all datasets is 7.92 for the OMOPSO and 7.97 for the NSGA-II.

For comparing tree sizes, we note that the new algorithm in both optimization methods was able to obtain lower costs only with larger tree sizes compared to other algorithms. As we can see from Table 5.9, ECSDT with both optimization methods produced an average tree size of 24.14 nodes which is higher than the other algorithms, except that recorded when applying (C-S-C + J48) which recorded the worst general average of the tree sizes with an average size of 25.37 nodes. Where the ECSDT was able to produce smaller decision trees than when applying (C-S-C + J48) on 6 out of the 10 datasets with which the ECSDT algorithm produced the best accuracy results.

Dataset	Optimizer	No. Ell	Avrg Cost	Accur -acy	SE	No. Ell	Avrg Cost	Accur -acy	SE	No. Ell	Avrg Cost	Accur -acy	SE	No. Ell	Avrg Cost	Accur -acy	SE	No. Ell	Avrg Cost	Accur -acy	SE
Bupa	OMOPSO	2	7.86	46.23	0.66	4	2.81	51.30	0.75	6	1.62	52.63	0.80	8	0.57	54.08	0.83	10	0.50	54.12	0.90
	NSGA-II	2	8.33	46.14	0.72	4	2.32	52.92	0.82	6	1.43	53.13	0.72	8	0.69	53.24	0.88	10	0.49	55.12	0.89
Hepatitis	OMOPSO	2	5.03	66.79	1.68	4	4.72	72.18	1.53	6	2.44	66.92	1.84	8	2.40	70.90	1.59	10	2.45	66.79	1.90
	NSGA-II	2	4.96	67.61	1.58	4	4.54	71.15	1.74	6	2.45	67.82	1.89	8	2.41	69.97	1.65	10	2.44	68.46	1.98
Heart	OMOPSO	2	9.27	48.58	0.41	4	3.76	52.48	0.62	6	1.85	56.74	0.58	8	0.97	58.70	0.58	10	0.40	60.29	0.89
	NSGA-II	2	8.32	48.70	0.43	4	3.51	51.81	0.61	6	1.67	56.40	0.73	8	0.97	58.85	0.61	10	0.39	61.11	0.90
Haberman	OMOPSO	2	3.26	49.08	2.38	4	2.29	50.43	2.40	6	1.79	54.02	2.20	8	0.63	55.65	2.14	10	0.44	55.98	2.21
	NSGA-II	2	3.06	49.25	2.41	4	2.07	50.20	2.38	6	1.52	54.18	2.16	8	0.70	55.13	2.07	10	0.44	56.21	2.14
Diabetes	OMOPSO	2	7.67	49.31	1.44	4	2.37	52.05	1.35	6	1.18	55.57	1.40	8	0.45	57.12	1.35	10	0.43	56.85	1.23
	NSGA-II	2	6.94	49.23	1.42	4	2.27	52.21	1.33	6	0.98	54.79	1.32	8	0.44	57.66	1.19	10	0.42	57.98	1.29
WDBC	OMOPSO	2	1.92	65.91	1.37	4	1.45	73.02	1.09	6	0.98	80.41	0.98	8	0.62	85.77	0.88	10	0.39	88.93	0.48
	NSGA-II	2	1.89	66.17	1.30	4	1.40	73.01	1.07	6	0.89	80.58	0.96	8	0.61	86.20	0.93	10	0.42	89.19	0.49
WPBC	OMOPSO	2	8.52	38.98	2.56	4	7.10	42.68	2.61	6	4.56	48.32	2.90	8	2.26	53.11	3.41	10	1.00	56.31	3.53
	NSGA-II	2	8.52	38.98	2.56	4	7.14	42.43	2.53	6	4.31	47.98	3.00	8	2.22	53.12	3.40	10	0.79	57.07	3.55
IRIS	OMOPSO	3	6.68	86.96	0.62	4	2.47	90.52	0.71	5	2.92	89.70	0.86	6	2.42	91.26	0.72	7	2.85	89.26	0.76
	NSGA-II	3	6.07	87.48	0.67	4	2.43	90.96	0.67	5	2.89	90.59	0.62	6	2.34	91.62	0.65	7	3.04	88.89	0.82
Hays	OMOPSO	3	34.08	45.84	0.36	6	31.19	50.63	0.65	9	28.70	54.27	0.73	12	25.52	59.17	0.82	15	21.91	63.44	0.83
	NSGA-II	3	34.27	45.94	0.46	6	30.90	50.84	0.62	9	28.41	54.17	0.70	12	25.36	58.86	0.81	15	22.36	63.03	0.90
Seeds	OMOPSO	3	27.01	60.90	0.36	6	22.51	64.60	0.51	9	17.63	71.11	0.42	12	12.42	77.72	0.28	15	8.15	82.80	0.32
	NSGA-II	3	27.02	61.37	0.36	6	22.36	64.55	0.46	9	17.83	70.68	0.45	12	12.27	77.30	0.34	15	8.46	82.91	0.30
Tae	OMOPSO	3	63.11	30.40	0.33	6	57.11	34.80	0.37	9	49.05	40.90	0.73	12	37.17	48.50	1.30	15	32.45	54.80	1.15
	NSGA-II	3	62.28	31.00	0.40	6	56.58	35.10	0.49	9	49.28	40.50	0.79	12	37.70	48.50	1.20	15	32.46	54.50	1.20
Thyroid	OMOPSO	3	14.76	82.33	0.32	6	11.17	86.44	0.13	9	6.89	90.93	0.24	12	5.09	91.39	0.42	15	4.51	93.95	0.13
	NSGA-II	3	13.21	82.87	0.47	6	10.43	87.52	0.27	9	5.73	91.16	0.34	12	4.65	92.48	0.27	15	4.63	94.26	0.18
Glass	OMOPSO	6	32.09	52.50	0.24	8	24.71	54.21	0.23	10	17.08	59.27	0.30	12	11.56	64.17	0.26	14	5.55	70.02	0.31
	NSGA-II	6	30.82	52.49	0.18	8	24.23	54.44	0.19	10	16.94	59.03	0.30	12	11.01	65.26	0.38	14	5.24	70.95	0.37
Ecoli	OMOPSO	6	30.20	48.44	1.58	8	26.28	51.41	1.64	10	23.33	53.76	1.71	12	20.06	58.78	1.90	14	19.83	62.99	1.96
	NSGA-II	6	29.76	48.93	1.56	8	26.22	51.82	1.71	10	22.15	56.25	1.83	12	20.06	59.08	1.92	14	14.69	62.39	1.96

Table 5.8: Cost-based results for all datasets using different number of ellipses

Dataset	C-S-C + J48					C-S-C + NBTree					MetaCost + J48				
	Size	Acc	SE	Cost	SE	Size	Acc	SE	Cost	SE	Size	Acc	SE	Cost	SE
Bupa	51	54.38	1.07	13.40	1.13	11	50.03	0.83	0.53	2.78	7.4	51.27	0.83	0.56	2.75
Hepatitis	2.6	64.49	3.14	0.60	1.90	5	58.72	3.12	0.50	2.00	3	60.64	3.16	0.60	2.00
Heart	39	57.33	0.75	11.50	2.37	17	57.37	0.99	0.80	2.38	8.2	55.07	0.92	0.52	2.66
Haberman	5	50.13	2.46	1.04	0.30	3	50.00	2.48	0.50	0.84	3	50.33	2.44	0.50	0.84
Diabetes	39	54.04	1.24	6.95	1.69	1	54.50	1.38	0.53	1.70	9.8	54.29	1.47	0.51	1.80
WDBC	25	88.31	1.79	1.63	0.61	23	86.70	1.05	1.07	0.05	22.3	90.65	0.43	0.48	0.54
WPBC	11.5	58.67	3.73	4.37	0.53	8.5	53.62	4.31	0.85	2.99	12.8	56.90	3.62	1.45	2.39
IRIS	8.11	91.41	0.95	3.95	0.31	9	90.74	0.65	3.98	0.33	6.44	91.85	0.75	3.84	0.19
Hays	23	81.67	0.61	16.95	3.13	9.67	60.42	0.67	33.99	3.11	21.3	80.10	0.49	20.84	3.27
Seeds	15	87.44	1.05	6.32	2.41	8	85.18	0.96	7.10	2.63	14	85.98	0.42	8.17	2.56
Tae	67	56.00	0.56	41.97	1.56	7	50.90	1.22	42.87	1.66	42	52.10	0.93	43.41	3.12
Thyroid	17	92.09	0.10	8.10	0.09	7	92.63	0.17	6.57	1.62	15	90.39	0.17	9.11	0.92
Glass	11	96.73	0.0	9.06	1.61	7	70.40	2.63	12.09	1.58	11	95.40	0.26	5.39	1.28
Ecoli	41	66.52	1.96	37.45	3.62	13	55.65	2.14	28.74	2.90	32	65.07	1.87	27.85	2.01
Overall Average	25.37	71.37	1.39	11.66	0.0	9.23	65.49	1.61	10.01	0.0	14.87	70.0	1.27	8.8	0.0

Dataset	MetaCost + NBTree					ECSDT+OMOPSO					ECSDT+NSGA-II				
	Size	Acc	SE	Cost	SE	Size	Acc	SE	Cost	SE	Size	Acc	SE	Cost	SE
Bupa	1	50.12	0.83	0.51	0.81	21	54.12	0.90	0.50	0.80	21	55.12	0.89	0.49	0.81
Hepatitis	2.4	59.23	3.2	0.50	0.10	17	70.90	1.59	2.40	0.20	17	69.97	1.65	2.41	0.20
Heart	3	53.40	0.89	0.51	0.67	21	60.30	0.89	0.40	0.78	21	61.11	0.90	0.39	0.79
Haberman	1	50.00	2.48	0.50	0.48	21	55.98	2.21	0.44	0.90	21	56.21	2.14	0.43	0.90
Diabetes	2	52.47	1.47	0.49	0.80	21	56.85	1.23	0.43	0.80	21	57.98	1.29	0.42	0.80
WDBC	16.3	89.07	0.33	0.53	0.49	21	88.93	0.48	0.39	0.65	21	89.19	0.49	0.42	0.60
WPBC	6.17	52.69	4.37	0.61	1.23	21	56.31	3.53	1.00	0.84	21	57.07	3.55	0.79	0.34
IRIS	7.67	87.63	0.87	5.11	1.46	13	91.26	0.72	2.42	0.21	13	91.62	0.65	2.33	0.30
Hays	1.33	56.88	1.19	35.73	2.38	31	63.44	0.83	32.39	4.72	31	63.03	0.90	33.10	3.01
Seeds	7	81.06	1.19	8.91	2.82	31	82.80	0.32	8.15	3.58	31	82.91	0.30	8.46	1.27
Tae	8.33	46.80	1.24	45.94	1.59	31	54.80	1.15	32.45	4.08	31	54.50	1.20	32.46	4.07
Thyroid	8	90.85	0.17	9.79	1.60	31	93.95	0.13	4.51	1.68	31	94.26	0.18	4.63	1.56
Glass	9	64.49	3.45	13.66	2.00	29	70.02	0.31	5.55	10.12	29	70.95	0.37	5.24	1.43
Ecoli	13	53.05	2.78	25.61	2.78	29	63.00	1.96	19.83	6.00	29	62.39	1.96	20.07	4.76
Overall Average	6.16	63.41	1.75	10.6		24.14	68.76	1.16	7.92	0.0	24.14	69.02	1.18	7.97	0.0

Table 5.9: Cost-based results for all datasets when applying different algorithms

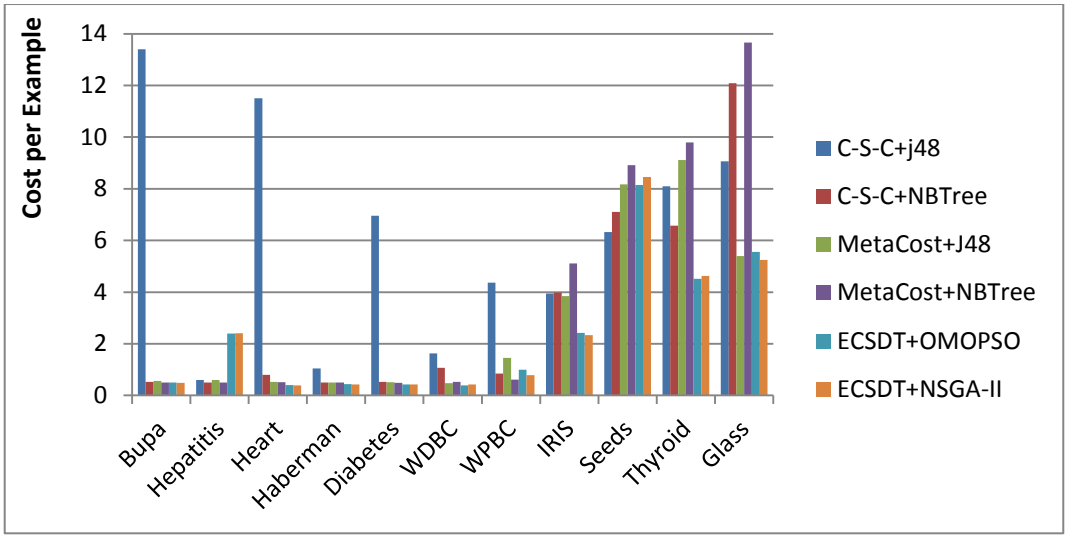

Figure 5.7: Cost comparison (Cost-based aspect)

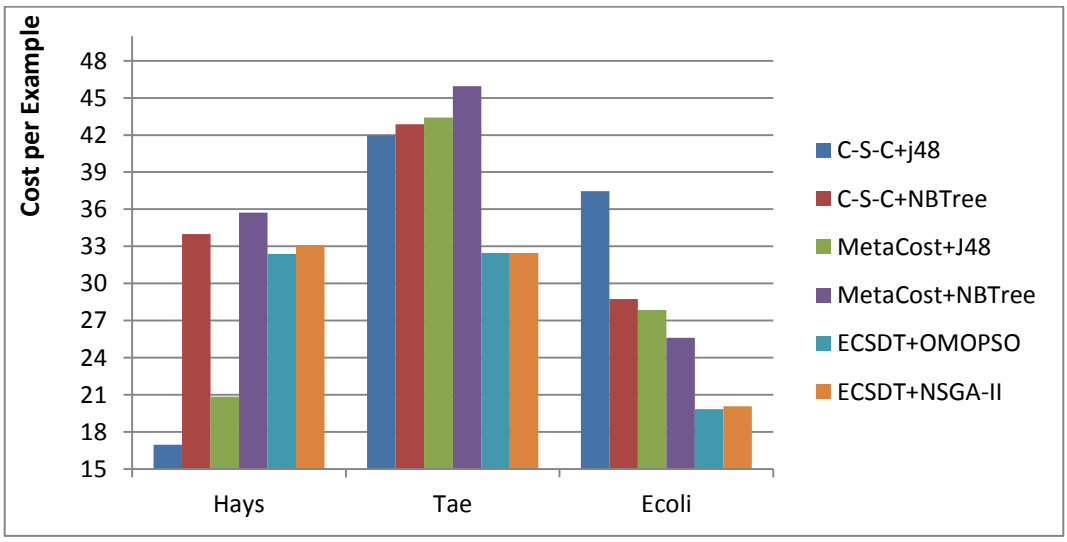

Figure 5.8: Cost comparison (Cost-based aspect)

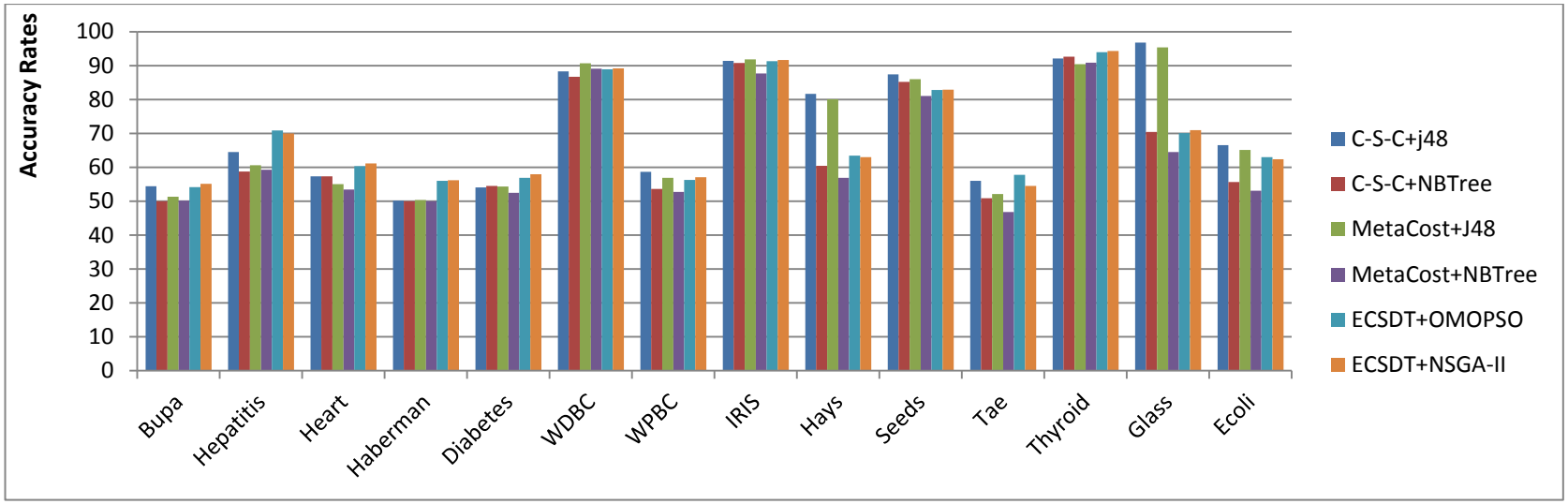

Figure 5.9: Accuracy comparison (Cost-based aspect)

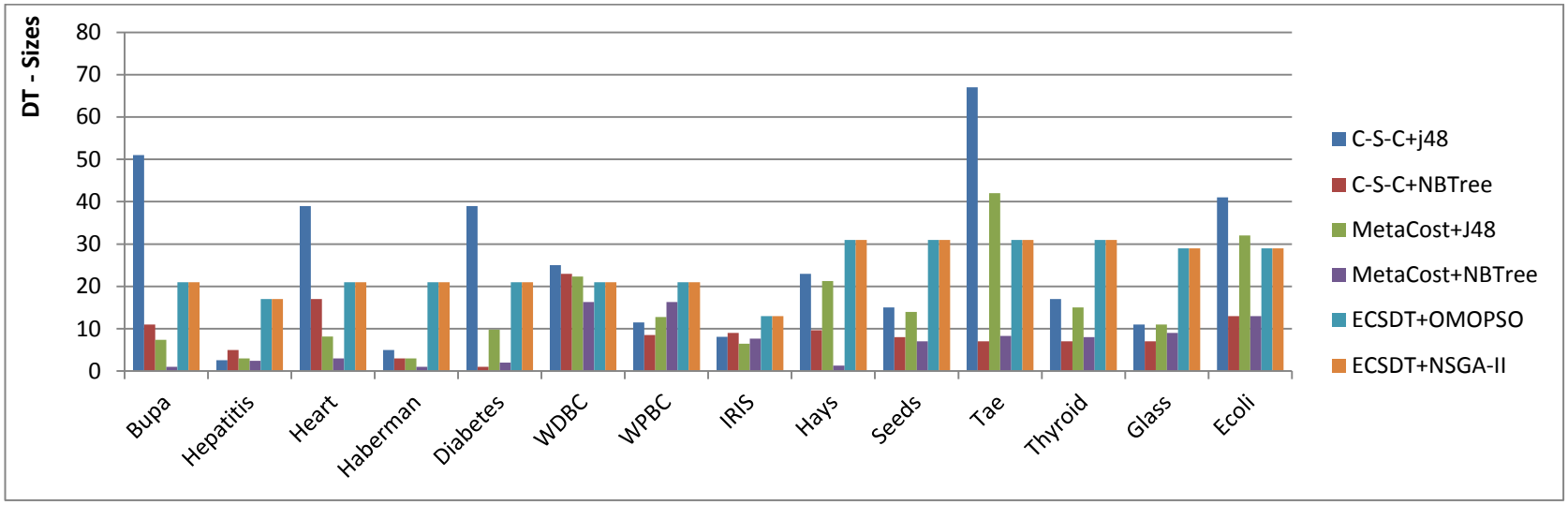

Figure 5.10: DT-size comparison (Cost-based aspect)

5.3.2.2 Cost-based results for the Bupa dataset:

Table 5.10 and Table 5.11 below present the misclassification costs along with the associated accuracies that are obtained when ECSDT is used with a different number of ellipses, each with different cost ratios applied to the Bupa dataset for both the optimization methods. From the tables, it can be observed that utilizing 10 ellipses in both optimization methods produced the lowest average costs per example from all the 10 used cost ratios.

	ECSDT + OMOPSO														
	2 ell			4 ell			6 ell			8 ell			10 ell		
	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE
Ratio - 1	0.72	43.76	0.71	0.57	42.61	0.67	0.58	42.03	0.54	0.59	48.69	0.71	0.53	48.82	0.62
Ratio - 2	1.00	42.89	0.54	0.81	47.24	0.57	0.80	48.40	0.72	0.80	48.98	0.56	0.57	43.19	0.49
Ratio - 3	1.18	39.71	0.48	0.83	45.50	0.77	1.10	47.53	0.58	0.84	44.92	0.52	0.54	46.37	0.53
Ratio - 4	3.50	39.13	0.54	3.47	42.02	0.49	3.47	42.02	0.66	0.57	43.18	0.78	0.56	44.05	0.55
Ratio - 5	9.32	37.10	0.60	6.34	44.63	0.81	3.42	47.53	0.57	0.54	46.37	0.63	0.53	46.66	0.74
Ratio - 6	1.41	48.11	0.54	0.51	57.10	0.58	0.48	57.39	0.61	0.41	59.71	0.68	0.39	60.86	0.66
Ratio - 7	4.91	48.98	0.72	0.68	60.00	0.73	0.54	60.57	0.77	0.50	64.63	0.58	0.50	63.76	0.72
Ratio - 8	5.06	53.33	0.49	0.99	58.84	0.55	0.65	63.76	0.82	0.66	62.31	0.93	0.63	65.21	0.84
Ratio - 9	19.26	54.20	0.76	4.77	57.10	0.67	1.87	57.68	0.52	0.39	60.86	0.62	0.37	62.60	0.56
Ratio - 10	32.30	55.07	0.98	9.11	57.97	0.65	3.30	59.42	0.84	0.39	61.15	0.55	0.40	59.71	0.59
Average	**7.86**	**46.23**	**0.64**	**2.81**	**51.30**	**0.65**	**1.62**	**52.63**	**0.66**	**0.57**	**54.08**	**0.66**	**0.50**	**54.12**	**0.63**

Table 5.10: Bupa Cost-Based results for each number of ellipses (ECSDT+OMOPSO)

	ECSDT + NSGA-II														
	2 ell			4 ell			6 ell			8ell			10 ell		
	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE
Ratio - 1	0.70	42.05	0.52	0.64	44.05	0.65	0.57	44.41	0.63	0.55	47.24	0.66	0.52	50.72	0.58
Ratio - 2	1.00	42.60	0.46	0.66	48.69	0.67	0.68	46.67	0.68	0.56	43.76	0.61	0.54	45.50	0.54
Ratio - 3	1.19	38.55	0.55	0.81	47.82	0.51	0.81	47.82	0.53	0.56	44.34	0.52	0.52	46.66	0.54
Ratio - 4	3.51	37.97	0.71	3.45	44.05	0.72	1.96	48.69	0.84	1.99	46.08	0.48	0.54	45.50	0.67
Ratio - 5	12.21	36.81	0.69	6.36	42.89	0.39	3.44	45.50	0.57	0.56	43.76	0.53	0.54	46.08	0.74
Ratio - 6	1.14	50.72	0.48	0.46	59.71	0.77	0.48	57.39	0.55	0.39	60.57	0.72	0.38	61.73	0.55
Ratio - 7	4.03	50.72	0.58	0.66	62.31	0.97	0.56	58.26	0.77	0.56	58.26	0.82	0.53	61.15	0.56
Ratio - 8	5.06	53.33	0.67	0.64	64.34	0.69	0.64	64.34	0.49	0.93	64.63	0.55	0.62	66.37	0.58
Ratio - 9	19.26	54.20	0.72	3.32	57.39	0.33	1.87	57.68	0.70	0.38	61.73	0.75	0.35	64.63	0.81
Ratio - 10	35.20	54.49	0.55	6.21	57.97	0.58	3.29	60.57	0.58	0.38	62.02	0.63	0.37	62.89	0.76
Average	**8.33**	**46.14**	**0.59**	**2.32**	**52.92**	**0.63**	**1.43**	**53.13**	**0.63**	**0.69**	**53.24**	**0.63**	**0.49**	**55.12**	**0.63**

Table 5.11: Bupa Cost-Based results for each number of ellipses (ECSDT+NSGA-II)

Therefore, the results obtained when using 10 ellipses were used in comparing the performance of ECSDT with the other algorithms as presented below in Table 5.12.

Algorithm	Bupa Cost-based Results								
	Ratio - 1			Ratio - 2			Ratio - 3		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	46.67	1.08	51	43.77	2.55	51	43.8	4.58	51
MetaCost+J48	50.14	0.81	37	42.32	0.72	1	42.6	0.57	1
C.S.C+NBTree	42.32	0.66	11	42.32	0.58	11	42.3	0.58	11
MetaCost+NBTree	43.19	0.57	1	42.03	0.58	1	42.00	0.58	1
ECSDT-PSO-10ell	48.82	0.53	21	43.19	0.57	21	46.37	0.54	21
ECSDT-GA-10ell	50.72	0.52	21	45.50	0.54	21	46.66	0.52	21

Algorithm	Ratio - 4			Ratio - 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	43.77	20.81	51	43.77	41.10	51	66.96	0.72	51
MetaCost+J48	42.03	0.58	1	42.03	0.58	1	61.74	0.49	23
C.S.C+NBTree	42.03	0.58	11	42.03	0.58	11	57.68	0.48	11
MetaCost+NBTree	42.03	0.58	1	42.03	0.58	1	57.97	0.42	1
ECSDT-PSO-10ell	44.05	0.56	21	46.66	0.53	21	60.86	0.39	21
ECSDT-GA-10ell	45.50	0.54	21	46.08	0.54	21	61.73	0.38	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	63.80	2.21	51	63.77	4.09	51	63.77	19.17	51
MetaCost+J48	58.00	0.56	7	57.97	0.42	1	57.97	0.42	1
C.S.C+NBTree	57.70	0.57	11	57.97	0.42	11	57.97	0.42	11
MetaCost+NBTree	58.00	0.42	1	57.97	0.42	1	57.97	0.42	1
ECSDT-PSO-10ell	63.76	0.50	21	65.21	0.63	21	62.60	0.37	21
ECSDT-GA-10ell	61.15	0.53	21	66.37	0.62	21	64.63	0.35	21

Algorithm	Ratio - 10			Averages		
	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	63.77	38.01	51	54.38	13.43	51
MetaCost+J48	57.97	0.42	1	51.27	0.56	7.4
C.S.C+NBTree	57.97	0.42	11	50.02	0.53	11
MetaCost+NBTree	57.97	0.42	1	50.11	0.50	1
ECSDT-PSO-10ell	59.71	0.40	21	54.12	0.50	21
ECSDT-GA-10ell	62.89	0.37	21	55.123	0.49	21

Table 5.12: Bupa Cost-Based results obtained by the comparative algorithms

From Table 5.12, we can see that applying the CostSensitiveClassifier with J48 (C.S.C+J48) on Bupa dataset always gives the worst results in terms of cost compared to all other algorithms. It has therefore been excluded from the cost comparison chart presented in Figure 5.11. Figure 5.11 and Figure 5.12 below; give a better idea about the results listed above in Table 5.12. The first chart presents a comparison in terms of cost, excluding the results of (C.S.C+J48) for the reason mentioned above, and the second chart presents a comparison in terms of accuracy including the results of the (C.S.C+J48).

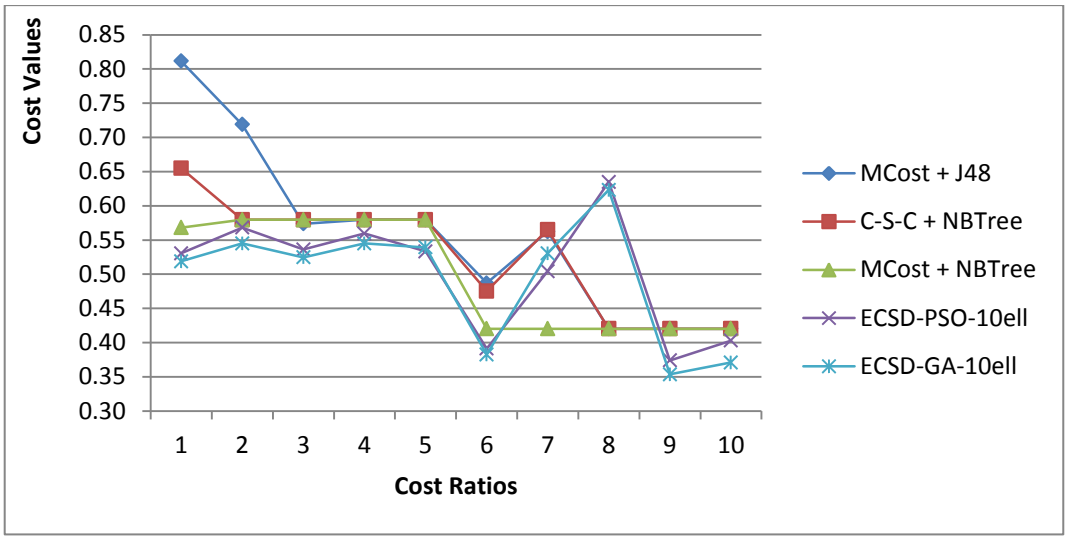

Figure 5.11: Bupa cost comparison for the Cost-Based aspect

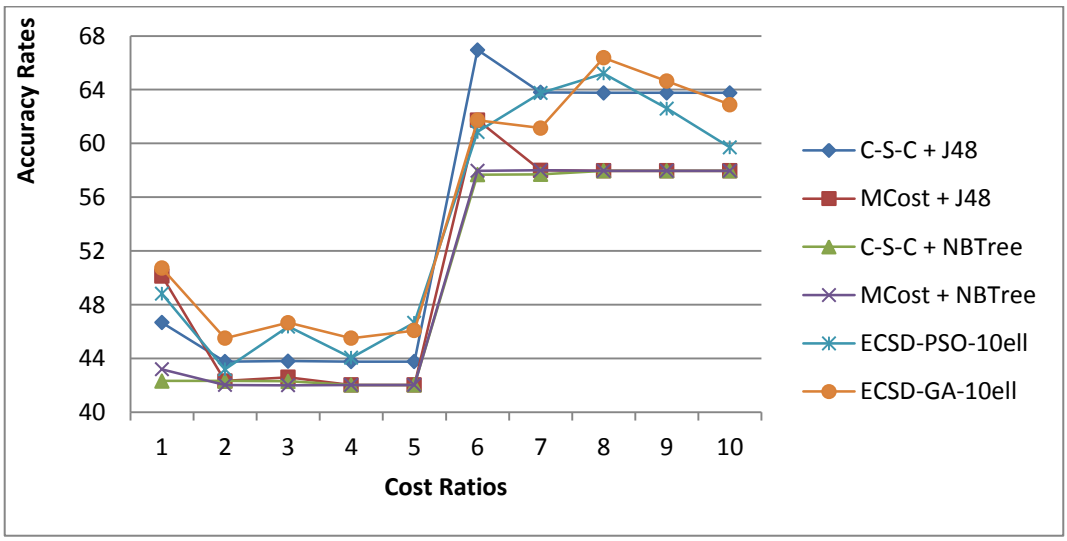

Figure 5.12: Bupa accuracy comparison for the Cost-Based aspect

Figure 5.11 and Figure 5.12 show that:

- ECSDT with both optimization methods (PSO and GA) obtained better results compared to the other algorithms in terms of reducing the misclassification costs with most cost ratios.
- ECSDT is also able to maintain higher rates of classification accuracy on 8 of the 10 cost ratios used.
- We can also note that there is little variation between the performance of the two optimization methods adopted (PSO and GA).
- The improvements in minimizing cost and maximizing accuracy that has been achieved by the ECSDT algorithm is at the expense of using larger trees. As we can

see from Figure 5.13 below, the sizes of the trees produced by the ECSDT algorithm with most of the cost ratios were bigger than those produced by most other algorithms except that with the (C.S.C+J48) that produced the largest trees compared to all other algorithms.

- With respect to the use of the MetaCost which in most cases produced the smallest trees, it is worth noting that it often produces trees with a size of 1 node only, and this is due to the fact that MetaCost usually neglects the less-costly class by placing only one splitting boundary which ensures that all examples belonging to the more-costly class fall within that boundary.

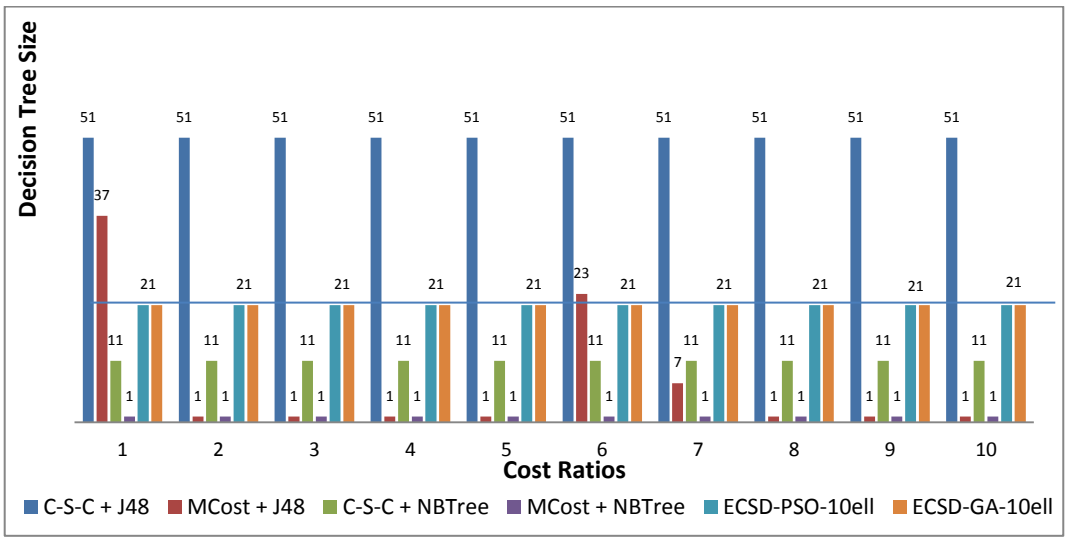

Figure 5.13: Bupa DT-size comparison for the Cost-Based aspect

5.3.2.3 Cost-based results for the IRIS dataset

Table 5.13 and Table 5.14 below present the misclassification costs along with the associated accuracies that are obtained when ECSDT is used with a different number of ellipses, each with different cost ratios applied to the IRIS dataset for both the optimization methods (OMOPSO and NSGA-II).

	ECSDT + OMOPSO														
	3 ell			4 ell			5 ell			6 ell			7 ell		
	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE
Ratio - 1	1.73	82.66	0.56	0.93	90.66	0.55	0.93	90.66	0.66	1.47	85.33	0.63	1.07	89.33	0.61
Ratio - 2	0.50	86.00	0.57	0.39	84.67	0.56	0.39	84.67	0.64	0.22	90.00	0.67	0.25	87.33	0.52
Ratio - 3	1.58	74.00	0.55	0.13	86.66	0.48	0.77	89.33	0.58	0.10	90.00	0.58	0.80	86.00	0.52
Ratio - 4	0.57	92.00	0.49	0.23	77.33	0.62	0.28	72.00	0.58	0.23	77.33	0.53	0.27	73.33	0.56
Ratio - 5	0.13	90.00	0.63	0.09	94.00	0.42	0.09	94.00	0.72	0.06	96.66	0.73	0.12	90.66	0.58
Ratio - 6	0.24	88.00	0.66	0.12	94.00	0.4	0.17	88.66	0.57	0.07	93.33	0.49	0.15	91.33	0.77
Ratio - 7	21.33	90.00	0.52	8.33	94.00	0.58	9.67	95.33	0.55	7.00	96.66	0.52	8.33	94.00	0.48
Ratio - 8	17.33	89.33	0.52	5.00	96.66	0.45	7.00	96.00	0.49	5.00	96.66	0.57	7.00	96.00	0.55
Ratio - 9	16.67	90.66	0.47	7.00	96.66	0.51	7.00	96.66	0.61	7.67	95.33	0.64	7.67	95.33	0.65
Average	**6.68**	**86.96**	**0.55**	**2.47**	**90.52**	**0.51**	**2.92**	**89.7**	**0.6**	**2.42**	**91.26**	**0.6**	**2.85**	**89.26**	**0.58**

Table 5.13: IRIS Cost-Based results for each number of ellipses (ECSDT+OMOPSO)

	ECSDT + NSGA-II														
	3 ell			4 ell			5 ell			6 ell			7 ell		
	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE
Ratio - 1	1.73	82.66	0.58	1.07	85.33	0.54	0.93	90.66	0.64	1.07	85.33	0.59	1.33	86.66	0.55
Ratio - 2	0.50	86.00	0.58	0.31	87.33	0.54	0.23	88.66	0.66	0.23	88.66	0.53	0.33	86.00	0.53
Ratio - 3	1.58	74.00	0.46	0.77	89.33	0.77	0.77	89.33	0.57	0.10	90.00	0.47	0.13	86.66	0.61
Ratio - 4	0.47	93.33	0.44	0.20	80.00	0.34	0.23	77.33	0.51	0.19	80.66	0.75	0.28	72.00	0.67
Ratio - 5	0.13	90.00	0.61	0.07	96.00	0.83	0.09	94.00	0.62	0.06	96.66	0.71	0.09	94.00	0.81
Ratio - 6	0.24	88.00	0.68	0.09	90.66	0.42	0.16	90.00	0.58	0.07	93.33	0.54	0.16	90.00	0.62
Ratio - 7	16.33	92.66	0.49	7.33	96.66	0.63	8.33	94.00	0.55	7.33	96.66	0.58	8.67	93.33	0.53
Ratio - 8	15.33	92.00	0.52	5.00	96.66	0.34	8.00	95.33	0.57	5.00	96.66	0.55	6.00	96.66	0.57
Ratio - 9	18.33	88.66	0.54	7.00	96.66	0.38	7.33	96.00	0.60	7.00	96.66	0.49	10.33	94.66	0.66
Average	**6.07**	**87.48**	**0.54**	**2.43**	**90.96**	**0.53**	**2.9**	**90.59**	**0.59**	**2.34**	**91.62**	**0.58**	**3.04**	**88.89**	**0.62**

Table 5.14: IRIS Cost-Based results for each number of ellipses (ECSDT+NSGA-II)

From the above tables, it can be observed that utilizing 6 ellipses in both optimization methods produced the lowest average costs per example from the 9 cost ratios. Therefore, the results obtained when using 6 ellipses were used in the comparison with the results of other algorithms as shown in Table 5.15.

Algorithm	IRIS Cost-based Results								
	Ratio - 1			Ratio - 2			Ratio - 3		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	96.00	1.54	9	96.00	0.82	9	69.33	1.03	9
MetaCost+J48	93.33	1.75	4	94.00	0.84	7	74.00	0.38	3
C.S.C+NBTree	84.00	2.44	9	80.67	0.55	9	84.67	0.15	9
MetaCost+NBTree	84.00	2.62	11	85.33	0.37	9	68.67	1.21	5
ECSDT-PSO-6ell	85.33	1.47	13	90.00	0.22	13	90.00	0.10	13
ECSDT-GA-6ell	85.33	1.07	13	88.66	0.23	13	90.00	0.10	13

Algorithm	Ratio - 4			Ratio - 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	93.33	0.41	11	94.67	0.14	5	90.00	0.31	7
MetaCost+J48	94.67	0.28	7	93.33	0.18	7	93.33	0.09	9
C.S.C+NBTree	94.67	0.40	9	95.33	0.13	9	93.33	0.13	9
MetaCost+NBTree	91.33	0.51	9	90.67	0.15	1	90.00	0.19	9
ECSDT-PSO-6ell	77.33	0.23	13	96.66	0.06	13	93.33	0.07	13
ECSDT-GA-6ell	80.66	0.19	13	96.66	0.06	13	93.33	0.07	13

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	96.00	8.33	7	92.67	12.67	7	94.66	10.33	9
MetaCost+J48	94.67	11.33	7	95.33	8.33	7	94.00	11.33	7
C.S.C+NBTree	94.67	12.00	9	94.67	9.33	9	94.66	10.67	9
MetaCost+NBTree	92.67	16.33	9	92.67	12.67	9	93.33	12.00	7
ECSDT-PSO-6ell	96.66	7.00	13	96.66	5.00	13	95.33	7.67	13
ECSDT-GA-6ell	96.66	7.33	13	96.66	5.00	13	96.66	7.00	13

Algorithm	Averages		
	Acc	Cost	Size
C.S.C+J48	91.41	3.95	8.11
MetaCost+J48	91.85	3.83	6.44
C.S.C+NBTree	90.74	3.98	9.00
MetaCost+NBTree	87.63	5.12	7.67
ECSDT-PSO-6ell	91.26	2.42	13.00
ECSDT-GA-6ell	91.62	2.34	13.00

Table 5.15: IRIS Cost-Based results obtained by the comparative algorithms (IRIS)

Figure 5.14, Figure 5.15 and Figure 5.16; present the results in charts. The first chart presents a comparison in terms of cost, the second chart presents a comparison in terms of accuracy and the third chart compares the sizes of the trees produced by the algorithms.

From the above table and figures, it can be observed that:

- ECSDT with both optimization methods obtained better results compared to the other algorithms in terms of reducing the misclassification costs with all cost ratios.
- ECSDT was able to achieve higher accuracy with 5 of the 9 cost ratios used.
- ECSDT with both optimization methods achieved higher accuracies than the NBTree in 8 of 9 cost ratios and higher accuracies than the J48 in 5 of 9 cost ratios.

- Although J48 was able to achieve better accuracy results than ECSDT in 3 of 9 cost ratios, it couldn't achieve lower costs than the ECSDT.
- As shown in the Figure 5.16, ECSDT sacrificed the size of decision trees in order to achieve better results.

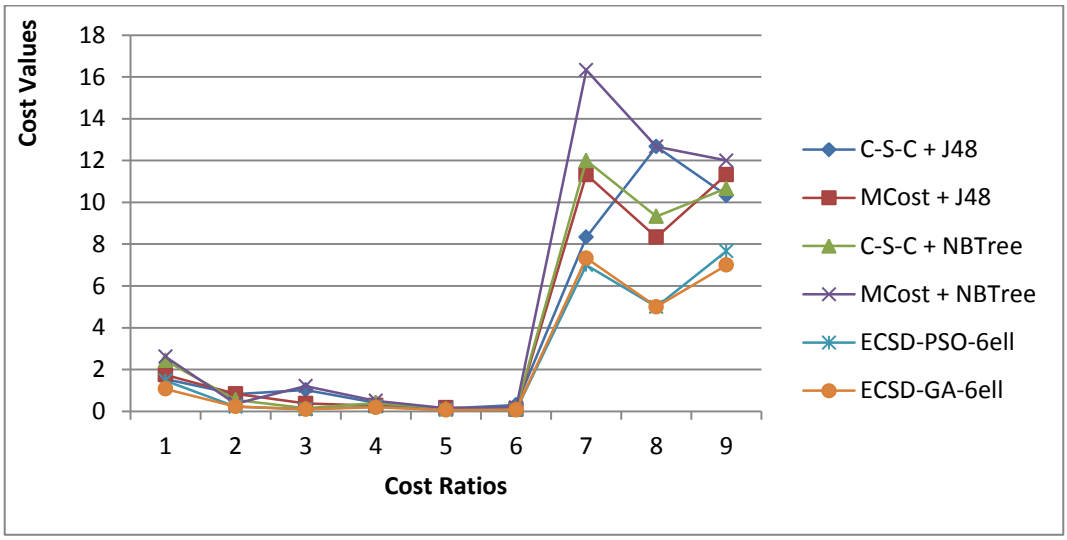

Figure 5.14: IRIS cost comparison for the Cost-Based aspect

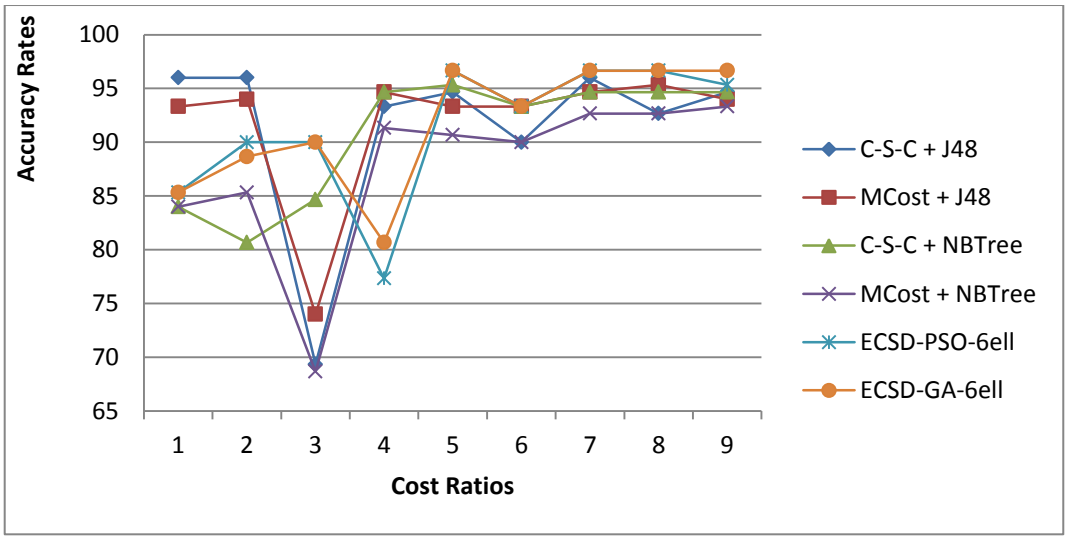

Figure 5.15: IRIS accuracy comparison for the Cost-Based aspect

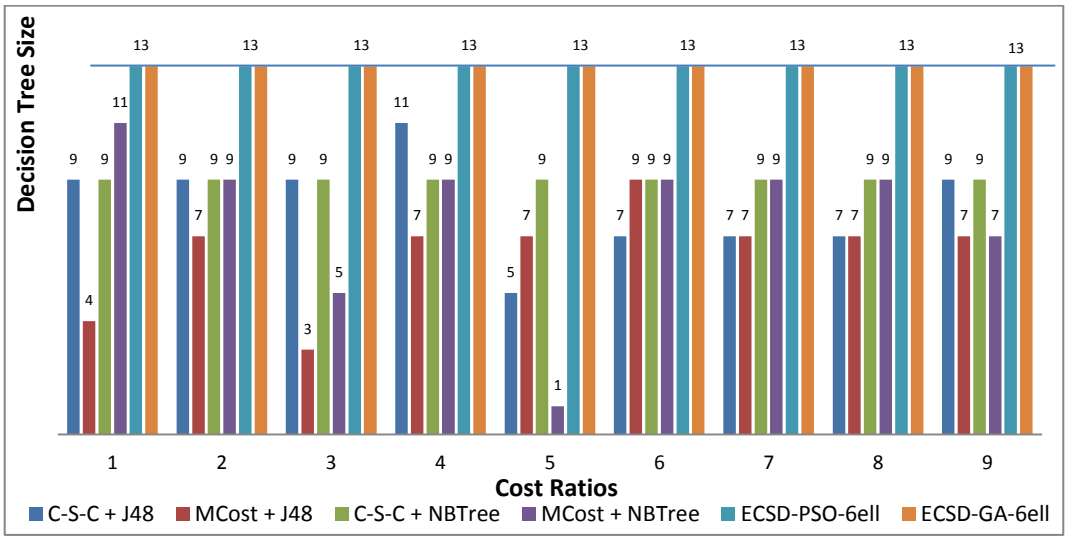

Figure 5.16: IRIS DT-size comparison for the Cost-Based aspect

5.3.2.4 Cost-based results for the Ecoli dataset

Table 5.16 and Table 5.17 present the results that are obtained when the cost-based aspect of ECSDT is used with a different number of ellipses, each with different cost ratios applied to the Ecoli dataset for both the optimization methods (OMOPSO and NSGA-II).

	ECSDT + OMOPSO														
	6 ell			8 ell			10 ell			12 ell			14 ell		
	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE
Ratio - 1	13.61	21.13	0.66	11.96	22.92	0.58	11.06	24.11	0.63	10.62	23.81	0.53	10.30	26.79	0.71
Ratio - 2	30.12	50.3	0.52	29.33	55.65	0.46	27.10	56.85	0.66	24.67	61.31	0.57	24.81	60.12	0.62
Ratio - 3	42.09	60.42	0.50	36.70	64.88	0.58	35.51	66.67	0.52	30.39	69.35	0.51	35.99	72.92	0.57
Ratio - 4	24.95	59.52	0.59	21.15	62.2	0.52	18.74	66.37	0.58	16.87	71.13	0.63	14.62	75.6	0.52
Ratio - 5	24.74	56.55	0.73	19.79	58.33	0.62	15.70	61.9	0.62	11.78	68.15	0.63	13.11	72.02	0.48
Ratio - 6	48.32	51.49	0.56	42.09	52.08	0.46	36.10	54.76	0.47	30.57	63.69	0.59	27.75	70.83	0.57
Ratio - 7	35.61	44.35	0.62	28.40	48.51	0.58	28.02	50.6	0.54	23.85	55.95	0.58	20.50	63.99	0.58
Ratio - 8	22.17	43.75	0.72	20.81	46.7	0.55	14.41	48.81	0.59	11.75	56.85	0.50	11.56	61.61	0.65
Average	**30.2**	**48.44**	**0.61**	**26.28**	**51.41**	**0.54**	**23.33**	**53.76**	**0.58**	**20.06**	**58.78**	**0.57**	**19.83**	**62.99**	**0.59**

Table 5.16: Ecoli Cost-Based results for each number of ellipses (ECSDT+OMOPSO)

	ECSDT + NSGA-II														
	6 ell			8 ell			10 ell			12 ell			14 ell		
	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE
Ratio - 1	13.45	22.02	0.51	12.59	23.21	0.56	11.07	24.11	0.54	10.36	24.7	0.49	10.27	25.89	0.65
Ratio - 2	29.52	51.19	0.53	29.03	55.95	0.64	25.72	58.63	0.56	24.07	60.42	0.56	23.92	62.8	0.63
Ratio - 3	40.57	60.4	0.56	36.02	65.77	0.73	31.29	69.05	0.47	30.07	67.56	0.67	34.50	73.81	0.51
Ratio - 4	24.50	60.12	0.54	20.69	64.58	0.44	18.24	70.24	0.61	16.63	72.92	0.65	14.69	74.7	0.57
Ratio - 5	24.74	56.55	0.64	18.84	58.63	0.63	16.71	63.99	0.62	13.59	71.13	0.51	12.59	69.94	0.61
Ratio - 6	47.87	52.08	0.58	42.41	54.17	0.52	35.21	58.63	0.53	30.06	64.88	0.58	27.93	69.05	0.52
Ratio - 7	35.28	45.24	0.59	28.45	47.92	0.53	24.60	55.65	0.65	23.81	56.85	0.53	25.01	62.8	0.63
Ratio - 8	22.17	43.8	0.62	21.72	44.35	0.49	14.34	49.7	0.67	11.91	54.17	0.66	11.62	60.12	0.55
Average	**29.76**	**48.93**	**0.57**	**26.22**	**51.82**	**0.57**	**22.15**	**56.25**	**0.58**	**20.06**	**59.08**	**0.58**	**20.07**	**62.39**	**0.58**

Table 5.17: Ecoli Cost-Based results for each number of ellipses (ECSDT+NSGA-II)

From Tables 5.16 and 5.17, it can be observed that, when applying OMOPSO with ECSDT, 14 ellipses produced the lowest average costs per example, whereas, when applying NSGA-II with ECSDT, 12 ellipses produced the lowest average costs per example. Therefore, the associated results of using 14 ellipses with OMOPSO and the associated results of using 12 ellipses with NSGA-II are used in the comparison with the results obtained by other algorithms as shown in Table 5.18.

Algorithm	Ecoli Cost-based Results								
	Ratio - 1			Ratio - 2			Ratio - 3		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	31.84	31.19	41	66.07	41.95	41	78.27	57.06	41
MetaCost+J48	32.14	19.55	21	65.17	40.94	31	78.86	33.52	17
C.S.C+NBTree	26.48	20.07	13	63.98	31.42	13	74.70	44.68	13
MetaCost+NBTree	25.00	19.19	9	61.01	34.99	23	76.78	37.24	13
ECSDT-PSO-12ell	26.79	10.30	29	60.12	24.81	29	72.92	35.99	29
ECSDT-GA-14ell	24.70	10.36	25	60.42	24.07	25	67.56	30.07	25
Algorithm	**Ratio - 4**			**Ratio - 5**			**Ratio - 6**		
	Acc	**Cost**	**Size**	**Acc**	**Cost**	**Size**	**Acc**	**Cost**	**Size**
C.S.C+J48	77.67	32.95	41	75.29	22.48	41	75.29	50.99	41
MetaCost+J48	78.27	20.02	29	74.40	17.81	25	74.40	42.50	41
C.S.C+NBTree	72.61	28.77	13	66.66	24.49	13	66.66	37.28	13
MetaCost+NBTree	72.61	28.80	13	65.17	21.83	5	65.17	35.16	9
ECSDT-PSO-12ell	75.60	14.62	29	72.02	13.11	29	70.83	27.75	29
ECSDT-GA-14ell	72.92	16.63	25	71.13	13.59	25	64.88	30.06	25
Algorithm	**Ratio - 7**			**Ratio - 8**			**Averages**		
	Acc	**Cost**	**Size**	**Acc**	**Cost**	**Size**	**Acc**	**Cost**	**Size**
C.S.C+J48	69.34	40.40	41	57.44	22.56	41	66.40	37.45	41.00
MetaCost+J48	62.50	31.65	43	60.71	16.78	45	65.81	27.85	31.50
C.S.C+NBTree	47.32	24.06	13	38.392	19.14	13	57.11	28.74	13.00
MetaCost+NBTree	50.59	27.60	11	14.583	32.12	19	53.87	29.62	12.75
ECSDT-PSO-12ell	63.99	20.5	29	61.61	11.56	29	62.99	19.83	29.00
ECSDT-GA-14ell	56.85	23.81	25	54.17	11.91	25	59.08	20.06	25.00

Table 5.18: Ecoli Cost-Based results obtained by the comparative algorithms

From the previous table and Figures 5.17, 5.18 and 5.19 we can observe the following conclusions for the Ecoli related cost-based results:

- With both optimization methods (OMOPSO and NSGA-II), ECSDT was able to obtain the lowest costs with all cost ratios compared to the other algorithms.
- OMOPSO achieved lower costs than NSGA-II in 7 of the 8 used cost ratios.
- In addition to the lowest costs achieved by OMOPSO, it also managed to achieve higher accuracies than C.S.C+NBTree and MetaCost+NBTree on 6 of the 8 cost ratios.
- As presented in Table 5.18 and shown in the Figure 5.19, ECSDT produced smaller trees on average compared to C.S.C+J48 and MetaCost+J48 while maintaining lower cost averages with both optimization methods.

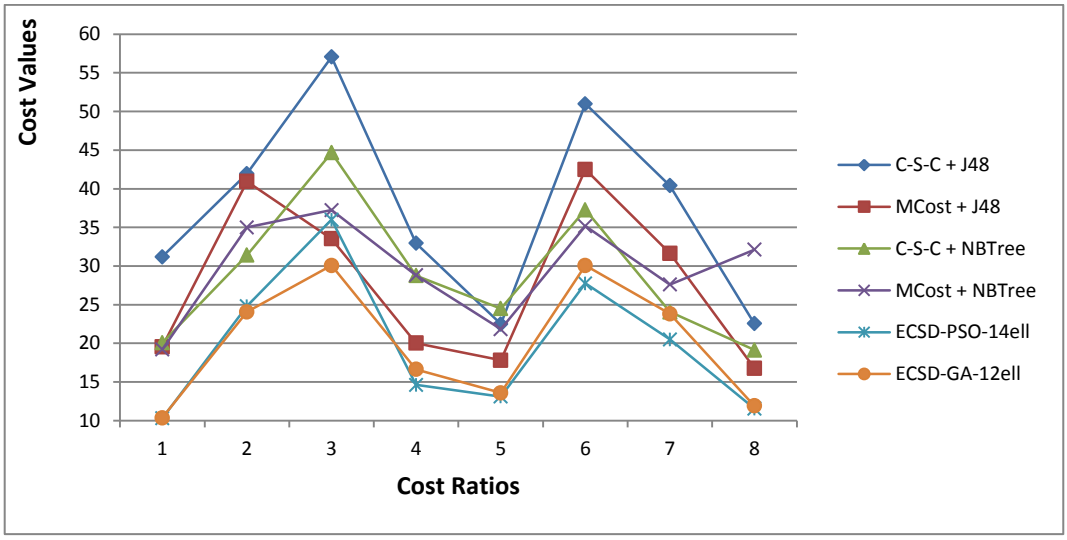

Figure 5.17: Ecoli cost comparison for the Cost-Based aspect

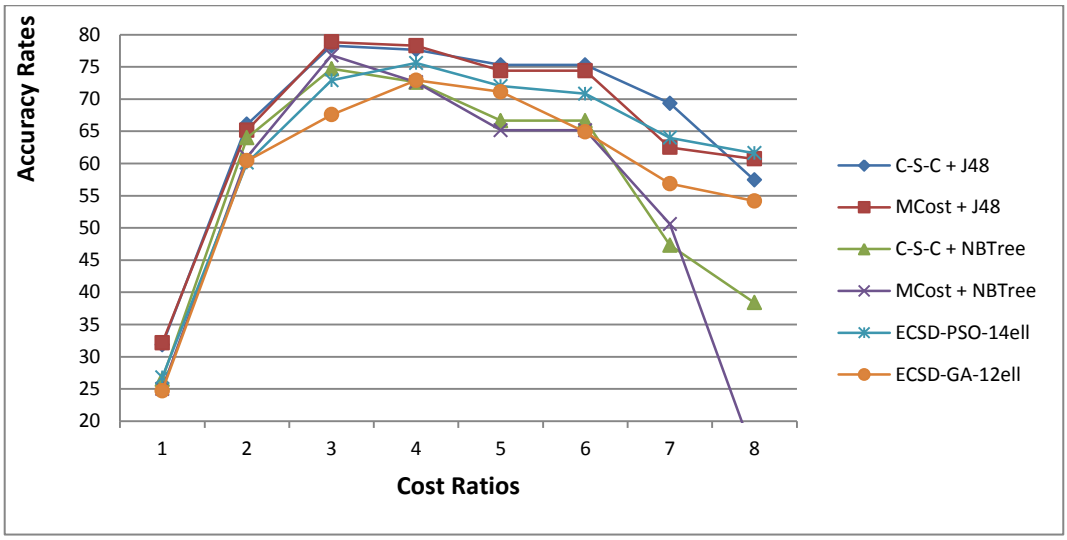

Figure 5.18: Ecoli accuracy comparison for the Cost-Based aspect

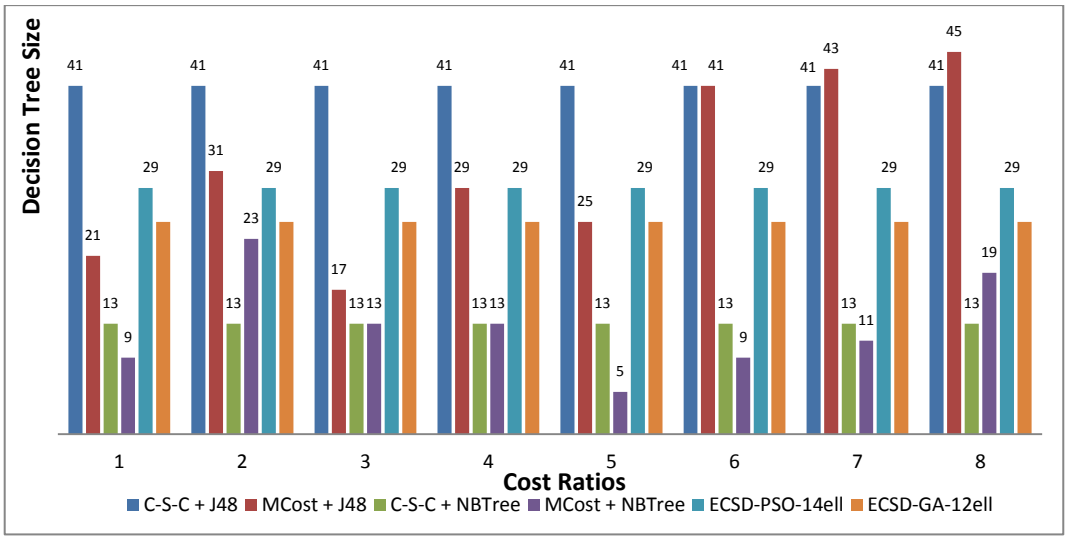

Figure 5.19: Ecoli DT-size comparison for the Cost-Based aspect

5.3.3 Empirical Comparison Based on Both Accuracy and Cost

As mentioned before in Section 5.3, this aspect considers accuracy that is influenced by the misclassification cost as the objective function for inducing the decision trees and ECSDT aims to find a solution that gives the best value when subtracting the average cost from the accuracy rate.

The methodology adopted for the comparison in this aspect is that, for each optimization method only the results related to the number of ellipses that gives the best accuracy are selected along with its associated cost for the comparison, and because the cost was included in this aspect, therefore the same algorithms used with the previous aspect namely CostSensitiveClassifier and MetaCost with J48 and NBTree were included for the assessment of the ECSDT in this aspect.

The following sub-sections explain and discuss the results obtained from the (Accuracy + cost) based evaluation. Section 5.3.3.1 presents the results for all the datasets; Sections 5.3.3.2, 5.3.3.3 and 5.3.3.4 provide some more detailed discussion for three data sets, namely Bupa, IRIS and Ecoli datasets respectively. All tables and results related to this aspect are provided in Appendix D.

5.3.3.1 Discussion for (Accuracy + Cost) based results

Table 5.19 depicts both accuracy and misclassification costs obtained from the (Accuracy + cost) based experiments for all datasets when using different numbers of ellipses in both optimization methods. From the table we can see that:

- Increasing the number of ellipses has a positive effect on improving the performance of ECSDT for both cost and accuracy with all datasets.
- The use of the NSGA-II method obtained higher accuracy results than the use of the OMOPSO method on 10 of 14 datasets and also recorded lower costs on 9 of those 10 datasets, while the use of the OMOPSO method achieved higher accuracy results than the NSGA-II method on only 3 of the 14 datasets and achieved lower cost with 2 of those 3 datasets.
- With the Diabetes and the IRIS datasets, the NSGA-II method was able to achieve better results in both accuracy and cost than the OMOPSO method and with fewer ellipses. As we can see from the table, for the Diabetes, NSGA-II obtained an accuracy rate of 63.20% and an average cost of 0.48 with 8 ellipses, whereas, the lowest average cost obtained by OMOPSO is 0.77 with an accuracy rate of 60.95% with the use of 10 ellipses. For IRIS, NSGA-II obtained an accuracy rate of 95.40% and an average cost of 2.79 with only the use of 6 ellipses, whereas, the lowest average cost obtained by OMOPSO is 3.32 with an accuracy rate of 93.26% and that is with the use of 7 ellipses.

Dataset	Optimization Method	No. Ell	Accu-racy	Avrg Cost	No. Ell	Accu-racy	Avrg Cost	No. Ell	Accu-racy	Avrg Cost	No. Ell	Accu-racy	Avrg Cost	No. Ell	Accu-racy	Avrg Cost
Bupa	OMOPSO	2	48.12±0.63	9.39	4	55.64±0.58	4.04	6	56.37±0.60	2.80	8	57.85±0.56	2.30	10	59.92±0.63	1.41
	NSGA-II	2	49.53±0.63	8.27	4	56.36±0.59	3.23	6	56.80±0.61	3.67	8	58.67±0.58	2.00	10	61.09±0.65	1.37
Hepatitis	OMOPSO	2	69.10±1.13	9.08	4	75.13±0.85	6.83	6	80.25±0.82	4.45	8	81.79±0.54	4.44	10	82.04±0.72	4.43
	NSGA-II	2	70.74±1.11	8.99	4	76.15±0.76	6.69	6	81.66±0.69	5.08	8	82.30±0.49	4.43	10	82.56±0.61	4.43
Heart	OMOPSO	2	51.48±0.57	12.47	4	57.77±0.67	6.77	6	63.52±0.53	2.99	8	67.51±0.56	2.25	10	69.84±0.57	1.61
	NSGA-II	2	53.07±0.71	11.03	4	59.33±0.61	5.70	6	64.11±0.58	2.71	8	67.81±0.58	1.91	10	69.58±0.61	1.54
Haberman	OMOPSO	2	49.28±2.44	4.61	4	51.99±2.45	3.39	6	55.95±2.14	2.69	8	59.48±2.13	1.57	10	61.21±2.16	1.48
	NSGA-II	2	49.84±2.45	4.23	4	51.99±2.45	3.20	6	57.19±2.11	2.61	8	59.35±2.04	1.52	10	61.63±2.22	1.48
Diabetes	OMOPSO	2	49.70±1.38	8.28	4	54.58±1.27	3.38	6	59.92±1.26	2.04	8	62.97±1.28	1.20	10	60.95±1.42	0.77
	NSGA-II	2	49.82±1.39	7.61	4	55.47±1.22	3.36	6	60.42±1.16	1.80	8	63.20±1.26	0.48	10	61.73±1.45	0.77
WDBC	OMOPSO	2	68.10±1.38	2.43	4	75.39±1.08	2.00	6	82.60±1.04	1.53	8	88.23±0.86	0.98	10	91.65±0.50	0.55
	NSGA-II	2	68.89±1.29	2.34	4	74.52±1.14	2.02	6	82.69±1.03	1.53	8	87.96±1.09	0.97	10	91.30±0.55	0.55
WPBC	OMOPSO	2	39.39±2.48	8.97	4	43.85±2.43	7.67	6	50.59±2.95	4.51	8	54.71±3.35	2.63	10	59.09±3.52	1.25
	NSGA-II	2	40.49±2.52	8.77	4	44.11±2.57	7.54	6	50.08±2.81	4.77	8	55.97±3.30	2.39	10	59.51±3.52	1.12
IRIS	OMOPSO	3	88.07±0.55	7.26	4	95.33±0.51	3.18	5	93.03±0.60	3.81	6	94.14±0.60	3.33	7	93.26±0.58	3.32
	NSGA-II	3	89.03±0.54	7.02	4	95.48±0.53	3.17	5	93.85±0.59	3.08	6	95.40±0.58	2.79	7	93.92±0.62	3.69
Hays	OMOPSO	3	46.25±0.56	36.08	6	52.92±0.85	32.74	9	57.29±1.03	30.12	12	62.02±1.07	26.12	15	68.27±1.27	22.56
	NSGA-II	3	47.08±0.60	36.13	6	52.78±0.85	33.16	9	58.47±1.01	29.14	12	62.15±0.93	26.04	15	69.45±1.18	22.34
Seeds	OMOPSO	3	59.00±0.42	27.66	6	68.00±0.55	22.94	9	75.00±0.51	18.46	12	83.00±0.43	12.65	15	87.00±0.41	8.58
	NSGA-II	3	59.00±0.41	27.72	6	67.00±0.46	22.71	9	75.00±0.55	18.32	12	83.00±0.44	12.46	15	87.00±0.36	8.81
Tae	OMOPSO	3	32.20±0.34	66.11	6	36.40±0.46	60.35	9	42.90±0.70	51.10	12	50.80±1.15	37.83	15	55.60±1.04	34.28
	NSGA-II	3	32.90±0.30	65.58	6	36.20±0.38	60.79	9	42.50±0.74	51.06	12	50.80±1.12	38.03	15	55.40±1.01	34.61
Thyroid	OMOPSO	3	83.46±0.27	15.65	6	88.06±0.13	12.28	9	93.17±0.18	7.48	12	92.78±0.33	6.09	15	95.49±0.14	5.58
	NSGA-II	3	84.57±0.03	15.12	6	88.29±0.27	12.46	9	93.56±0.30	7.04	12	94.02±0.32	5.61	15	96.19±0.11	4.73
Glass	OMOPSO	6	53.04±0.29	36.30	8	56.23±0.27	31.57	10	62.07±0.38	23.33	12	67.06±0.40	16.77	14	74.92±0.32	8.45
	NSGA-II	6	53.36±0.25	36.10	8	56.23±0.27	32.23	10	61.60±0.50	22.22	12	68.77±0.30	14.77	14	75.08±0.29	8.38
Ecoli	OMOPSO	6	50.78±0.56	35.67	8	54.95±0.54	31.93	10	59.75±0.59	28.23	12	63.58±0.58	24.34	14	67.48±0.56	23.27
	NSGA-II	6	51.32±0.59	35.51	8	55.36±0.60	31.57	10	60.64±0.56	27.31	12	64.36±0.59	24.13	14	67.90±0.64	22.99

Table 5.19: (Accuracy + Cost) based results for all datasets using different number of ellipses

Table 5.20 compares the best results obtained by the ECSDT when considering both accuracy and cost as a combined multi-objective function along with those obtained by the comparative algorithms. These results are presented graphically in Figures 5.16, 5.17, 5.18 and 5.19. Figure 5.16 and Figure 5.17 presents the cost comparison. Figure 5.18 presents the accuracy comparison and Figure 5.19 presents a comparison of the decision tree sizes.

Dataset	C-S-C + J48					C-S-C + NBTree					MetaCost + J48				
	Size	Acc	SE	Cost	SE	Size	Acc	SE	Cost	SE	Size	Acc	SE	Cost	SE
Bupa	51	54.38	1.07	13.40	3.27	11	50.03	0.83	0.53	0.63	7.4	51.27	0.83	0.56	0.60
Hepatitis	2.6	64.49	3.14	0.60	0.70	5	58.72	3.12	0.50	0.80	3	60.64	3.16	0.60	0.80
Heart	39	57.33	0.75	11.50	3.09	17	57.37	0.99	0.80	0.65	8.2	55.07	0.92	0.52	0.93
Haberman	5	50.13	2.46	1.04	0.55	3	50.00	2.48	0.50	0.59	3	50.33	2.44	0.50	0.59
Diabetes	39	54.04	1.24	6.95	1.18	1	54.50	1.38	0.53	0.20	9.8	54.29	1.47	0.51	0.30
WDBC	25	88.31	1.79	1.63	0.30	23	86.70	1.05	1.07	0.26	22.3	90.65	0.43	0.48	0.85
WPBC	11.5	58.67	3.73	4.37	1.31	8.5	53.62	4.31	0.85	0.21	12.8	56.90	3.62	1.45	0.61
IRIS	8.11	91.41	0.95	3.95	0.14	9	90.74	0.65	3.98	0.12	6.44	91.85	0.75	3.84	1.26
Hays	23	81.67	0.61	16.95	2.30	9.67	60.42	0.67	33.99	4.28	21.3	80.10	0.49	20.84	3.44
Seeds	15	87.44	1.05	6.32	2.74	8	85.18	0.96	7.10	2.96	14	85.98	0.42	8.17	2.89
Tae	67	56.00	0.56	41.97	4.33	7	50.90	1.22	42.87	4.42	42	52.10	0.93	43.41	4.89
Thyroid	17	92.09	0.10	8.10	2.87	7	92.63	0.17	6.57	1.40	15	90.39	0.17	9.11	2.13
Glass	11	96.73	0.0	9.06	3.24	7	70.40	2.63	12.09	3.21	11	95.40	0.26	5.39	1.91
Ecoli	41	66.52	1.96	37.45	4.54	13	55.65	2.14	28.74	3.16	32	65.07	1.87	27.85	3.05
Overall Average	**25.37**	**71.37**	**1.39**	**11.66**	**0.0**	**9.23**	**65.49**	**1.61**	**10.01**	**0.0**	**14.87**	**70.0**	**1.27**	**8.8**	**0.0**

Dataset	MetaCost + NBTree					ECSDT+OMOPSO					ECSDT+NSGA-II				
	Size	Acc	SE	Cost	SE	Size	Acc	SE	Cost	SE	Size	Acc	SE	Cost	SE
Bupa	1	50.12	0.83	0.51	0.65	21	59.92	0.63	1.41	0.75	21	61.09	0.65	1.37	0.79
Hepatitis	2.4	59.23	3.2	0.50	0.90	21	82.04	0.72	4.43	0.10	21	82.56	0.61	4.43	1.10
Heart	3	53.40	0.89	0.51	0.94	21	69.84	0.57	1.61	0.84	21	69.58	0.61	1.54	0.91
Haberman	1	50.00	2.48	0.50	0.59	21	61.21	2.16	1.48	0.61	21	61.63	2.22	1.48	0.62
Diabetes	2	52.47	1.47	0.49	0.30	21	62.97	1.28	1.20	0.41	17	63.20	1.26	0.48	0.26
WDBC	16.3	89.07	0.33	0.53	0.80	21	91.65	0.50	0.55	0.38	21	91.30	0.55	0.55	0.38
WPBC	6.17	52.69	4.37	0.61	0.45	21	59.09	3.52	1.25	0.81	21	59.51	3.52	1.12	0.94
IRIS	7.67	87.63	0.87	5.11	1.00	9	95.33	0.18	3.17	1.92	13	95.47	0.20	3.17	0.39
Hays	1.33	56.88	1.19	35.73	2.55	31	68.27	1.27	22.56	3.05	31	69.45	1.18	22.34	3.43
Seeds	7	81.06	1.19	8.91	2.15	31	87.00	0.41	8.58	2.48	31	87.00	0.36	8.81	2.25
Tae	8.33	46.80	1.24	45.94	4.35	31	55.60	1.04	34.28	4.79	31	55.40	1.01	34.61	3.28
Thyroid	8	90.85	0.17	9.79	1.81	31	95.49	0.14	5.58	1.38	31	96.19	0.11	4.73	1.24
Glass	9	64.49	3.45	13.66	3.63	29	74.92	0.32	8.45	2.85	29	75.08	0.29	8.38	2.92
Ecoli	13	53.05	2.78	25.61	3.29	29	67.48	2.02	23.27	3.63	29	67.90	1.93	22.99	3.91
Overall Average	**6.16**	**63.41**	**1.75**	**10.6**		**24.14**	**73.38**	**1.09**	**8.47**		**24.14**	**73.88**	**1.05**	**8.29**	**0.0**

Table 5.20: (Accuracy + Cost) based results for all datasets when applying different algorithms

From the results shown in the Table 5.20 and from the charts shown in the Figures 5.20, 5.21, 5.22 and 5.23, we can summarise the following findings:

- ECSDT was able to achieve higher accuracy than other algorithms on 10 of the 14 datasets.
- ECSDT achieved lower costs than other algorithms on 5 of the 14 datasets.
- In general, the performance of the two optimization methods of ECSDT is similar, both optimization methods able to achieve better overall average accuracy and cost. More specifically, ECSDT obtained an overall average accuracy of 73.38% and an overall average cost of 8.47 when utilizing OMOPSO and an overall average accuracy of 73.88% with an overall average cost of 8.29 when utilizing the NSGA-II. In contrast, the highest overall average accuracy obtained by other algorithms was 71.37%, which was achieved by CostSensitiveClassifier with J48, and the lowest average cost recorded by other algorithms was 8.8, which was achieved by MetaCost with J48.
- Similar to what happened with the previous aspect, the improvements by ECSDT, noted above, however, are at the expense of larger trees except that when applying (C-S-C + J48) which recorded the worst general average of the tree sizes with an average size of 25.37 nodes. Where the ECSDT was able to produce smaller decision trees than when applying (C-S-C + J48) on 5 out of the 10 datasets with which the ECSDT algorithm produced the best accuracy results.

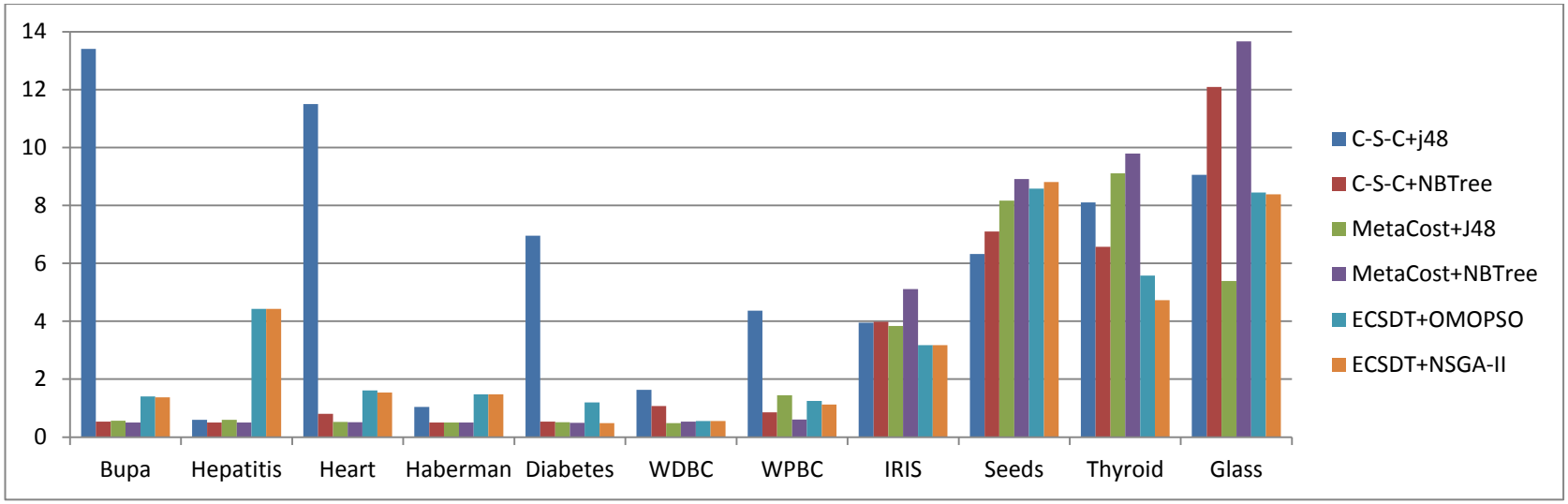

Figure 5.20: Cost comparison (Accuracy + Cost) based aspect

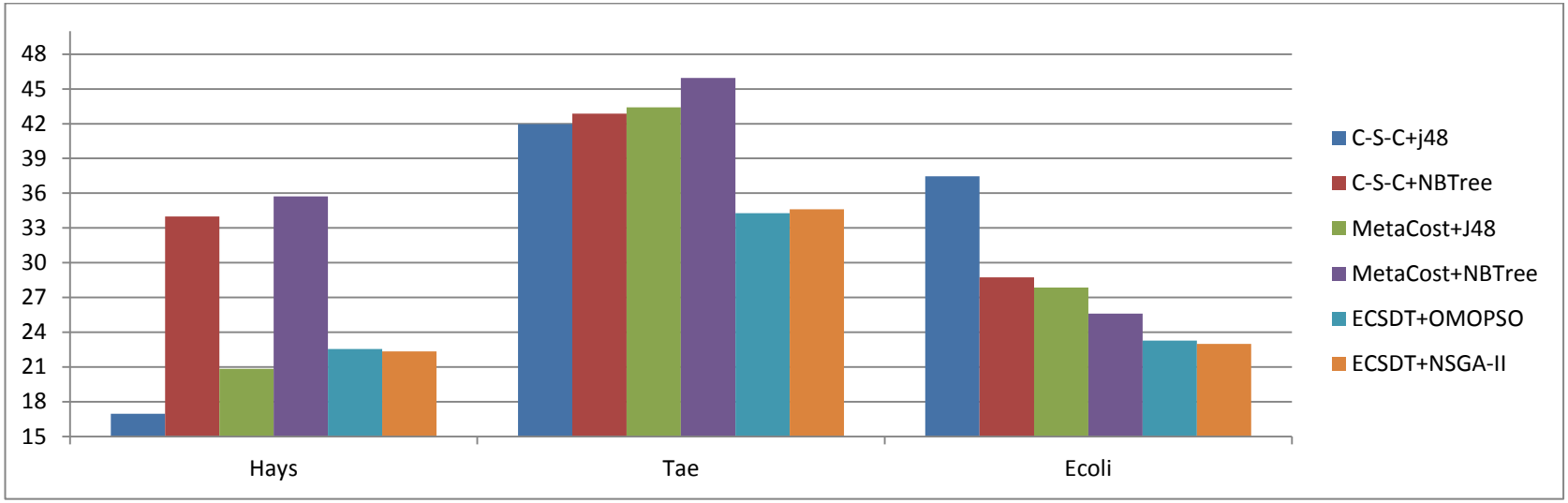

Figure 5.21: Cost comparison (Accuracy + Cost) based aspect

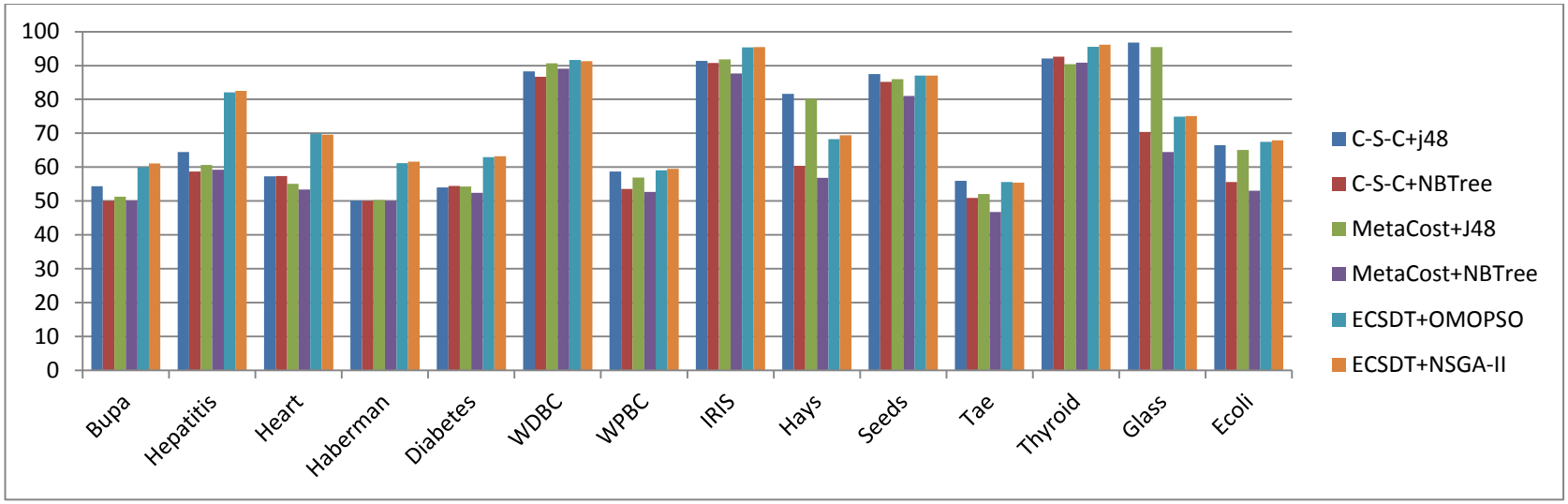

Figure 5.22: Accuracy comparison (Accuracy + Cost) based aspect

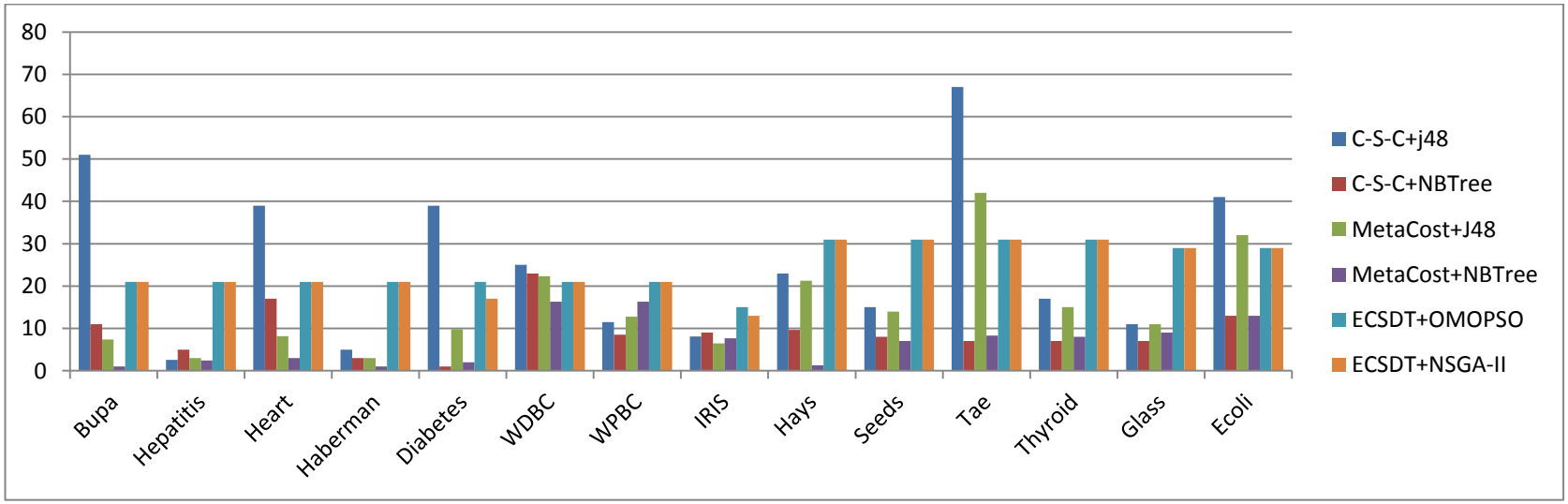

Figure 5.23: DT-size comparison (Accuracy + Cost) based aspect

5.3.3.2 Accuracy + Cost based results for the Bupa dataset

Tables 5.21 and 5.22 below present the results when ECSDT is applied to the Bupa dataset considering both accuracy and cost for building, testing and evaluating the classifier. The tables show that utilizing 10 ellipses in both optimization methods produced the lowest average costs as well the highest accuracy for all the 10 used cost ratios. Therefore, the associated results of using 10 ellipses in both methods were compared with those obtained by other algorithms as shown in Table 5.23.

	ECSDT + OMOPSO														
	2 ell			4 ell			6 ell			8 ell			10 ell		
	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE
Ratio - 1	0.74	44.1	0.72	0.59	48.7	0.61	0.63	49.1	0.56	0.60	50.7	0.54	0.56	54.8	0.66
Ratio - 2	1.14	42.6	0.56	0.79	46.7	0.73	0.82	45.9	0.53	0.81	47.2	0.51	0.65	49.6	0.58
Ratio - 3	1.43	42.9	0.58	1.39	47.2	0.68	0.80	48.4	0.64	0.79	50.1	0.48	0.76	52.5	0.52
Ratio - 4	4.92	41.4	0.58	3.45	44.6	0.55	3.42	47.2	0.68	3.40	49.6	0.59	1.95	49.9	0.83
Ratio - 5	12.14	43.8	0.70	9.21	48.1	0.58	6.32	47.0	0.56	6.28	51.0	0.60	3.39	50.4	0.67
Ratio - 6	1.09	51.3	0.64	0.55	65.8	0.53	0.54	67.0	0.52	0.44	68.7	0.64	0.40	70.4	0.75
Ratio - 7	2.74	53.3	0.62	0.79	63.8	0.55	0.78	64.3	0.52	0.62	66.7	0.59	0.45	69.0	0.49
Ratio - 8	5.05	54.2	0.59	1.25	61.4	0.63	0.94	63.8	0.58	0.92	64.3	0.57	0.91	66.7	0.64
Ratio - 9	26.51	52.8	0.56	7.59	64.6	0.50	4.69	64.9	0.65	2.94	66.4	0.49	1.76	68.4	0.56
Ratio - 10	38.10	54.8	0.78	14.82	65.5	0.47	9.03	66.1	0.71	6.15	63.8	0.55	3.22	67.5	0.59
Average	**9.39**	**48.12**	**0.63**	**4.04**	**55.64**	**0.58**	**2.8**	**56.37**	**0.6**	**2.3**	**57.85**	**0.56**	**1.41**	**59.92**	**0.63**

Table 5.21: Bupa (Accuracy + Cost) based results for each number of ellipses (ECSDT+OMOPSO)

	ECSDT + NSGA-II														
	2 ell			4 ell			6 ell			8ell			10 ell		
	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE
Ratio - 1	0.73	45.2	0.68	0.64	48.5	0.53	0.59	53.2	0.57	0.60	53.0	0.66	0.57	55.7	0.63
Ratio - 2	1.31	41.8	0.61	0.82	46.1	0.55	1.08	48.5	0.51	0.66	47.6	0.46	0.63	51.3	0.67
Ratio - 3	1.42	43.8	0.53	0.79	49.3	0.64	0.81	47.2	0.66	0.79	49.6	0.57	0.76	52.8	0.58
Ratio - 4	7.81	42.9	0.58	4.87	47.0	0.63	4.83	50.4	0.64	4.85	49.0	0.55	1.93	51.3	0.55
Ratio - 5	12.14	43.8	0.53	9.21	47.8	0.72	9.20	48.7	0.70	6.29	50.1	0.63	3.37	52.5	0.72
Ratio - 6	0.85	54.5	0.63	0.58	65.5	0.47	0.52	63.8	0.63	0.46	69.3	0.60	0.39	71.3	0.81
Ratio - 7	2.15	55.7	0.67	0.78	64.9	0.61	0.66	62.9	0.56	0.46	68.4	0.57	0.43	70.7	0.66
Ratio - 8	4.76	54.5	0.80	0.93	64.1	0.66	0.94	63.8	0.51	0.92	65.5	0.54	0.60	68.4	0.70
Ratio - 9	22.12	58.0	0.77	4.69	65.2	0.50	6.15	63.7	0.59	1.76	68.4	0.49	1.74	70.4	0.64
Ratio - 10	29.41	55.1	0.52	9.03	65.2	0.58	11.92	65.8	0.72	3.24	65.8	0.70	3.23	66.7	0.56
Average	**8.27**	**49.53**	**0.63**	**3.23**	**56.36**	**0.59**	**3.67**	**56.8**	**0.61**	**2.00**	**58.67**	**0.58**	**1.37**	**61.09**	**0.65**

Table 5.22: Bupa (Accuracy + Cost) based results for each number of ellipses (ECSDT+NSGA-II)

Algorithm	Bupa (Accuracy + Cost) based Results								
	Ratio - 1			Ratio - 2			Ratio - 3		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	46.67	1.08	51	43.77	2.55	51	43.8	4.58	51
MetaCost+J48	50.14	0.81	37	42.32	0.72	1	42.6	0.57	1
C.S.C+NBTree	42.32	0.66	11	42.32	0.58	11	42.3	0.58	11
MetaCost+NBTree	43.19	0.57	1	42.03	0.58	1	42.00	0.58	1
ECSDT-PSO-10ell	54.80	0.56	21	49.60	0.65	21	52.50	0.76	21
ECSDT-GA-10ell	55.70	0.57	21	51.30	0.63	21	52.80	0.76	21

Algorithm	Ratio - 4			Ratio - 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	43.77	20.81	51	43.77	41.10	51	66.96	0.72	51
MetaCost+J48	42.03	0.58	1	42.03	0.58	1	61.74	0.49	23
C.S.C+NBTree	42.03	0.58	11	42.03	0.58	11	57.68	0.48	11
MetaCost+NBTree	42.03	0.58	1	42.03	0.58	1	57.97	0.42	1
ECSDT-PSO-10ell	49.90	1.95	21	50.40	3.39	21	70.40	0.40	21
ECSDT-GA-10ell	51.30	1.93	21	52.50	3.37	21	71.30	0.39	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	63.80	2.21	51	63.77	4.09	51	63.77	19.17	51
MetaCost+J48	58.00	0.56	7	57.97	0.42	1	57.97	0.42	1
C.S.C+NBTree	57.70	0.57	11	57.97	0.42	11	57.97	0.42	11
MetaCost+NBTree	58.00	0.42	1	57.97	0.42	1	57.97	0.42	1
ECSDT-PSO-10ell	69.00	0.45	21	66.70	0.91	21	68.40	1.76	21
ECSDT-GA-10ell	70.70	0.43	21	68.40	0.60	21	70.40	1.74	21

Algorithm	Ratio - 10			Averages		
	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	63.77	38.01	51	54.38	13.43	51
MetaCost+J48	57.97	0.42	1	51.27	0.56	7.4
C.S.C+NBTree	57.97	0.42	11	50.02	0.53	11
MetaCost+NBTree	57.97	0.42	1	50.11	0.50	1
ECSDT-PSO-10ell	67.50	3.22	21	59.92	1.405	21
ECSDT-GA-10ell	66.70	3.23	21	61.09	1.365	21

Table 5.23: Bupa (Accuracy + Cost) based results obtained by the comparative algorithms

Similar to what happened previously with the cost-based aspect, Table 5.23 shows that the CostSensitiveClassifier with J48 (C.S.C+J48) on the Bupa dataset results in very high costs compared to other algorithms, so that it has been excluded from the cost comparison chart presented in Figure 5.24. The results listed above in Table 5.23 are graphically illustrated in Figures 5.24, 5.25 and 5.26 for cost, accuracy and tree sizes respectively.

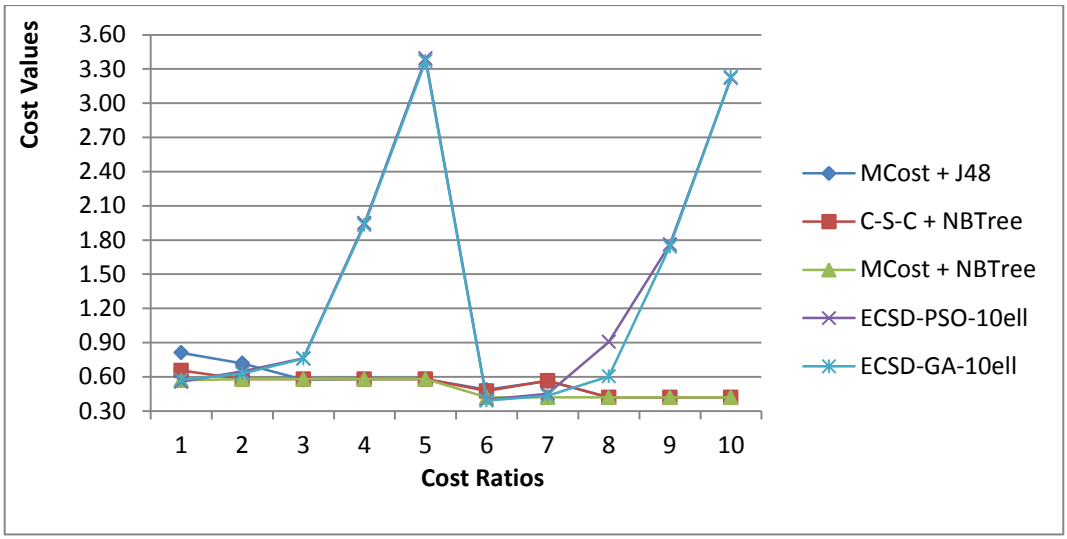

Figure 5.24: Bupa cost comparison for the (Accuracy + Cost) based aspect

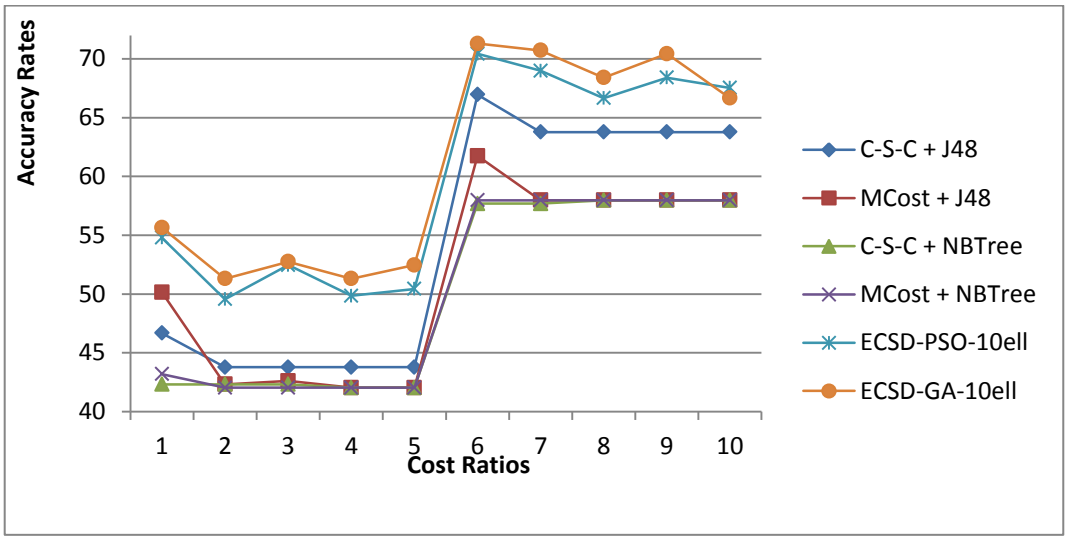

Figure 5.25: Bupa accuracy comparison for the (Accuracy + Cost) based aspect

From Table 5.23 and Figures 5.24, 5.25 and 5.26 we can draw the following:

- ECSDT obtains higher accuracies compared to all other algorithms with all cost ratios, irrespective of whether GAs or PSO optimization methods are used.
- ECSDT with both optimization methods (PSO and GA) obtained lower costs than the use of CostSensitiveClassifier with J48 with all cost ratios.
- The reason for not achieving lower costs can be attributed to the fact that the cost has been included in the objective function only as a penalty for accuracy and not as a main objective of the optimizations process.

- The performance of ECSDT is slightly better when using GAs than when using PSO for optimization.
- ECSDT failed to produce smaller trees than other algorithms except with the (C.S.C+J48) that produced the largest trees compared to all other algorithms.
- MetaCost + NBTree produced the smallest trees in most cases, and that is due to the same fact that has been mentioned earlier that MetaCost sometimes neglects the less-costly class and places all examples into the more-costly class.

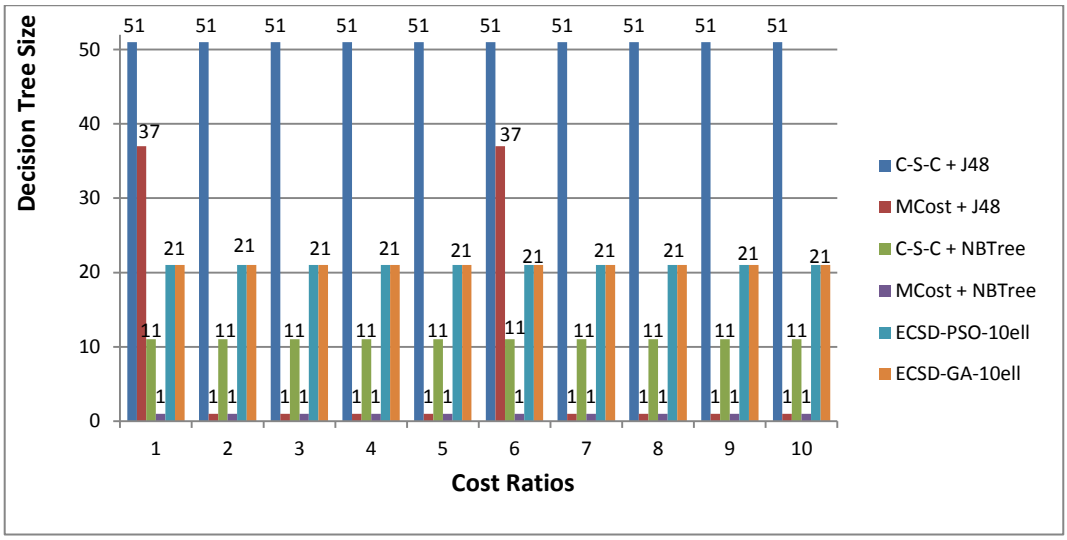

Figure 5.26: Bupa DT-size comparison for the (Accuracy + Cost) based aspect

5.3.3.3 Accuracy + Cost based results for the IRIS dataset

Table 5.24 and Table 5.25 below present the results obtained for the IRIS dataset when the (Accuracy + Cost) aspect is considered. The tables show the accuracy along with the associated misclassification costs when ECSDT is used with a different number of ellipses, each with different cost ratios and with both optimization methods (OMOPSO and NSGA-II).

	ECSDT + OMOPSO														
	3 ell			4 ell			5 ell			6 ell			7 ell		
	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE
Ratio - 1	2.80	90.00	0.56	1.53	96.66	0.55	1.80	94.00	0.66	1.87	93.33	0.63	1.80	94.00	0.61
Ratio - 2	0.69	90.67	0.57	0.28	96.00	0.56	0.87	91.33	0.64	0.47	94.66	0.67	0.88	90.00	0.52
Ratio - 3	2.25	72.66	0.55	1.40	92.00	0.48	1.46	86.00	0.58	0.73	92.66	0.58	1.45	87.33	0.52
Ratio - 4	0.73	86.66	0.49	0.37	95.33	0.62	0.37	95.33	0.58	0.57	90.66	0.53	0.40	94.00	0.56
Ratio - 5	0.19	91.33	0.63	0.14	96.66	0.42	0.19	91.33	0.72	0.17	94.00	0.73	0.17	93.33	0.58
Ratio - 6	0.33	91.33	0.66	0.19	93.33	0.4	0.24	94.00	0.57	0.17	94.66	0.49	0.19	93.33	0.77
Ratio - 7	23.33	89.33	0.52	9.00	96.00	0.58	13.33	94.00	0.55	7.67	96.66	0.52	9.00	96.00	0.48
Ratio - 8	18.33	90.00	0.52	6.00	96.66	0.45	9.00	94.66	0.49	6.00	96.66	0.57	8.00	95.33	0.55
Ratio - 9	16.67	90.66	0.47	9.67	95.33	0.51	7.00	96.66	0.61	12.33	94.00	0.64	8.00	96.00	0.65
Average	**7.26**	**88.07**	**0.55**	**3.18**	**95.33**	**0.51**	**3.81**	**93.03**	**0.6**	**3.33**	**94.14**	**0.6**	**3.32**	**93.26**	**0.58**

Table 5.24: IRIS (Accuracy + Cost) based results for each number of ellipses (ECSDT+OMOPSO)

	ECSDT + NSGA-II														
	3 ell			4 ell			5 ell			6 ell			7 ell		
	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE
Ratio - 1	2.53	92.66	0.58	1.53	96.66	0.54	1.87	93.33	0.64	1.53	96.66	0.59	1.80	94.00	0.55
Ratio - 2	0.61	92.66	0.58	0.21	96.66	0.54	0.73	92.66	0.66	0.28	96.00	0.53	0.61	92.66	0.53
Ratio - 3	2.23	74.66	0.46	1.41	91.33	0.77	1.43	88.66	0.57	0.73	92.66	0.47	0.76	90.00	0.61
Ratio - 4	0.73	86.66	0.44	0.33	96.00	0.34	0.27	96.66	0.51	0.27	94.66	0.75	0.33	95.33	0.67
Ratio - 5	0.19	91.33	0.61	0.14	96.66	0.83	0.17	93.33	0.62	0.16	94.66	0.71	0.16	94.66	0.81
Ratio - 6	0.26	92.00	0.68	0.24	94.00	0.42	0.21	91.33	0.58	0.17	94.66	0.54	0.19	93.33	0.62
Ratio - 7	24.00	89.33	0.49	10.67	95.33	0.63	9.00	96.00	0.55	9.00	96.00	0.58	11.67	94.66	0.53
Ratio - 8	16.00	91.33	0.52	6.00	96.66	0.34	7.00	96.00	0.57	6.00	96.66	0.55	8.00	95.33	0.57
Ratio - 9	16.67	90.66	0.54	8.00	96.00	0.38	7.00	96.66	0.60	7.00	96.66	0.49	9.67	95.33	0.66
Average	**7.02**	**89.03**	**0.54**	**3.17**	**95.48**	**0.53**	**3.08**	**93.85**	**0.59**	**2.79**	**95.4**	**0.58**	**3.69**	**93.92**	**0.62**

Table 5.25: IRIS (Accuracy + Cost) based results for each number of ellipses (ECSDT+NSGA-II)

From the tables, it can be observed that when using OMOPSO, utilizing 4 ellipses produced the lowest average costs per example, while when using NSGA-II, 6 ellipses produced the lowest average costs per example. Therefore, the results associated with the use of 4 ellipses with the OMOPSO optimization methods and the results associated with the use of 6 ellipses with the NSGA-II optimization methods were used in the comparison with the results of other algorithms. Table 5.26 summarizes the results and Figures 5.27 to 5.29 present the comparison in terms of cost, accuracy and size of trees.

When the (Accuracy + Cost) method is adopted and applied to the IRIS dataset, the following points are observed:

- ECSDT was able to obtain higher accuracies compared to all other algorithms with all cost ratios.

- The performance of OMOPSO and NSGA-II was very close, where they perform equally in 4 of the 9 cost ratios, the NSGA-II was better with 3 cost ratios whereas the OMOPSO was better with 2 cost ratios.
- ECSDT was able to achieve the best overall averages of both accuracy and cost with both optimization methods. ECSDT recorded with NSGA-II an average accuracy of 95.40% associated with an average cost per example of 2.79, and recorded with OMOPSO an average accuracy of 95.33% associated with an average cost of 3.18, while the highest average accuracy recorded by other algorithms was 91.85% that is obtained by MetaCost+J48 and the lowest average cost recorded by other algorithms was 3.83.
- ECSDT with the use of OMOPSO achieved higher accuracies than all other algorithms and satisfactory costs in about 7 of the 9 cost ratios, and this was achieved with trees of similar or even smaller size to the other methods.

Algorithm	IRIS (Accuracy + Cost) based Results								
	Ratio - 1			Ratio - 2			Ratio - 3		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	96.00	1.54	9	96.00	0.82	9	69.33	1.03	9
MetaCost+J48	93.33	1.75	4	94.00	0.84	7	74.00	0.38	3
C.S.C+NBTree	84.00	2.44	9	80.67	0.55	9	84.67	0.15	9
MetaCost+NBTree	84.00	2.62	11	85.33	0.37	9	68.67	1.21	5
ECSDT-PSO-4ell	96.66	1.53	9	96.00	0.28	9	92.00	1.40	9
ECSDT-GA-6ell	96.66	1.53	13	96.00	0.28	13	92.66	0.73	13

Algorithm	Ratio - 4			Ratio - 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	93.33	0.41	11	94.67	0.14	5	90.00	0.31	7
MetaCost+J48	94.67	0.28	7	93.33	0.18	7	93.33	0.09	9
C.S.C+NBTree	94.67	0.40	9	95.33	0.13	9	93.33	0.13	9
MetaCost+NBTree	91.33	0.51	9	90.67	0.15	1	90.00	0.19	9
ECSDT-PSO-4ell	95.33	0.37	9	96.66	0.14	9	93.33	0.19	9
ECSDT-GA-6ell	94.66	0.27	13	94.66	0.16	13	94.66	0.17	13

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	96.00	8.33	7	92.67	12.67	7	94.66	10.33	9
MetaCost+J48	94.67	11.33	7	95.33	8.33	7	94.00	11.33	7
C.S.C+NBTree	94.67	12.00	9	94.67	9.33	9	94.66	10.67	9
MetaCost+NBTree	92.67	16.33	9	92.67	12.67	9	93.33	12.00	7
ECSDT-PSO-4ell	96.00	9.00	9	96.66	6.00	9	95.33	9.67	9
ECSDT-GA-6ell	96.00	9.00	13	96.66	6.00	13	96.66	7.00	13

Algorithm	Averages		
	Acc	Cost	Size
C.S.C+J48	91.41	3.95	8.11
MetaCost+J48	91.85	3.83	6.44
C.S.C+NBTree	90.74	3.98	9.00
MetaCost+NBTree	87.63	5.11	7.67
ECSDT-PSO-4ell	95.33	3.18	9.00
ECSDT-GA-6ell	95.40	2.79	13.00

Table 5.26: IRIS (Accuracy + Cost) based results obtained by the comparative algorithms

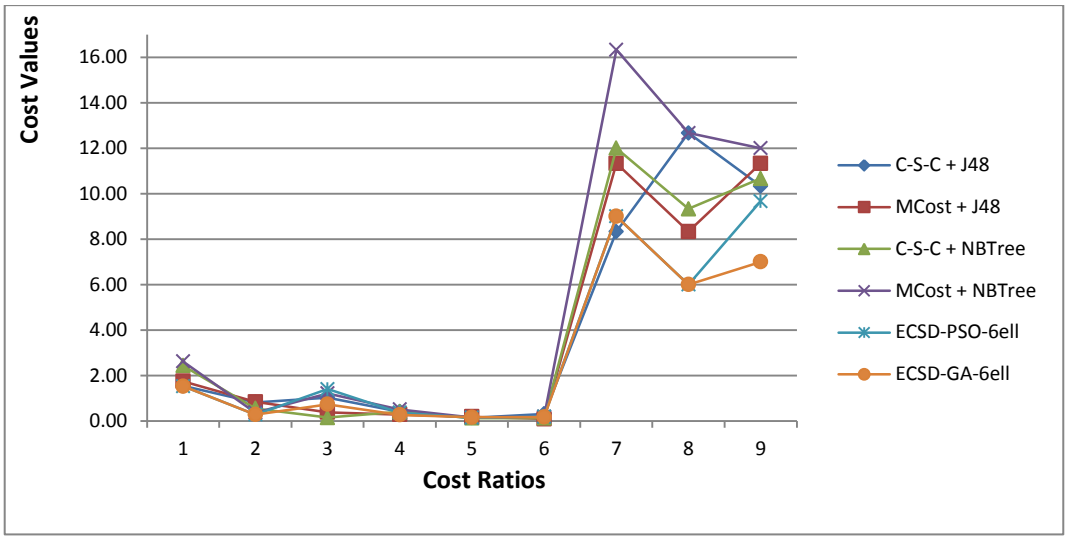

Figure 5.27: IRIS cost comparison for the (Accuracy + Cost) based aspect

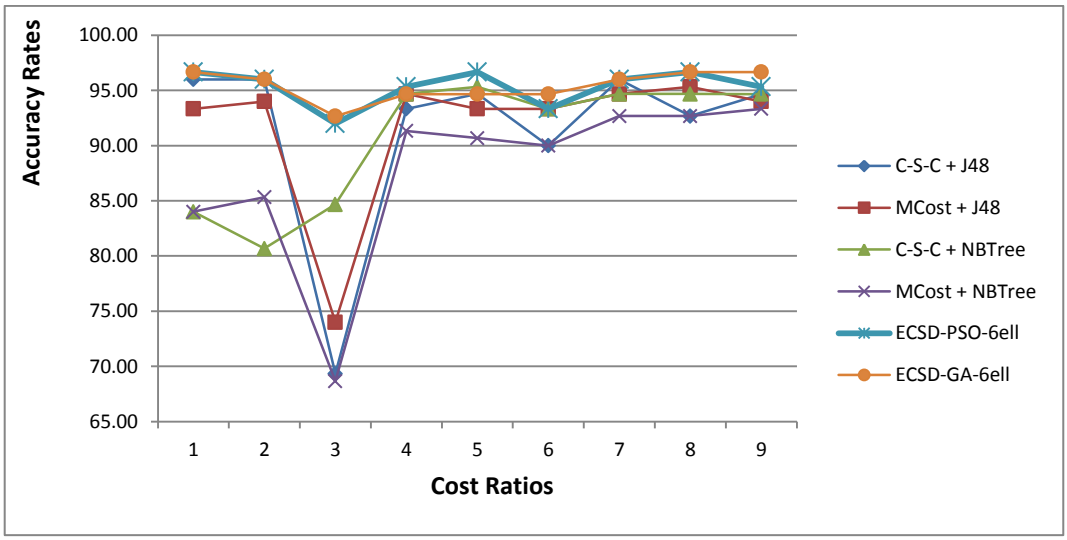

Figure 5.28: IRIS accuracy comparison for the (Accuracy + Cost) based aspect

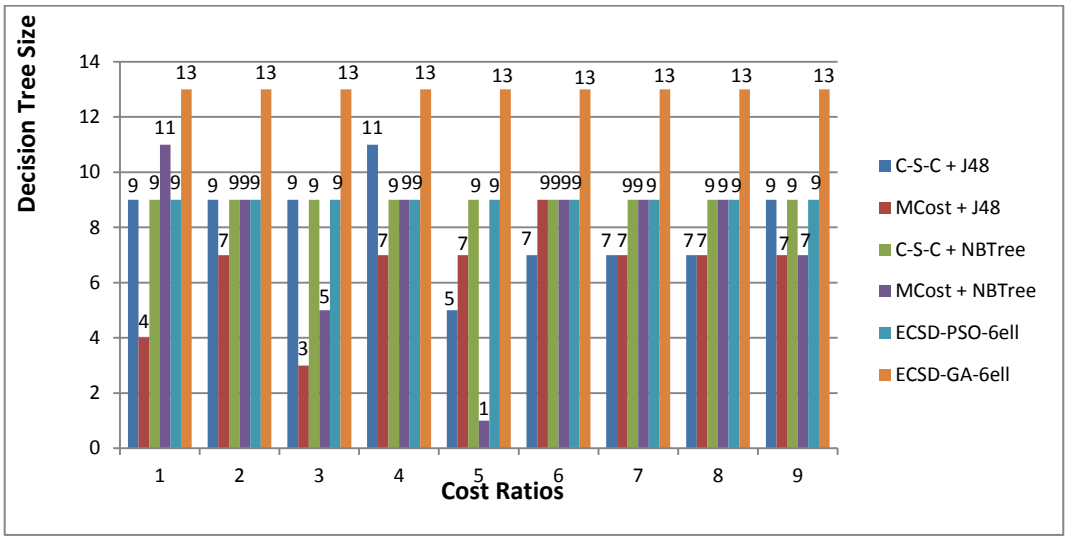

Figure 5.29: IRIS DT-size comparison for the (Accuracy + Cost) based aspect

5.3.3.4 Accuracy + Cost based results for the Ecoli dataset

Table 5.27 and Table 5.28 below present the results obtained for Ecoli dataset when the (Accuracy + Cost) aspect is considered. The tables show the accuracy along with the associated misclassification costs when ECSDT is used with a different number of ellipses, each with different cost ratios and the two optimization methods (OMOPSO and NSGA-II).

	ECSDT + OMOPSO														
	6 ell			8 ell			10 ell			12ell			14 ell		
	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE
Ratio - 1	28.32	23.21	0.56	23.50	25.89	0.48	22.27	28.57	0.53	19.31	26.79	0.58	17.19	31.25	0.56
Ratio - 2	40.47	55.65	0.58	35.38	59.23	0.56	31.79	62.80	0.62	29.52	66.67	0.47	27.46	64.88	0.52
Ratio - 3	52.18	66.67	0.53	45.93	70.24	0.48	40.10	73.81	0.62	32.74	75.89	0.62	38.76	81.24	0.47
Ratio - 4	31.11	64.88	0.49	27.40	68.15	0.55	22.71	74.70	0.59	19.74	77.38	0.53	18.98	80.95	0.62
Ratio - 5	24.98	57.74	0.53	22.15	62.20	0.60	18.59	67.56	0.72	15.41	73.81	0.66	15.15	77.68	0.58
Ratio - 6	48.76	50.6	0.66	46.06	55.95	0.56	43.66	60.42	0.57	36.93	66.67	0.69	30.25	72.62	0.51
Ratio - 7	37.63	45.83	0.52	34.18	51.19	0.51	29.34	56.85	0.56	25.61	61.90	0.48	21.26	66.37	0.53
Ratio - 8	21.90	41.67	0.62	20.81	46.73	0.54	17.36	53.27	0.53	15.43	59.52	0.57	17.14	64.88	0.67
Average	**35.67**	**50.78**	**0.56**	**31.93**	**54.95**	**0.54**	**28.23**	**59.75**	**0.59**	**24.34**	**63.58**	**0.58**	**23.27**	**67.48**	**0.56**

Table 5.27: Ecoli (Accuracy + Cost) based results for each number of ellipses (ECSDT+OMOPSO)

	ECSDT + NSGA-II														
	6 ell			8 ell			10 ell			12ell			14 ell		
	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE	Cost	Acc	SE
Ratio - 1	27.35	24.70	0.55	22.91	26.49	0.50	22.47	26.79	0.64	17.68	30.06	0.59	15.73	33.63	0.59
Ratio - 2	40.01	57.44	0.56	34.64	59.52	0.54	30.72	64.58	0.66	30.11	67.26	0.66	27.74	66.37	0.48
Ratio - 3	53.07	65.48	0.56	45.04	70.83	0.70	38.90	75.30	0.49	32.83	74.40	0.57	37.63	80.36	0.56
Ratio - 4	30.97	66.67	0.64	27.57	68.45	0.63	23.01	74.40	0.67	19.51	78.87	0.53	18.89	81.85	0.67
Ratio - 5	24.53	59.63	0.54	21.34	63.99	0.56	18.95	66.37	0.52	15.34	74.70	0.49	15.24	76.79	0.80
Ratio - 6	48.32	51.49	0.68	45.96	57.44	0.62	37.92	63.69	0.56	36.49	67.26	0.67	30.21	74.40	0.62
Ratio - 7	37.84	44.94	0.49	34.18	51.19	0.59	29.23	58.33	0.45	25.49	63.69	0.66	21.31	65.77	0.73
Ratio - 8	22.01	40.18	0.67	20.95	44.94	0.65	17.26	55.65	0.48	15.59	58.63	0.52	17.18	63.99	0.65
Average	**35.51**	**51.32**	**0.59**	**31.57**	**55.36**	**0.6**	**27.31**	**60.64**	**0.56**	**24.13**	**64.36**	**0.59**	**22.99**	**67.9**	**0.64**

Table 5.28: Ecoli (Accuracy + Cost) based results for each number of ellipses (ECSDT+NSGA-II)

The tables show that increasing the number of ellipses has a positive effect in reducing the cost with the majority of the used cost ratios, so the use of 14 ellipses with both optimization methods produced the lowest average costs per example. Therefore, the associated results of using 14 ellipses with both optimization methods are used in the comparison with the results obtained by other algorithms as shown in Table 5.29.

Algorithm	Ecoli (Accuracy + Cost) based Results								
	Ratio - 1			Ratio - 2			Ratio - 3		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	31.84	31.19	41	66.07	41.95	41	78.27	57.06	41
MetaCost+J48	32.14	19.55	21	65.17	40.94	31	78.86	33.52	17
C.S.C+NBTree	26.48	20.07	13	63.98	31.42	13	74.70	44.68	13
MetaCost+NBTree	25.00	19.19	9	61.01	34.99	23	76.78	37.24	13
ECSDT-PSO-12ell	31.25	17.19	29	64.88	27.46	29	81.24	38.76	29
ECSDT-GA-14ell	33.63	15.73	29	66.37	27.74	29	80.36	37.63	29

Algorithm	Ratio - 4			Ratio - 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	77.67	32.95	41	75.29	22.48	41	75.29	50.99	41
MetaCost+J48	78.27	20.02	29	74.40	17.81	25	74.40	42.50	41
C.S.C+NBTree	72.61	28.77	13	66.66	24.49	13	66.66	37.28	13
MetaCost+NBTree	72.61	28.80	13	65.17	21.83	5	65.17	35.16	9
ECSDT-PSO-12ell	80.95	18.98	29	77.68	15.15	29	72.62	30.25	29
ECSDT-GA-14ell	81.85	18.89	29	76.79	15.24	29	74.40	30.21	29

Algorithm	Ratio - 7			Ratio - 8			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	69.34	40.40	41	57.44	22.56	41	66.40	37.45	41.00
MetaCost+J48	62.50	31.65	43	60.71	16.78	45	65.81	27.85	31.50
C.S.C+NBTree	47.32	24.06	13	38.392	19.14	13	57.11	28.74	13.00
MetaCost+NBTree	50.59	27.60	11	14.583	32.12	19	53.87	29.62	12.75
ECSDT-PSO-12ell	66.37	21.26	29	64.88	17.14	29	67.48	23.27	29.00
ECSDT-GA-14ell	65.77	21.31	29	63.99	17.18	29	67.90	22.99	29.00

Table 5.29: Ecoli (Accuracy + Cost) based results obtained by the comparative algorithms

From the previous table and using Figures 5.30, 5.31 and 5.32 we can get the following conclusions for the Ecoli related (Accuracy+cost) based results:

- ECSDT was able to obtain lowest costs than other algorithms in 6 of the 8 cost ratios.
- ECSDT was also able to obtain higher accuracies than other algorithms in 6 of the 8 cost ratios.
- The performance of both OMOPSO and NSGA-II was similar; each achieved the highest accuracy with 3 cost ratios and the lowest cost with 3 cost ratios.
- In general, ECSDT with both optimization methods achieved the highest averages of accuracy that reached 67.48% for OMOPSO and 67.90% for NSGA-II and also achieved the lowest averages of costs with both optimization methods that recorded 23.27% for OMOPSO and 22.99% for NSGA-II. These achievements in terms of accuracy and cost were achieved with trees, which on average were smaller than those produced by C.S.C+J48 and MetaCost+J48.

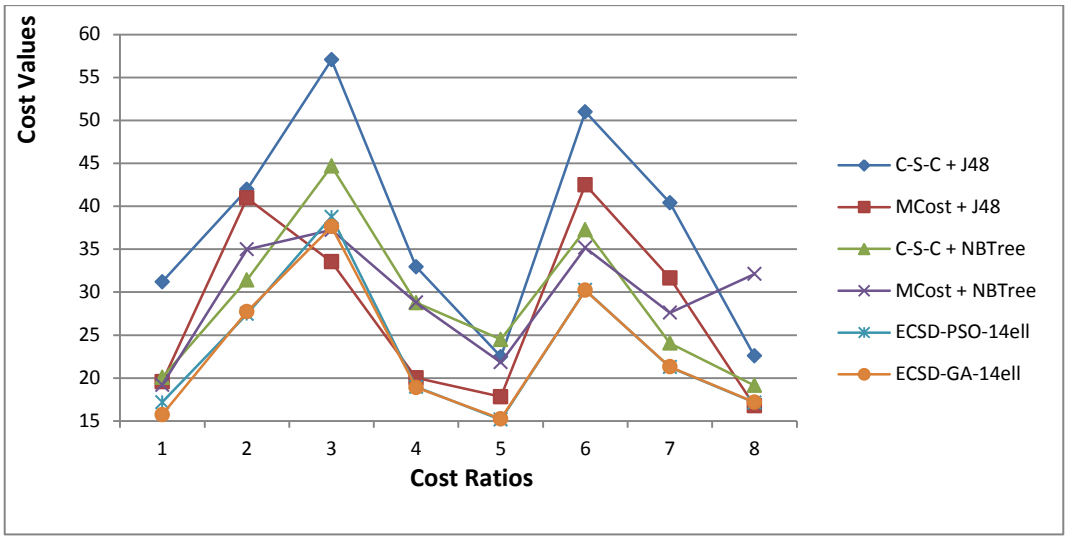

Figure 5.30: Ecoli cost comparison for the (Accuracy + Cost) based aspect

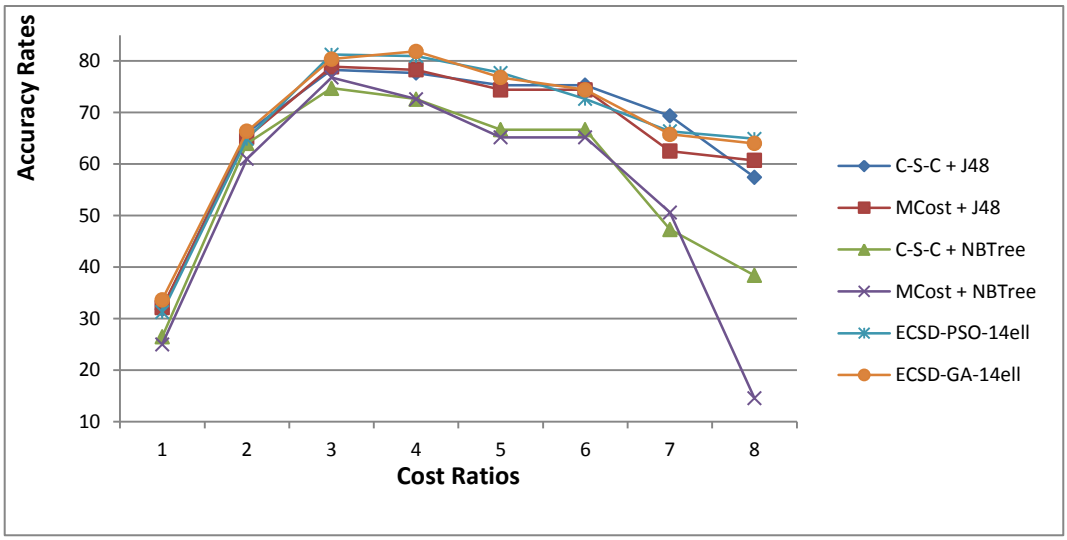

Figure 5.31: Ecoli accuracy comparison for the (Accuracy + Cost) based aspect

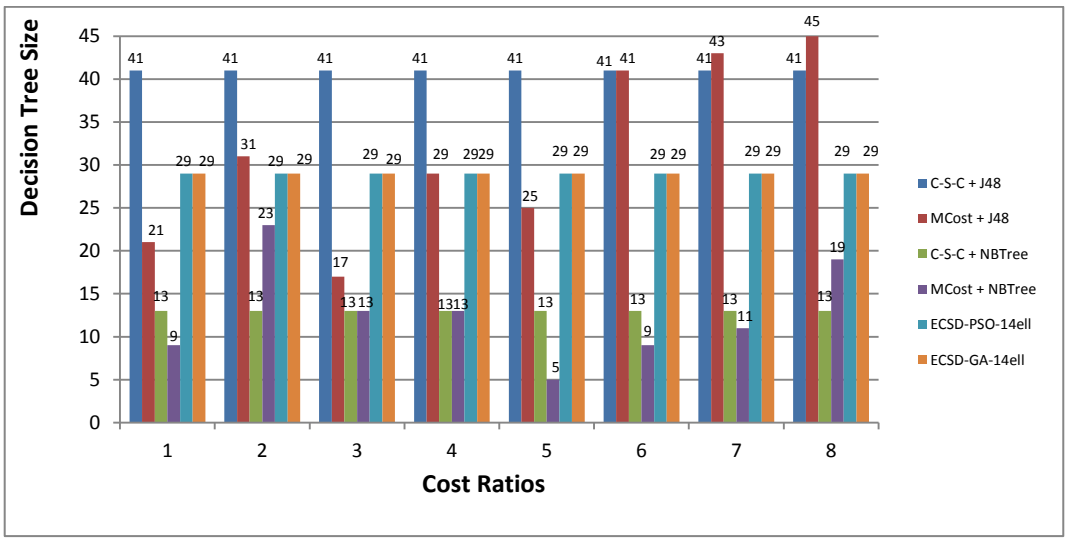

Figure 5.32: Ecoli DT-size comparison for the (Accuracy + Cost) based aspect

Chapter 6 : CONCLUSION AND FUTURE WORK

The area of learning cost-sensitive multiclass decision trees continues to be one of the major challenging areas of data mining and machine learning. Cost-sensitive multiclass learning aims to build classifiers that minimize the expected misclassification costs for multiclass problems.

Decision trees are one of the most common and widely used classification methods for cost-sensitive learning, due to the simplicity of constructing them, their transparency and comprehensibility. Most of the cost-sensitive algorithms that have been developed during the last decade are aimed at solving binary classification problems where examples are classified into one of two available classes.

A review of the literature shows that research on inducing nonlinear multiclass cost-sensitive decision trees is still in its early stages and further research may result in improvements over the current state of the art. Hence, this thesis has explored the following question:

> ***How can non-linear regions be identified for multiclass problems and utilized to construct decision trees so as to maximize the accuracy of classification, and minimize costs?***

Hence, the main goal of this research was to develop a new algorithm called the Elliptical Cost-Sensitive Decision Tree algorithm (ECSDT) that induces elliptical cost-sensitive decision trees for multiclass classification problems using two different evolutionary optimization methods: OMOPSO which is a particle swarm optimization method (PSO) and NSGA-II which is a genetic algorithm (GA).

In Chapter 1, a set of objectives that would help find a solution to the above question were listed. Section 6.1 revisits these objectives, summarizes the main findings and presents to what extent these objectives have been accomplished and Section 6.2 presents directions for future work.

6.1 Evaluating the Achievement of the Objectives

This section reviews the research objectives, summarizes the main findings and shows how well the research objectives have been accomplished by presenting each research objective separately and discussing the extent to which it has been achieved.

1- For the literature review, firstly a study was conducted on a range of topics such as supervised learning, experimental methods and decision tree learning. This was followed by a deep survey of the field of cost-sensitive classification which included different cost-sensitive learning theories and strategies as well some ways of categorizing cost-sensitive learning methods such as direct cost-sensitive learning methods and meta-learning methods. The literature review covered four main areas related to the research which are cost-sensitive decision tree learning, nonlinear decision trees, multi-class classification and multi-objective optimization methods. The literature review revealed that there wasn't much research on developing non-linear cost-sensitive decision tree learning algorithms for multi-class problems.

2- For the development and the implementation of the proposed algorithm, the ECSDT algorithm has been developed for learning cost-sensitive classifiers for multiclass problems. ECSDT uses ellipses for splitting the data and the main task is to find a suitable number of ellipses and place them in the instances space so that some measures such as misclassification error and cost are minimized. To minimise these measures, two different optimisation methods, namely genetic algorithms (GAs) and particle swarm optimization (PSO) were explored. The ECSDT algorithm was implemented in a framework known as MOEA, which provides implementations of PSO (OMOPSO) and GAs (NSGAII).

3- For training, testing and evaluating the accuracy and the cost-sensitivity of ECSDT, 14 different real-world datasets from the UCI Machine Learning Repository (Lichman 2013) named (Iris, Seeds, Glass, Hepatitis, Bupa, Heart, Diabetes, Haberman, Ecoli, Hayes, Tae, Thyroid, WDBC and WPBC) have been used. The results obtained when applying ECSDT have been compared against some common cost-sensitive decision tree methods available in the Weka system such as the J48, NBTree, MetaCost and CostSensitiveClassifier.

4- ECSDT was evaluated in three different settings, each with a different objective function: the first setting considered the accuracy of classification only, the second

setting considered misclassification costs only, while the third considered accuracy and cost together.

5- To determine the appropriate number of ellipses that gives the best results compared to the other alternatives, all experiments were repeated five times with a different number of ellipses for each individual experiment and then the number of ellipses that gives the best results is selected.

6- To evaluate performance, ECSDT and several well-known algorithms were applied on 14 data sets and the results compared.

7- For the first approach that considers accuracy only, 6 accuracy-based algorithms from the Weka system were used for the comparison, namely J48, NBTree, BFTree, ADTree, LADTree and REPTree.

8- For the second and third approaches that consider misclassification costs, two cost-sensitive meta-classifiers, CostSensitiveClassifier and MetaCost with the use of two decision tree base learners named J48 and NBTree were used.

9- Based on the empirical evaluations, we can conclude that:

   (a) Applying ECSDT with the first approach that takes into account accuracy only, ECSDT has achieved its goal of achieving higher accuracy to a reasonable extent as it was able to obtain the highest accuracy on 10 out of the 14 datasets and also was able to produce smaller trees when compared with J48, LADTree and ADTree.

   (b) Applying ECSDT with the second approach that takes into account cost only, ECSDT achieved its goal of achieving lower costs to a reasonable extent as it was able to record the lowest cost on 10 out of the 14 datasets, but that was at the expense of larger trees than those produced by other algorithms.

   (c) When ECSDT considers cost and accuracy, it achieved higher accuracy on 10 out of the 14 datasets. However, for cost results, ECSDT was able to record the lowest cost on only 5 out of the 14 datasets, and also that was at the expense of larger trees than those produced by other algorithms.

Although ECSDT performs well on many of the datasets, there are some where its performance is not as good as existing algorithms. The reasons why machine learning algorithms work for some datasets and others vary and attempts to resolve this issue have led

to studies such as meta-learning (Shilbayeh & Vadera 2014). The most likely reasons for ECSDT not performing well on some of the datasets can be attributed to some factors such as:

- The nature of the dataset, where in some cases the classes overlap in a way which makes it very difficult to separate using ellipses as the case with Hays and Tae datasets.
- Some datasets are unbalanced such that there are very few examples belonging to some classes but many belonging to other classes. An example of that is the Glass dataset where the number of examples in each of the 7 classes is 70, 76, 17, 0, 13, 9, and 29.
- When there are many features, with several that may be irrelevant. The ECSDT optimization process (GAs, PSO) uses a fixed number of cycles which may not converge to selecting the optimal features in cases where there are many features as the case with the WDBC dataset.
- As mentioned previously, an important factor that plays a vital role in improving the performance of the algorithm is determining the optimal number of ellipses to be used for inducing the decision tree. When visualizing the datasets in Weka it was clear that with some datasets, increasing the number of ellipses might have improved performance.

Some of the above issues, such as the effect of imbalanced data and having many features also affect other algorithms. There are methods, such as sampling and feature selection methods that can be adopted to improve performance (Chandrashekar & Sahin 2014). Further work, specifically on developing a method for deciding the most ellipses and integration of feature selection methods in ECSDT could lead to improved versions of ECSDT in the future.

6.2 Future Work

Despite the satisfactory results achieved by the ECSDT algorithm, there are some limitations and some ideas that could be explored in the future:

- ECSDT deals with only numeric data but does not deal with nominal data. To use ECSDT at present, the nominal values are converted to numeric values. A future

implementation that codes nominal values might reduce search time and improve results.

- In this research, only misclassification costs were considered and the other cost types such as test costs were not considered. Turney (1995) mentioned that there is a trade-off between the cost of misclassification errors and the cost of tests in classification learning. For example, in medical diagnoses, blood tests or x-rays may help in reducing the misclassification costs, but when the cost of conducting the required tests are much more costly than the costs of misclassification errors, then it is obvious that there is no point in conducting the tests. So, in the future, this trade-off between the cost of misclassification errors and the cost of tests could be examined by including the different cost tests associated with each classification problem in the objective function.
- The cost-sensitivity performance of ECSDT algorithm was evaluated by comparing its results with those obtained by two other meta-learning algorithms: MetaCost and CostSensitiveClassifier. In the future, a comparison will be extended to include other cost-sensitive algorithms such as ICET (Turney 1995) and CSNL (Vadera 2010).
- As mentioned in Section 5.3, one of the challenges facing ECSDT is how to determine the appropriate number of ellipses that should be used to build the decision tree for each classification problem. The methodology followed in this research was to determine the appropriate number of ellipses based on the number of classes. Therefore, in the future, we suggest adopting a specific methodology by which to determine the appropriate number of ellipses. One of the ideas that can be explored is by increasing the number of ellipses by 1 in each turn until a specific condition is achieved such as reaching a specific high accuracy rate, recording certain low-cost value or a particular maximum number of ellipses are reached.
- In this research, 14 diverse datasets were used which vary in the number of classes, number of features, number of examples, etc. However, to make the research more comprehensive and more diverse, we recommend applying the algorithm to more real-world datasets that have more of examples, more features and also have a diversity of data types (numeric, nominal, discrete, continuous, real, integer,…etc.).
- ECSDT takes advantage of two optimization methods, OMOPSO and NSGA-II that are available in MOEA framework. These optimization methods are controlled by several parameters, ECSDT uses these methods with only the default values. Some

examples of these properties are populationSize, mutationProbability, maxEvaluations and archiveSize, and the effect of changing these properties on the performance of ECSDT was not examined and could be explored in the future.

In conclusion, the main hypothesis explored in this thesis is that using evolutionary optimization methods such as GAs and PSO to induce elliptical boundaries provides a basis for a new cost-sensitive learning algorithm for multiclass problems and that such an algorithm will perform better than existing algorithms in terms of classification accuracy, cost minimisation, and decision tree size. The primary contribution of this study is, therefore, the development of a novel cost-sensitive decision tree algorithm called the Elliptical Cost-Sensitive Decision Tree algorithm (ECSDT) that induces cost-sensitive elliptical decision trees for multiclass classification problems using two different evolutionary optimization methods called OMOPSO which is a particle swarm optimization method and NSGA-II which is a genetic algorithm. The algorithm achieved promising results in terms of reducing the cost of classification errors as well as maintaining good rates of classification accuracy compared to the other algorithms.

REFERENCES

Abe, N., Zadrozny, B. & Langford, J., 2004. An Iterative Method for Multi-class Cost-sensitive Learning. *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 1, pp.3–11.

Abe, S. & Inoue, T., 2001. Fuzzy Support Vector Machines for Pattern Classification. *Proceedings. IJCNN'01. International Joint Conference*, 2, pp.1449–1454.

Alvarez, S.A., 2002. An exact analytical relation among recall, precision, and classification accuracy in information retrieval.'Boston College. *Boston, Technical Report BCCS-02-01*, pp.1–22.

Aly, M., 2005. Survey on Multiclass Classification Methods Extensible algorithms. *Neural Networks*, 19, pp.1–9.

Anguita, D. et al., 2012. The " K " in K-fold Cross Validation. *ESANN 2012 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pp.25–27.

Ankerst, M. et al., 1999. Visual classification: an interactive approach to decision tree construction. *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, 1, pp.392–396.

Ardjani, F., Sadouni, K. & Benyettou, M., 2010. Optimization of SVM multiclass by particle swarm (PSO-SVM). *2010 2nd International Workshop on Database Technology and Applications, DBTA2010 - Proceedings*, (December), pp.32–38.

Arlot, S. & Celisse, A., 2010. A survey of cross-validation procedures for model selection. *Statistics surveys*, 4, pp.40–79.

Bai, Q., 2010. Analysis of Particle Swarm Optimization Algorithm. *Computer and Information Science*, 3(1), p.180.

Ballester, P.J. & Mitchell, J.B.O., 2010. A machine learning approach to predicting protein-ligand binding affinity with applications to molecular docking. *Bioinformatics (Oxford, England)*, 26(9), pp.1169–1175.

Bandyopadhyay, S. & Saha, S., 2013. *Unsupervised Classification*, Berlin, Heidelberg: Springer Berlin Heidelberg.

Barbara, F., 1977. Concept learning and the recognition and classification of exemplars. *Journal of Verbal Learning and Verbal Behavior*, 16(3), pp.321–338.

Barros, R.C. et al., 2012. A Survey of Evolutionary Algorithms for Decision-Tree Induction. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(3), pp.291–312.

Bhuvaneswari, C., 2015. *Application of Evolutionary Algorithms for Multi-objective Optimization in VLSI and Embedded Systems* M. C. Bhuvaneswari, ed., New Delhi: Springer India.

Board, R. & Pitt, L., 1989. Semi-Supervised Learning. *Machine Learning*, 4(1), pp.41–65.

Bobrowski, L. & Kretowski, M., 2000. Induction of Multivariate Decision Trees by Using Dipolar Criteria. In *Lecture Notes in Computer Science <D>*. Springer, pp. 331–336.

Borrego, M., Douglas, E.P. & Amelink, C.T., 2009. Quantitative, Qualitative, and Mixed Research Methods in Engineering Education. *Journal of Engineering Education*, 98(1), pp.53–66.

Breiman et al., 1984. *Classification and Regression Trees*,

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

Carlos A Coello, Dhaenens, C. & Jourdan, L., 2010. *Advances in Multi-Objective Nature Inspired Computing* C. A. Coello Coello, C. Dhaenens, & L. Jourdan, eds., Berlin, Heidelberg: Springer Berlin Heidelberg.

Celikyilmaz, A. et al., 2009. Increasing accuracy of two-class pattern recognition with enhanced fuzzy functions. *Expert Systems with Applications*, 36(2), pp.1337–1354.

Celisse, A. & Robin, S., 2008. A leave-p-out based estimation of the proportion of null hypotheses. *Arxiv preprint arXiv08041189*, p.21.

Chandrashekar, G. & Sahin, F., 2014. A survey on feature selection methods. *Computers and Electrical Engineering*, 40(1), pp.16–28.

Chandrupatla, T.R. & Osier, T.J., 2010. The perimeter of an ellipse. *The Mathematical Scientist*, 35(2), pp.122–131.

Charytanowicz, M. et al., 2010. Complete Gradient Clustering Algorithm for Features Analysis of X-Ray Images. In *Advances in Intelligent and Soft Computing*. pp. 15–24.

Chen, K.H. et al., 2014. Applying particle swarm optimization-based decision tree classifier for cancer classification on gene expression data. *Applied Soft Computing Journal*, 24, pp.773–780.

Chen, N. et al., 2011. A genetic algorithm-based approach to cost-sensitive bankruptcy prediction. *Expert Systems with Applications*, 38(10), pp.12939–12945.

Cho, Y.-J., Lee, H.-S. & Jun, C.-H., 2011. Optimization of Decision Tree for Classification Using a Particle Swarm. *Industrial Engineering and Management Systems*, 10(4), pp.272–278.

Coello, C.A., Dhaenens, C. & Laetitia, J., 2010. *Advances in Multi-Objective Nature Inspired Computing* C. A. Coello Coello, C. Dhaenens, & L. Jourdan, eds., Berlin, Heidelberg: Springer Berlin Heidelberg.

Coello, C. a. & Reyes-Sierra, M., 2006. Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art. *International Journal of Computational Intelligence Research*, 2(3), pp.287–308.

Coello, C. a C., Lamont, G.B. & Veldhuizen, D. a Van, 2007. *Evolutionary Algorithms for Solving Multi-Objective Problems*, Boston, MA: Springer US.

Costa, E. et al., 2007. A review of performance evaluation measures for hierarchical classifiers. *Evaluation Methods for Machine …*, 5(12), pp.815–824.

Dai, G., Yeung, D. & Chang, H., 2006. Extending Kernel Fisher Discriminant Analysis with the Weighted Pairwise Chernoff Criterion. *ECCV 2006*, IV, pp.308–320.

Dai, J. et al., 2016. Discrete particle swarm optimization approach for cost sensitive attribute reduction. *Knowledge-Based Systems*, 102, pp.116–126.

Dash, Manoranjan and Liu, H., 1997. Feature selection for classification. *Intelligent data analysis*, 1(3), pp.131--156.

David Hadka, 2014. *MOEA Framework, User Guide* 2.1.,

Davis, J. & Goadrich, M., 2006. The Relationship Between Precision-Recall and ROC Curves. *Proceedings of the 23rd International Conference on Machine learning -- ICML'06*, pp.233–240.

Deb, K. et al., 2005. *Evolutionary Multiobjective Optimization* A. Abraham, L. Jain, & R. Goldberg, eds., London: Springer-Verlag.

Devasena, L. & Hyderabad, I.S.B., 2015. Proficiency Comparison Ofladtree and Reptree Classifiers for Credit. *International Journal on Computational Sciences & Applications (IJCSA)*, 5(1), pp.39–50.

Diaconis, P. & Efron, B., 1983. Computer-Intensive Methods in Statistics. *Scientific American*, 248(5), pp.116–131.

Dietterich, T.G. & Bakiri, G., 1995. Solving Multiclass Learning Problems via Error-Correcting Output Codes. *Jouranal of Artifical Intelligence Research*, 2, pp.263–286.

Domingos, P., 1999. MetaCost: A General Method for Making Classifiers Cost-Sensitive. *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, 55, pp.155–164.

Drummond, C. & Holte, R.C., 2000. Exploiting the Cost(In)sensitivity of Decisions Tree Splitting Criteria. *International Conference on Machine Learning*, 1(1), pp.239–246.

Duan, W. & Ding, C., 2016. Non-linear Cost-sensitive Decision Tree for Multi-classification. *International Journal of Software Engineering and Its Applications*, 10(2), pp.217–228.

Eberhart, R. & Kennedy, J., 1995. A new optimizer using particle swarm theory. *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pp.39–43.

Elkan, C., 2001. The Foundations of Cost-Sensitive Learning. *Proceedings of International Joint Conference on Artificial Intelligence*, 17(1), pp.973–978.

Fawcett, T., 2006. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8), pp.861–874.

Fei, B. & Liu, J., 2006. Binary tree of SVM: A new fast multiclass training and classification algorithm. *IEEE Transactions on Neural Networks*, 17(3), pp.696–704.

Fisher, R.A., 1936. The Use of Multiple Measurements in Taxonomic Problems. *Annals of eugenics*, 2(7), pp.179–188.

Frank, E., Hall, M.A. & Witten, I.H., 2016. *Data Mining: Practical Machine Learning Tools and Techniques* Fourth Edi., Todd Green.

Freitas, A., Brazdil, P. & Costa-Pereira, A., 2009. Cost-Sensitive Learning in Medicine. In *Data Mining and Medical Knowledge Management: Cases and Applications*. pp. 57–75.

Freund, Y. & Mason, L., 1999. The alternating decision tree learning algorithm. *ICML '99 Proceedings of the Sixteenth International Conference on Machine Learning*, 99, pp.124–133.

Friedman, J.H., 1996. *Another approach to polychotomous classification*, Stanford.

Friedman, J.H., 2002. Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4), pp.367–378.

Fu, Z., Robles-Kelly, A. & Zhou, J., 2010. Mixing linear SVMs for nonlinear classification. *IEEE Transactions on Neural Networks*, 21, pp.1963–1975.

Fulkerson, B. et al., 1995. Machine Learning, Neural and Statistical Classification. *Technometrics*, 37(4), p.459.

Fumera, G. & Roli, F., 2002. Cost-sensitive Learning in Support Vector Machines. *Workshop on Machine Learning, 8th Meeting of the Italian Association of Artificial Intelligence*, (1).

Fung, G.M., Mangasarian, O.L. & Smola, A.J., 2002. Minimal Kernel Classifiers. *Journal of Machine Learning Research*, 3, pp.303–321.

Fürnkranz, J., 1997. Pruning methods for rule learning algorithms. *Machine Learning*, 27(2), pp.139–171.

Godínez, A.C., Espinosa, L.E.M. & Montes, E.M., 2010. An experimental comparison of multiobjective algorithms: NSGA-II and OMOPSO. *Proceedings - 2010 IEEE Electronics, Robotics and Automotive Mechanics Conference, CERMA 2010*, 1(1), pp.28–33.

Grefenstette, J.J., 1986. Optimization of Control Parameters for Genetic Algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 16(1), pp.122–128.

Gupta, D. & Ghose, U., 2015. A comparative study of classification algorithms for forecasting rainfall. *In 2015 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO)(Trends and Future Directions)*, pp.1–6.

Hadka, D., 2014. MOEA Framework User Guide.

Hassan, R. & Cohanim, B., 2005. A comparison of particle swarm optimization and the genetic algorithm. *1st AIAA multidisciplinary design optimization specialist*

conference, pp.1–13.

Holmes, G., Pfahringer, B. & Kirkby, R., 2002. Multiclass Alternating Decision Trees. *Proceedings of the 13th European Conference on Machine Learning (ECML '02)*, 2430, pp.161–172.

Hormann, A.M., 1962. Programs for machine learning Part I. *Information and Control*, 5(4), pp.347–367.

Hsu, C. & Lin, C., 2002. A comparison of methods for multiclass support vector machines. *Neural Networks, IEEE Transactions on*, 13(2), pp.415–425.

Ittner, A. & Schlosser, M., 1996. Non-linear decision trees-NDT. In *In Proceedings of the Thirteenth International Conference on International Conference on Machine Learning (ICML)*. Morgan Kaufmann Publishers, pp. 252–257.

J.R. Quinlan, 1987. Simplifying Decision Trees. *International journal of man-machine studies*, 27(2), pp.221–234.

Jin Li, Xiaoli Li & Xin Yao, 2005. Cost-Sensitive Classification with Genetic Programming. In *2005 IEEE Congress on Evolutionary Computation*. IEEE, pp. 2114–2121.

Kale, S., Kumar, R. & Vassilvitskii, S., 2011. Cross-validation and mean-square stability. *In Proceedings of the Second Symposium on Innovations in Computer Science (ICS2011)*.

Kalmegh, S., 2015. Analysis of WEKA Data Mining Algorithm REPTree , Simple Cart and RandomTree for Classification of Indian News. *International Journal of Innovative Science, Engineering & Technology*, 2(2), pp.438–446.

Kalmegh, S.R., 2015. Comparative Analysis of WEKA Data Mining Algorithm RandomForest, RandomTree and LADTree for Classification of Indigenous News Data. *International Journal of Emerging Technology and Advanced Engineering*, 5(1), pp.507–517.

Kaur, G. & Chhabra, A., 2014. Improved J48 Classification Algorithm for the Prediction of Diabetes. *International Journal of Computer Applications*, 98(22), pp.13–17.

Kennedy, J. & Eberhart, R., 1995. Particle swarm optimization. *Neural Networks, 1995. Proceedings., IEEE International Conference on*, 4, pp.1942–1948 vol.4.

Kohavi, R., 1995. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. *IJCAI International Joint Conference on Artificial Intelligence*, 14(2), pp.1137–1145.

Kohavi, R., 1996. Scaling Up the Accuracy of Naive-Bayes Classi ers : a Decision-Tree Hybrid. In *KDD-96 Proceedings*.

Konak, A., Coit, D.W. & Smith, A.E., 2006. Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering and System Safety*, 91(9), pp.992–1007.

Korn, G.A. & Korn, T.M., 2000. Conic Sections. In *Mathematical handbook for scientists and engineers: Definitions, theorems, and formulas for reference and review*. pp. 946–1008.

Kothari, C., 2009. *Research Methodology: methods and techniques.* Second Edi., New Age International.

Kotsiantis, S.B., 2007. Supervised Machine Learning : A Review of Classification Techniques. *Informatica, An International Journal of Computing and Informatics*, 3176(31), pp.249–268.

Kr, M. & Grze's, M., 2007. Evolutionary Induction of Decision Trees for Misclassification Cost Minimization. In *Adaptive and Natural Computing Algorithms*. Springer Berlin Heidelberg New York, pp. 1–10.

Kraska, M., 2010. Quantitative Research. In *Encyclopedia of Research Design*. 2455 Teller Road, Thousand Oaks California 91320 United States: SAGE Publications, Inc., pp. 1167–1172.

Kukar, M. & Kononenko, I., 1998. Cost-sensitive learning with neural networks. *13th European Conference on Artificial Intelligence*, pp.445–449.

Kumar, S., Ghosh, J. & Crawford, M.M., 2002. Hierarchical Fusion of Multiple Classifiers for Hyperspectral Data Analysis. *Pattern Analysis {&} Applications*, 5(2), pp.210–220.

Kumar, V., 2014. Feature Selection: A literature Review. *The Smart Computing Review*, 4(3), pp.211–229.

Lichman, M., 2013. Machine Learning Repository. *University of California; Irvine; School of Information and Computer Sciences*.

Ling, C.X. et al., 2004. Decision trees with minimal costs. *Twenty-first international conference on Machine learning - ICML '04*, (Icml), p.69.

Ling, C.X. et al., 2006. Maximum Profit Mining and Its Application in Software Development. *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp.929–934.

Ling, C.X. & Sheng, V.S., 2008. Cost-Sensitive Learning and the Class Imbalance Problem. *Encyclopedia of Machine Learning*, pp.231–235.

Lomax, S. & Vadera, S., 2011. An empirical comparison of cost-sensitive decision tree induction algorithms. *Expert Systems*, 28(3), pp.227–268.

Von Lücken, C., Barán, B. & Brizuela, C., 2014. A survey on multi-objective evolutionary algorithms for many-objective problems. *Computational Optimization and Applications*, 58(3), pp.707–756.

Madzarov, G., Gjorgjevikj, D. & Chorbev, I., 2009. A Multi-class SVM Classifier Utilizing Binary Decision Tree Support vector machines for pattern recognition. *Informatica*, 33, pp.233–241.

Manda, K., Satapathy, S.C. & Poornasatyanarayana, B., 2012. Population based meta-heuristic techniques for solving optimization problems : A selective survey. *International Journal of Emerging Technology and Advanced Engineering*, 2(11), pp.206–211.

Masnadi-Shirazi, H. & Vasconcelos, N., 2010. Risk minimization, probability elicitation, and cost-sensitive SVMs. In *In Proceedings of the 27th International Conference on Machine Learning, 2010*. pp. 759–766.

McDermott, J. & Forsyth, R.S., 2016. Diagnosing a disorder in a classification benchmark. *Pattern Recognition Letters*, 73, pp.41–43.

Mehra, N. & Gupta, S., 2013. Survey on Multiclass Classification Methods. *International Journal of Computer Science and Information Technologies*, 4(4), pp.572–576.

Mika, S. et al., 1999. Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing IX: Proceedings of the 1999 IEEE Signal Processing Society Workshop (Cat. No.98TH8468)*. pp. 41–48.

Min-Ling Zhang & Zhou, Z.-H., 2005. A k-Nearest Neighbor Based Algorithm for Multi-label Classification. *Granular Computing, 2005 IEEE International Conference*, 2, pp.718–721.

Mitchell, T.M., 1999. Machine Learning and Data Mining. *Communications of the ACM*, 41(11), pp.30–36.

Mudry, A. & Tjellström, A., 2011. Historical background of bone conduction hearing devices and bone conduction hearing aids. *Advances in Oto-Rhino-Laryngology*, 71, pp.1–9.

Muni, D.P., Pal, N.R. & Das, J., 2006. Genetic Programming for Simultaneous Feature

Selection and Classifier Design. *IEEE Transactions On Systems, Man, And Cybernetics—Part B: Cybernetics*, 36(1), pp.106–117.

Nashnush, E. & Vadera, S., 2017. Learning cost-sensitive Bayesian networks via direct and indirect methods. *Integrated Computer-Aided Engineering*, 24(1), pp.17–26.

Ngatchou, P., Zarei, A. & El-Sharkawi, A., 2005. Pareto Multi Objective Optimization. In *Proceedings of the 13th International Conference on, Intelligent Systems Application to Power Systems*. IEEE, pp. 84–91.

Norton, S.W., 1989. Generating better decision trees. *IJCAI'89 Proceedings of the 11th international joint conference on Artificial intelligence*, 1, pp.800–805.

Nuñez, M., 1991. The Use of Background Knowledge in Decision Tree Induction. *Machine Learning*, 6(3), pp.231–250.

Oellrich, A. et al., 2014. Using association rule mining to determine promising secondary phenotyping hypotheses. *Bioinformatics*, 30(12), pp.52–59.

Olivas, R., 2007. Decision trees, A Primer for Decision-making Professionals. In *SpringerReference*. Berlin/Heidelberg: Springer-Verlag.

Omielan, A. & Vadera, S., 2012. ECCO: A new evolutionary classifier with cost optimisation. *IFIP Advances in Information and Communication Technology*, 385 AICT, pp.97–105.

Osei-Bryson, K.-M., 2004. Evaluation of decision trees: a multi-criteria approach. *Computers & Operations Research*, 31(11), pp.1933–1945.

Ou, G. & Murphey, Y.L., 2007. Multi-class pattern classification using neural networks. *Pattern Recognition*, 40, pp.4–18.

Paquet, U. & Engelbrecht, A.P., 2003. A new particle swarm optimiser for linearly constrained optimisation. *2003 Congress on Evolutionary Computation, CEC 2003 - Proceedings*, 1, pp.227–233.

Passerini, A., 2013. Kernel Methods for Structured Data. In M. Bianchini, M. Maggini, & L. C. Jain, eds. *Radial Basis Function Networks: Design and Applications*. Intelligent Systems Reference Library. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 283–333.

Patel, N. & Upadhyay, S., 2012. Study of Various Decision Tree Pruning Methods with their Empirical Comparison in WEKA. *International Journal of Computer Applications*, 60(12), pp.20–25.

Peng, W., Chen, J. & Zhou, H., 2009. An Implementation of IDE3 Decision Tree Learning Algorithm. *Machine Learning*, 1, pp.1–20.

Platt, J., Cristianini, N. & Shawe-Taylor, J., 2000. Large Margin DAGs for Multiclass Classification. *International Conference on Neural Information Processing Systems*, pp.547–553.

Powers, D.M.W., 2011. Evaluation: From Precision, Recall and F-Measure To Roc, Informedness, Markedness & Correlation. *Journal of Machine Learning Technologies*, 2(1), pp.37–63.

Price, D. et al., 1994. Pairwise neural network classifiers with probabilistic outputs. *In Proceedings of the 7th International Conference on Neural Information Processing Systems*, 7, pp.1109–1116.

Qin, Z., Zhang, S. & Zhang, C., 2004. Cost-Sensitive Decision Trees with Multiple Cost Scales. In *In Proceedings of the 17th Australian joint conference on Advances in Artificial Intelligence*. pp. 380–390.

Quinlan, J.R., 1986. ID3:Induction of Decision Trees. *Machine Learning*, 1(1573–565), pp.81–106.

Rokach, L. & Maimon, O., 2005. Decision trees. In *The Data mining and knowledge discovery handbook*. New York: Springer, pp. 165–192.

Santra, a K. & Christy, C.J., 2012. Genetic algorithm and confusion matrix for document clustering. *International Journal of Computer Science*, 9(1), pp.322–328.

Sebban, M. et al., 2000. Impact of learning set quality and size on decision tree performances. *International Journal of Computers, Systems and Signals*, 1(1), pp.85–105.

Šeděnka, V. & Raida, Z., 2010. Critical Comparison of Multi-objective optimization methods: genetic algorithms versus swarm intelligence. *Radioengineering*, 19(3), pp.369–377.

Shalev-Shwartz, S. & Ben-David, S., 2014. *Understanding Machine Learning*, Cambridge: Cambridge University Press.

Shannon, C.E., 1948. A Mathematical Theory of Communication. *Mobile Computing and Communications Review (reprint)*, 5(I), pp.3–55.

Sheng, V. & Ling, C., 2006. Thresholding for making classifiers cost-sensitive. *Proceedings of the National Conference on Artificial Intelligence*, pp.476–481.

Shi, H., 2007. *Best-first Decision Tree Learning*. The University of Waikato, Hamilton, New Zealand.

Shilbayeh, S. & Vadera, S., 2014. Feature selection in meta learning framework. *Proceedings of 2014 Science and Information Conference, SAI 2014*, pp.269–275.

Sibanda, N.D., 2009. Quantitative Research. In *Encyclopedia of Research Design*. 2455 Teller Road, Thousand Oaks California 91320 United States: SAGE Publications, Inc., pp. 1–33.

Sierra, M.R. & Coello, C.A., 2005. Improving PSO-Based Multi-objective Optimization Using Crowding, Mutation and E-Dominance. *Lecture Notes in Computer Science*, 3410, pp.505–519.

Song, Y. & Lu, Y., 2015. Decision tree methods: applications for classification and prediction. *Biostatistics in psychiatry*, 27(April 2015), pp.130–135.

Sousa, T., Silva, A. & Neves, A., 2004. Particle Swarm based Data Mining Algorithms for classification tasks. *Parallel Computing*, 30, pp.767–783.

Stehman, S. V, 1997. Selecting and Interpreting Measures of Thematic Classification Accuracy. *Remote sensing of Environment*, 62, pp.77–89.

Steinberg, D., 2009. CART: Classification and Regression Trees. In *The Top Ten Algorithms in Data Mining*. pp. 179–201.

Steyerberg, E.W. et al., 2001. Internal validation of predictive models. *Journal of Clinical Epidemiology*, 54(8), pp.774–781.

Suthaharan, S., 2016. *Machine Learning Models and Algorithms for Big Data Classification*, Boston, MA: Springer US.

Suykens, J. a K., 2001. Nonlinear modelling and support vector machines. *IMTC 2001 Proceedings of the 18th IEEE Instrumentation and Measurement Technology Conference Rediscovering Measurement in the Age of Informatics Cat No01CH 37188*, 1, pp.287–294.

Tan, P.-N., Steinbach, M. & Kumar, V., 2006. Introduction to Data Mining. In *Cluster analysis: basic concepts and algorithms. Introduction to data mining*. Michigan State University and University of Minnesota, pp. 207–223.

Tang, J., Alelyani, S. & Liu, H., 2014. Feature Selection for Classification: A Review. In *Data Classification: Algorithms and Applications*. CRC Press-Taylor & Francis Group, LLC, pp. 37–64.

Ting, K.M., 2008. Cost-Sensitive Classification Using Decision Trees, Boosting and MetaCost. In *Heuristic and Optimization for Knowledge Discovery*. IGI Global, pp. 27–53.

Ting, K.M., 1998. Inducing cost-sensitive trees via instance weighting. In *In European*

Symposium on Principles of Data Mining and Knowledge Discovery. Springer Berlin Heidelberg, pp. 139–147.

Toledo-Pereyra, L.H., 2012. Research design. *Journal of Investigative Surgery*, 25(5), pp.279–280.

Turney, P., 1995. Cost-Sensitive Classification: Empirical Evaluation of a Hybrid Genetic Decision Tree Induction Algorithm. *Journal of Artificial Intelligence Research*, 2, pp.369–409.

Turney, P.D., 2002. Types of Cost in Inductive Concept Learning. In *17th International Conference on Machine Learning*. p. 7.

Vadera, 2010. CSNL: A cost-sensitive non-linear decision tree algorithm. *ACM Transactions on Knowledge Discovery from Data*, 4(2), pp.1–25.

Vapnik, V.N., 2000. *The Nature of Statistical Learning Theory* Second Edi., New York, NY: Springer New York.

Vens, C. et al., 2008. Decision trees for hierarchical multi-label classification. *Machine Learning*, 73(2), pp.185–214.

Vural, V. & Dy, J.G., 2004. A Hierarchical Method for Multi-class Support Vector Machines. *Proceedings of the Twenty-first International Conference on Machine Learning*, (2), p.105--.

Weiss, G., McCarthy, K. & Zabar, B., 2007. Cost-sensitive learning vs. sampling: Which is best for handling unbalanced classes with unequal error costs? *Dmin*, pp.35–41.

Witten, I.H., Frank, E. & Hall, M. a, 2016. *Data Mining: Practical Machine Learning Tools and Techniques* Fourth Edi., Morgan Kaufmann.

Xiaoyong Chai et al., 2004. Test-Cost Sensitive Naive Bayes Classification. *Fourth IEEE International Conference on Data Mining (ICDM'04)*, pp.51–58.

Yoav Freund & Schapire, R.E., 1999. A Short Introduction to Boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(5), pp.771–780.

Zadrozny, B., Langford, J. & Abe, N., 2003. Cost-sensitive learning by cost-proportionate example weighting. In *In Data Mining, 2003. ICDM 2003. Third IEEE International Conference*. IEEE Comput. Soc, pp. 435–442.

Zhang, S., Zhang, C. & Yang, Q., 2003. Data preparation for data mining. *Applied Artificial Intelligence*, 17(5–6), pp.375–381.

Zhao, H., 2007. A multi-objective genetic programming approach to developing Pareto optimal decision trees. *Decision Support Systems*, 43(3), pp.809–826.

Zitzler, E. & Thiele, L., 1999. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE T Evolut Comput*, 3(4), pp.257–271.

APPENDIX – A: Details of the Datasets Used in the Research

This appendix gives some details about the datasets used in this PhD search.

Bupa-Liver Disorders:

The Liver Disorders (BUPA) dataset is a medical dataset that has been created by BUPA Medical Research Ltd (McDermott & Forsyth 2016) and it is available in the UCI machine learning repository (Lichmakn 2013a). This dataset is a binary classification problem which contains 345 instances with 7 features (attributes) and a class label for each instance. The Bupa dataset analyses some liver disorders that might emerge from extreme alcohol consuming. The classification task of using this dataset is to predict if a particular person is suffering from alcoholism or not. The 345 instances are divided into 2 classes; class1=145 instances and class2= 200 instances.

The data set is available for download from UCI machine learning repository using the following link.

https://archive.ics.uci.edu/ml/datasets/liver+disorders

Hepatitis Dataset

The Hepatitis dataset is a medical dataset available in the UCI machine learning repository (Lichmakn 2013a). This dataset is a binary classification problem which contains 155 instances with 20 features (attributes) and a class label for each instance. The Hepatitis dataset analyses some information about a particular patient and the symptoms that are related to the Hepatitis disease that can appear on the patient. The classification task is to make a prediction for the patient if his illness may lead him to death or there is a big hope in life based on the symptoms that the particular has (Diaconis & Efron 1983).The data set is available for download from UCI machine learning repository using the following link.

https://archive.ics.uci.edu/ml/datasets/hepatitis

Statlog Heart Disease

The Statlog Heart Disease dataset is a medical dataset available in the UCI machine learning repository (Lichmakn 2013a). This dataset is a binary classification problem which contains 270 instances with 14 features (attributes) and a class label for each instance. The Statlog Heart Disease dataset analyses some personal and life style information about a particular patient and some results of medical tests that are related to the Heart Disease. The classification task is to diagnosis and makes a prediction for the patient if he or she suffers from the disease or not.

The data set is available for download from UCI machine learning repository using the following link.

http://archive.ics.uci.edu/ml/datasets/statlog+(heart)

Haberman's Survival Dataset

This dataset is a medical binary classification dataset available in the UCI machine learning repository (Lichmakn 2013a) which contains some information about the survival of patients who had undergone surgery for breast cancer. This information obtained from a study that was carried out at the Billings Hospital-University of Chicago's. This dataset contains 306 instances with 4 features (attributes) and a class label for each instance. The classification task with this dataset is to make a prediction for the survival of patients who had undergone surgery for breast cancer.

The data set is available for download from UCI machine learning repository using the following link.

https://archive.ics.uci.edu/ml/datasets/Haberman's+Survival

Diabetes Dataset

This dataset is also a medical binary classification dataset available in the UCI machine learning repository (Lichmakn 2013a) which contains some information about the daily lifestyle for a person such as the times and amounts of meals a person takes a day as well as the number of times and the type of the exercises and activities performed by the person on the day. This dataset also contains some information on the types and doses of insulin taken in the day as well as many readings and measurements of different blood tests performed on patients. This dataset contains 768 instances with 20 features (attributes) and a class label for each instance. The classification task with this dataset is to make a prediction for a particular person whether his information show signs of having diabetes or not.

The data set is available for download from UCI machine learning repository using the following link.

https://archive.ics.uci.edu/ml/datasets/diabetes

Breast Cancer Wisconsin (Diagnostic) – WDBC Dataset

The Breast Cancer Wisconsin (Diagnostic) dataset is a medical dataset available in the UCI machine learning repository (Lichmakn 2013a). This dataset is a binary classification problem which contains 569 instances with 32 features (attributes) and a class label for each instance. The WDBC dataset analyses some measurements that are computed for the nuclei of some cells on the tumor. The classification task is to makes a prediction for the type of the tumor as either malignant or benign.

The data set is available for download from UCI machine learning repository using the following link.

https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)

Breast Cancer Wisconsin (Prognostic) – WPBC Dataset

The Breast Cancer Wisconsin (Prognostic) dataset is similar to WDBC mentioned before. It contains 198 instances with 34 features (attributes). The classification task is also similar to WDBC where it makes a prediction for the type of the tumor as either malignant or benign. The data set is available for download from UCI machine learning repository using the following link.

https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Prognostic)

IRIS Datase:

The IRIS dataset was originally introduced by (Fisher 1936). It is one of the most well-known datasets that is widely used in the fields of data mining and pattern recognition. The data set comprises of 3 classes that represent 3 different types of iris plants, each class consists of 50 instances. One class can be separated linearly from the other two, but the other two cannot be separated linearly from each other. The dataset contains 4 Real-valued attributes in addition to the class label as follows:

1. Sepal length in cm
2. Sepal width in cm
3. Petal length in cm
4. Petal width in cm
5. The class label that holds one of 3 types of iris plants:
    - Iris Setosa - 1
    - Iris Versicolour - 2
    - Iris Virginica – 3

The classification task is to predict to which one of the three types of IRIS flowers a new unseen IRIS flowre will be classified.

The data set is available for download from UCI machine learning repository using the following link.

https://archive.ics.uci.edu/ml/datasets/iris

Hayes-Roth Dataset

This database is a social artificial dataset created by Barbara and Frederick Hayes-Roth (1977) to examine the behaviour of classifiers. It is a 3-class classification problem that contains 160 examples and 5 numerical attributes which are: name, hobby, age, educational level, marital status and the class label.

The data set is available for download from UCI machine learning repository using the following link.

https://archive.ics.uci.edu/ml/datasets/Hayes-Roth

Seeds Dataset:

Seeds dataset comprised of 7 geometric parameters related to the kernels of three different types of wheat named: the Kama, Rosa and Canadian, each variety represented by 70 instances (Charytanowicz et al. 2010). The dataset contains 7 Real-valued attributes in addition to the class label as follows:

1. Area
2. Perimeter
3. Compactness
4. Length of kernel
5. Width of kernel
6. Asymmetry coefficient
7. Length of kernel groove
8. The class label that holds one of 3 types of wheat:
    - Kama - 1
    - Rosa - 2
    - Canadian – 3

The classification task is to predict to which one of the three types of wheat a new unseen wheat seed will be classified.

The data set is available for download from UCI machine learning repository using the following link.

https://archive.ics.uci.edu/ml/datasets/seeds

Teaching Assistant Evaluation -Tae

The dataset is a 3-class problem that presents an estimation of teaching performance for 3 winter semesters and 2 summer semesters of 151 teaching assistant (TA) assignments. The classification task is to predict to which one of the three score categories ("low", "medium", and "high") a teaching assistant (TA) assignments will be classified.

The data set is available for download from UCI machine learning repository using the following link.

http://archive.ics.uci.edu/ml/datasets/Teaching%2BAssistant%2BEvaluation

Thyroid Disease Dataset

The Thyroid Disease dataset is a medical dataset available in the UCI machine learning repository (Lichmakn 2013a). This dataset is a 3-class problem which contains 216 instances with 21 features (attributes). The Thyroid Disease dataset contains some information about the medical history of the patient and some of the required test results. The classification task is to makes a prediction for the patient if he is normal or suffers from hyperthyroidism or hypothyroidism.

The data set is available for download from UCI machine learning repository using the following link.

http://archive.ics.uci.edu/ml/datasets/thyroid+disease

Glass Identification Dataset

The Glass identification dataset is a 7-class dataset that presents 7 different types of glass that are used to assists investigators in forensic science to find forensic evidences at crime scenes. This dataset contains 214 instances with 9 features (attributes) that give the characteristics of the different 7 types of glass. The data set is available for download from UCI machine learning repository using the following link.

https://archive.ics.uci.edu/ml/datasets/glass+identification

Ecoli Dataset

The Ecoli dataset is a medical 8-class dataset that presents 8 different localization sites of proteins that are presented as measures about the cell. This dataset contains 336 instances with 9 features (attributes). The main goal of using this dataset is to make a prediction for the localization site of proteins by using the provided measures of the cell.

The data set is available for download from UCI machine learning repository using the following link.

https://archive.ics.uci.edu/ml/datasets/ecoli

APPENDIX – B: Results of the Empirical Comparison Based on Accuracy

This appendix presents the results obtained by the ECSDT when adopting the first aspect which considers accuracy only as an objective function for the evaluation.

Bupa Accuracy-based Results				
Algorithm		**DT Size**	**Accuracy**	**SE**
J48		51	68.70	0.50
NBTree		11	66.38	0.48
BFTree		19	64.92	0.50
ADTree		31	59.71	0.47
LADTree		31	65.50	0.46
REPTree		23	64.05	0.49
ECSDT + OMOPSO	**2-ell**	5	49.41	0.51
	4-ell	9	59.71	0.87
	6-ell	13	58.82	0.51
	8-ell	17	65.85	0.43
	10-ell	21	68.52	0.77
ECSDT + NSGA-II	**2-ell**	5	53.53	0.43
	4-ell	9	59.41	0.76
	6-ell	13	60.59	0.62
	8-ell	17	66.47	0.39
	10-ell	21	72.35	0.46

Table Apx-B-01: Bupa Accuracy-based results

Hepatitis Accuracy-based Results				
Algorithm		**DT Size**	**Accuracy**	**SE**
J48		11	85.89	0.17
NBTree		5	84.61	0.35
BFTree		5	83.33	0.38
ADTree		31	87.17	0.31
LADTree		31	80.76	0.40
REPTree		11	87.17	0.31
ECSDT + OMOPSO	**2-ell**	5	81.43	0.46
	4-ell	9	82.86	0.62
	6-ell	13	85.71	0.49
	8-ell	17	84.29	0.55
	10-ell	21	84.29	0.41
ECSDT + NSGA-II	**2-ell**	5	80.00	0.44
	4-ell	9	82.86	0.57
	6-ell	13	85.71	0.44
	8-ell	17	88.57	0.49
	10-ell	21	87.14	0.54

Table Apx-B-02: Hepatitis Accuracy-based results

Heart Accuracy-based Results				
Algorithm		**DT Size**	**Accuracy**	**SE**
J48		39	77.40	0.44
NBTree		17	80.74	0.38
BFTree		37	77.40	0.44
ADTree		31	79.25	0.38
LADTree		31	80.00	0.36
REPTree		7	76.66	0.42
ECSDT + OMOPSO	**2-ell**	5	68.15	0.53
	4-ell	9	73.70	0.57
	6-ell	13	77.40	0.44
	8-ell	17	73.70	0.46
	10-ell	21	84.81	0.48
ECSDT + NSGA-II	**2-ell**	5	67.40	0.48
	4-ell	9	71.48	0.46
	6-ell	13	80.37	0.51
	8-ell	17	77.03	0.54
	10-ell	21	83.33	0.52

Table Apx-B-03: Heart Accuracy-based results

Haberman Accuracy-based Results				
Algorithm		**DT Size**	**Accuracy**	**SE**
J48		5	72.00	0.46
NBTree		3	73.00	0.43
BFTree		5	74.00	0.40
ADTree		31	72.00	0.48
LADTree		31	74.00	0.44
REPTree		5	71.00	0.53
ECSDT + OMOPSO	**2-ell**	5	73.00	0.53
	4-ell	9	76.00	0.49
	6-ell	13	77.00	0.47
	8-ell	17	78.00	0.56
	10-ell	21	79.00	0.55
ECSDT + NSGA-II	**2-ell**	5	75.00	0.54
	4-ell	9	75.00	0.57
	6-ell	13	78.00	0.47
	8-ell	17	80.00	0.44
	10-ell	21	80.00	0.53

Table Apx-B-04: Haberman Accuracy-based results

Diabetes Accuracy-based Results				
Algorithm		**DT Size**	**Accuracy**	**SE**
J48		39	73.80	0.44
NBTree		1	73.56	0.42
BFTree		5	73.56	0.44
ADTree		31	72.91	0.41
LADTree		31	74.08	0.42
REPTree		49	75.26	0.42
ECSDT + OMOPSO	**2-ell**	5	74.73	0.52
	4-ell	9	75.52	0.53
	6-ell	13	76.97	0.49
	8-ell	17	78.81	0.44
	10-ell	21	75.13	0.45
ECSDT + NSGA-II	**2-ell**	5	73.42	0.46
	4-ell	9	76.44	0.51
	6-ell	13	78.28	0.53
	8-ell	17	77.89	0.46
	10-ell	21	76.84	0.44

Table Apx-B-05: Diabetes Accuracy-based results

WDBC Accuracy-based Results				
Algorithm		**DT Size**	**Accuracy**	**SE**
J48		25	92.97	0.26
NBTree		23	92.79	0.25
BFTree		17	92.97	0.25
ADTree		31	94.72	0.18
LADTree		31	95.60	0.18
REPTree		9	92.44	0.25
ECSDT + OMOPSO	**2-ell**	5	85.88	0.34
	4-ell	9	92.14	0.36
	6-ell	13	85.35	0.28
	8-ell	17	93.20	0.25
	10-ell	21	87.32	0.26
ECSDT + NSGA-II	**2-ell**	5	85.71	0.28
	4-ell	9	88.92	0.31
	6-ell	13	90.88	0.29
	8-ell	17	93.34	0.28
	10-ell	21	89.63	0.25

Table Apx-B-6: WDBC Accuracy-based results

WPBC Accuracy-based Results				
Algorithm		**DT Size**	**Accuracy**	**SE**
J48		21	73.73	0.49
NBTree		11	71.21	0.47
BFTree		1	75.75	0.43
ADTree		31	73.23	0.43
LADTree		31	75.25	0.44
REPTree		7	72.22	0.45
ECSDT + PSO	**2-ell**	5	76.84	0.57
	4-ell	9	77.35	0.54
	6-ell	13	78.42	0.49
	8-ell	17	76.84	0.47
	10-ell	21	77.90	0.53
ECSDT + GA	**2-ell**	5	76.84	0.56
	4-ell	9	78.42	0.51
	6-ell	13	75.26	0.55
	8-ell	17	76.31	0.48
	10-ell	21	73.16	0.48

Table Apx-B-07: WPBC Accuracy-based results

IRIS Accuracy-based Results				
Algorithm		**DT Size**	**Accuracy**	**SE**
J48		9	96.00	0.158
NBTree		9	94.66	0.170
BFTree		11	94.66	0.175
ADTree		N/A	N/A	N/A
LADTree		31	94.00	0.193
REPTree		5	94.00	0.193
ECSDT + PSO	**3ell**	7	95.33	0.30
	4ell	9	98.66	0.28
	5ell	11	97.33	0.34
	6ell	13	96.66	0.35
	7ell	15	95.33	0.32
ECSDT + GA	**3ell**	7	94.66	0.52
	4ell	9	98.66	0.28
	5ell	11	96.66	0.46
	6ell	13	96.00	0.34
	7ell	15	95.33	0.44

Table Apx-B-08: IRIS Accuracy-based results

Hayes Accuracy-based Results				
Algorithm		**DT Size**	**Accuracy**	**SE**
J48		23	83.12	0.27
NBTree		13	64.37	0.38
BFTree		29	81.25	0.27
ADTree		N/A	N/A	N/A
LADTree		31	82.50	0.26
REPTree		25	83.75	0.27
ECSDT + PSO	**3ell**	7	57.50	0.36
	6ell	13	61.87	0.33
	9ell	19	65.62	0.41
	12ell	25	68.12	0.29
	15ell	31	62.50	0.31
ECSDT + GA	**3ell**	7	60.62	0.33
	6ell	13	68.12	0.34
	9ell	19	68.12	0.29
	12ell	25	71.25	0.28
	15ell	31	63.12	0.35

Table Apx-B-09: Hayes Accuracy-based results

Seeds Accuracy-based Results				
Algorithm		**DT Size**	**Accuracy**	**SE**
J48		15	91.90	0.23
NBTree		7	90.95	0.22
BFTree		19	93.33	0.21
ADTree		N/A	N/A	N/A
LADTree		31	91.90	0.23
REPTree		5	90.00	0.24
ECSDT + PSO	**3ell**	7	83.32	0.33
	6ell	13	87.62	0.35
	9ell	19	87.14	0.29
	12ell	25	93.80	0.29
	15ell	31	86.66	0.33
ECSDT + GA	**3ell**	7	85.71	0.34
	6ell	13	89.52	0.36
	9ell	19	87.14	0.28
	12ell	25	95.71	0.25
	15ell	31	89.04	0.27

Table Apx-B-10: Seeds Accuracy-based results

Tae Accuracy-based Results				
Algorithm		**DT Size**	**Accuracy**	**SE**
J48		67	59.60	0.46
NBTree		7	58.27	0.43
BFTree		28	57.61	0.45
ADTree		N/A	N/A	N/A
LADTree		31	59.60	0.43
REPTree		29	53.64	0.46
ECSDT + PSO	**3ell**	7	44.00	0.55
	6ell	13	51.33	0.57
	9ell	19	51.33	0.52
	12ell	25	54.00	0.48
	15ell	31	50.00	0.46
ECSDT + GA	**3ell**	7	43.33	0.49
	6ell	13	47.33	0.52
	9ell	19	49.32	0.52
	12ell	25	50.00	0.44
	15ell	31	55.32	0.47

Table Apx-B-011: Tae Accuracy-based results

Thyroid Accuracy-based Results				
Algorithm		**DT Size**	**Accuracy**	**SE**
J48		17	92.09	0.21
NBTree		7	93.02	0.20
BFTree		17	92.09	0.23
ADTree		N/A	N/A	N/A
LADTree		31	93.95	0.18
REPTree		7	92.09	0.22
ECSDT + PSO	**3ell**	7	85.70	0.29
	6ell	13	91.43	0.27
	9ell	19	87.14	0.32
	12ell	25	87.61	0.29
	15ell	31	89.04	0.26
ECSDT + GA	**3ell**	7	88.57	0.28
	6ell	13	94.28	0.24
	9ell	19	92.37	0.31
	12ell	25	89.52	0.29
	15ell	31	91.90	0.24

Table Apx-B-012: Thyroid Accuracy-based results

Glass Accuracy-based Results				
Algorithm		**DT Size**	**Accuracy**	**SE**
J48		11	96.72	0.22
NBTree		7	93.45	0.19
BFTree		11	98.13	0.21
ADTree		N/A	N/A	N/A
LADTree		28	98.13	0.21
REPTree		11	98.59	0.20
ECSDT + PSO	**6ell**	12	79.04	0.30
	8ell	17	89.04	0.28
	10ell	21	90.47	0.31
	12ell	25	92.37	0.27
	14ell	29	93.32	0.29
ECSDT + GA	**6ell**	12	81.89	0.29
	8ell	17	90.47	0.27
	10ell	21	93.32	0.33
	12ell	25	94.28	0.31
	14ell	29	95.23	0.26

Table Apx-B-13: Glass Accuracy-based results

Ecoli Accuracy-based Results				
Algorithm		**DT Size**	**Accuracy**	**SE**
J48		41	79.76	0.42
NBTree		13	80.06	0.39
BFTree		29	78.86	0.51
ADTree		N/A	N/A	N/A
LADTree		31	82.44	0.38
REPTree		25	76.79	0.55
ECSDT + PSO	**6ell**	12	72.72	0.37
	8ell	17	76.35	0.48
	10ell	21	76.35	0.44
	12ell	25	82.72	0.55
	14ell	29	74.50	0.52
ECSDT + GA	**6ell**	12	73.32	0.52
	8ell	17	77.26	0.48
	10ell	21	76.96	0.46
	12ell	25	83.33	0.54
	14ell	29	76.35	0.48

Table Apx-B-014: Ecoli Accuracy-based results

APPENDIX – C: Results of the Empirical Comparison Based on Cost

This appendix presents the results obtained by the ECSDT when adopting the second aspect which considers cost only as an objective function for the evaluation.

Algorithm	Bupa Cost-based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio - 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	46.67	1.08	51	43.77	20.81	51	43.77	20.81	51	43.77	20.81	51	43.77	41.10	51	66.96	0.72	51
MetaCost+J48	50.14	0.81	37	42.03	0.58	1	42.03	0.58	1	42.03	0.58	1	42.03	0.58	1	61.74	0.49	23
C.S.C+NBTree	42.32	0.66	11	42.03	0.58	11	42.03	0.58	11	42.03	0.58	11	42.03	0.58	11	57.68	0.48	11
MetaCost+NBTree	43.19	0.57	1	42.03	0.58	1	42.03	0.58	1	42.03	0.58	1	42.03	0.58	1	57.97	0.42	1
ECSDT-PSO-2ell	43.76	0.72	5	42.89	1.00	5	39.71	1.18	5	39.13	3.50	5	37.1	9.32	5	48.11	1.41	5
ECSDT-PSO-4ell	42.608	0.57	9	47.24	0.81	9	45.5	0.83	9	42.02	3.47	9	44.63	6.34	9	57.1	0.51	9
ECSDT-PSO-6ell	42.028	0.58	13	48.4	0.80	13	47.53	1.10	13	42.02	3.47	13	47.53	3.42	13	57.39	0.48	13
ECSDT-PSO-8ell	48.69	0.59	17	48.98	0.80	17	44.92	0.84	17	43.18	0.57	17	46.37	0.54	17	59.71	0.41	17
ECSDT-PSO-10ell	48.82	0.53	21	43.19	0.57	21	46.37	0.54	21	44.05	0.56	21	46.66	0.53	21	60.86	0.39	21
ECSDT-GA-2ell	42.05	0.70	5	42.6	1.00	5	38.55	1.19	5	37.97	3.51	5	36.81	12.21	5	50.72	1.14	5
ECSDT-GA-4ell	44.05	0.64	9	48.69	0.66	9	47.82	0.81	9	44.05	3.45	9	42.89	6.36	9	59.71	0.46	9
ECSDT-GA-6ell	44.41	0.57	13	46.67	0.68	13	47.82	0.81	13	48.69	1.96	13	45.5	3.44	13	57.39	0.48	13
ECSDT-GA-8ell	47.24	0.55	17	43.76	0.56	17	44.34	0.56	17	46.08	1.99	17	43.76	0.56	17	60.57	0.39	17
ECSDT-GA-10ell	50.72	0.52	21	45.5	0.54	21	46.66	0.52	21	45.5	0.54	21	46.08	0.54	21	61.73	0.38	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Ratio - 10			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	63.80	2.21	51	63.77	4.09	51	63.77	19.17	51	63.77	38.01	51	54.378	13.43	51
MetaCost+J48	58.00	0.56	7	57.97	0.42	1	57.97	0.42	1	57.97	0.42	1	51.275	0.56	7.4
C.S.C+NBTree	57.70	0.57	11	57.97	0.42	11	57.97	0.42	11	57.97	0.42	11	50.028	0.53	11
MetaCost+NBTree	58.00	0.42	1	57.97	0.42	1	57.97	0.42	1	57.97	0.42	1	50.115	0.50	1
ECSDT-PSO-2ell	48.98	4.91	5	53.33	5.06	5	54.2	19.26	5	55.07	32.30	5	46.228	7.86	5
ECSDT-PSO-4ell	60.00	0.68	9	58.84	0.99	9	57.10	4.77	9	57.97	9.11	9	51.301	2.81	9
ECSDT-PSO-6ell	60.57	0.54	13	63.76	0.65	13	57.68	1.87	13	59.42	3.30	13	52.633	1.62	13
ECSDT-PSO-8ell	64.63	0.50	17	62.31	0.66	17	60.86	0.39	17	61.15	0.39	17	54.08	0.57	17
ECSDT-PSO-10ell	63.76	0.50	21	65.21	0.63	21	62.6	0.37	21	59.71	0.40	21	54.123	0.50	21
ECSDT-GA-2ell	50.72	4.03	5	53.33	5.06	5	54.2	19.26	5	54.49	35.20	5	46.144	8.33	5
ECSDT-GA-4ell	62.31	0.66	9	64.34	0.64	9	57.39	3.32	9	57.97	6.21	9	52.922	2.32	9
ECSDT-GA-6ell	58.26	0.56	13	64.34	0.64	13	57.68	1.87	13	60.57	3.29	13	53.133	1.43	13
ECSDT-GA-8ell	58.26	0.56	17	64.63	0.93	17	61.73	0.38	17	62.02	0.38	17	53.239	0.69	17
ECSDT-GA-10ell	61.15	0.53	21	66.37	0.62	21	64.63	0.35	21	62.89	0.37	21	55.123	0.49	21

Table Apx-C-1: Bupa Cost-based results

Algorithm	Hepatitis Cost-based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	79.49	0.55	9	64.1	1.62	5	58.97	1.68	5	19.23	0.81	1	19.23	0.81	1	80.77	0.19	1
MetaCost+J48	70.51	0.64	13	60.26	0.40	9	32.05	1.95	1	19.23	0.81	1	19.23	0.81	1	82.05	0.18	1
C.S.C+NBTree	70.51	0.64	5	39.74	1.23	5	28.21	0.72	5	25.64	0.74	5	19.23	0.81	5	80.77	0.31	5
MetaCost+NBTree	73.08	0.73	5	50	0.50	5	26.92	0.73	5	19.23	0.81	1	19.23	0.81	3	80.77	0.31	1
ECSDT-PSO-2ell	55.12	0.56	5	61.53	1.64	5	67.94	2.86	5	43.58	6.96	5	43.58	13.37	5	78.2	0.45	5
ECSDT-PSO-4ell	60.25	0.51	9	71.79	0.91	9	82.05	1.45	9	51.28	6.88	9	48.71	13.32	9	80.76	0.31	9
ECSDT-PSO-6ell	58.97	0.41	13	55.12	0.45	13	41.02	0.59	13	52.56	0.47	13	50	0.50	13	80.76	0.19	13
ECSDT-PSO-8ell	58.97	0.41	17	64.1	0.36	17	48.71	0.51	17	60.25	0.40	17	56.41	0.44	17	82.05	0.18	17
ECSDT-PSO-10ell	55.12	0.45	21	57.69	0.42	21	43.58	0.56	21	47.43	0.53	21	50	0.50	21	84.61	0.27	21
ECSDT-GA-2ell	47.43	0.64	5	69.23	1.56	5	65.38	2.88	5	50.5	6.90	5	47.43	13.33	5	78.2	0.45	5
ECSDT-GA-4ell	51.28	0.60	9	43.58	1.19	9	83.33	1.44	9	61.53	6.78	9	58.97	13.22	9	80.76	0.31	9
ECSDT-GA-6ell	60.25	0.40	13	57.69	0.42	13	43.58	0.56	13	46.15	0.54	13	53.84	0.46	13	84.61	0.27	13
ECSDT-GA-8ell	58.97	0.41	17	61.25	0.38	17	48.71	0.51	17	57.69	0.42	17	53.84	0.46	17	80.76	0.19	17
ECSDT-GA-10ell	55.12	0.45	21	55.12	0.45	21	37.17	0.63	21	57.69	0.42	21	57.69	0.42	21	85.89	0.26	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Ratio - 10			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	80.77	0.19	1	80.77	0.19	1	80.77	0.19	1	80.77	0.19	1	64.49	0.64	2.6
MetaCost+J48	80.77	0.19	1	80.77	0.19	1	80.77	0.19	1	80.77	0.19	1	60.64	0.56	3
C.S.C+NBTree	80.77	0.19	5	80.77	0.19	5	80.77	0.19	5	80.77	0.19	5	58.72	0.52	5
MetaCost+NBTree	80.77	0.19	1	80.77	0.19	1	80.77	0.19	1	80.77	0.19	1	59.23	0.47	2.4
ECSDT-PSO-2ell	80.76	2.08	5	78.2	2.76	5	79.49	6.60	5	79.49	13.01	5	66.79	5.03	5
ECSDT-PSO-4ell	78.2	1.47	9	80.76	2.73	9	83.33	6.56	9	84.62	12.96	9	72.18	4.71	9
ECSDT-PSO-6ell	82.05	0.81	13	82.05	1.45	13	82.05	6.58	13	84.62	12.96	13	66.92	2.44	13
ECSDT-PSO-8ell	85.89	0.77	17	82.05	1.45	17	84.62	6.55	17	85.9	12.95	17	70.90	2.40	17
ECSDT-PSO-10ell	82.05	0.81	21	80.76	1.46	21	82.05	6.58	21	84.62	12.96	21	66.79	2.45	21
ECSDT-GA-2ell	78.2	1.47	5	80.76	2.73	5	79.49	6.60	5	79.49	13.01	5	67.61	4.96	5
ECSDT-GA-4ell	80.76	0.82	9	80.76	1.46	9	84.62	6.55	9	85.9	12.95	9	71.15	4.53	9
ECSDT-GA-6ell	80.76	0.82	13	80.76	1.46	13	85.9	6.54	13	84.62	12.96	13	67.82	2.44	13
ECSDT-GA-8ell	80.76	0.82	17	84.61	1.42	17	87.17	6.53	17	85.9	12.95	17	69.97	2.41	17
ECSDT-GA-10ell	84.61	0.78	21	84.61	1.42	21	82.05	6.58	21	84.62	12.96	21	68.46	2.44	21

Table Apx-C-2: Hepatitis Cost-based results

Algorithm	Heart Cost-based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	68.51	1.11	39	60.00	2.40	39	60.00	4.43	39	60.00	20.73	39	60.00	41.10	39	67.40	0.96	39
MetaCost+J48	66.29	0.64	31	61.11	0.39	15	55.55	0.44	1	55.55	0.44	1	55.55	0.44	1	71.11	0.52	29
C.S.C+NBTree	74.44	0.39	17	67.03	0.51	17	60.37	0.40	17	58.88	0.41	17	55.55	0.44	17	66.29	0.60	17
MetaCost+NBTree	70.00	0.50	11	56.66	0.43	1	55.55	0.44	1	55.55	0.44	1	55.55	0.44	1	62.96	0.57	11
ECSDT-PSO-2ell	49.62	1.04	5	51.11	2.48	5	50.74	5.26	5	51.85	18.96	5	52.96	26.37	5	53.33	0.47	5
ECSDT-PSO-4ell	53.7	0.46	9	56.66	0.94	9	57.77	1.89	9	58.14	5.96	9	54.44	11.56	9	60.37	0.40	9
ECSDT-PSO-6ell	54.81	0.45	13	55.18	0.63	13	63.33	0.73	13	56.29	4.13	13	63.70	4.06	13	65.18	0.35	13
ECSDT-PSO-8ell	66.66	0.33	17	62.22	0.38	17	62.22	0.38	17	55.55	0.44	17	56.29	0.44	17	67.77	0.32	17
ECSDT-PSO-10ell	68.88	0.31	21	68.14	0.32	21	65.92	0.34	21	64.44	0.36	21	58.55	0.41	21	72.59	0.27	21
ECSDT-GA-2ell	47.77	1.22	5	51.85	2.30	5	50.74	4.89	5	52.22	17.11	5	53.70	26.36	5	54.07	0.46	5
ECSDT-GA-4ell	50.74	0.49	9	53.70	1.37	9	61.11	1.49	9	55.92	5.99	9	54.81	11.55	9	58.88	0.41	9
ECSDT-GA-6ell	55.55	0.44	13	57.03	0.61	13	67.40	0.69	13	58.51	2.26	13	62.56	4.07	13	65.18	0.35	13
ECSDT-GA-8ell	65.92	0.34	17	63.70	0.36	17	55.55	0.44	17	68.14	0.32	17	55.56	0.44	17	65.18	0.35	17
ECSDT-GA-10ell	70.37	0.30	21	66.29	0.34	21	67.03	0.33	21	65.55	0.34	21	60.74	0.39	21	73.33	0.27	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Ratio - 10			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	50.74	1.76	39	48.88	3.08	39	48.88	13.45	39	48.88	26.41	39	57.33	11.54	39
MetaCost+J48	52.22	0.66	1	44.44	0.56	1	44.44	0.56	1	44.44	0.56	1	55.07	0.52	8.2
C.S.C+NBTree	52.22	1.02	17	48.14	1.25	17	45.55	2.39	17	45.18	0.55	17	57.37	0.80	17
MetaCost+NBTree	44.44	0.56	1	44.44	0.56	1	44.44	0.56	1	44.44	0.56	1	53.41	0.51	3
ECSDT-PSO-2ell	46.66	1.62	5	43.33	2.03	5	43.33	11.66	5	42.96	22.77	5	48.59	9.27	5
ECSDT-PSO-4ell	50.74	1.04	9	43.33	1.30	9	45.92	6.09	9	43.70	7.96	9	52.48	3.76	9
ECSDT-PSO-6ell	54.44	0.82	13	46.66	0.90	13	51.11	2.34	13	56.66	4.13	13	56.74	1.85	13
ECSDT-PSO-8ell	58.88	0.41	17	52.59	0.47	17	54.44	2.30	17	50.37	4.20	17	58.70	0.97	17
ECSDT-PSO-10ell	56.66	0.43	21	50.74	0.49	21	50.37	0.50	21	46.66	0.53	21	60.30	0.40	21
ECSDT-GA-2ell	46.66	1.44	5	43.33	0.57	5	42.96	9.81	5	43.70	19.06	5	48.70	8.32	5
ECSDT-GA-4ell	48.88	1.06	9	42.59	0.57	9	45.18	4.24	9	46.29	7.94	9	51.81	3.51	9
ECSDT-GA-6ell	52.96	0.83	13	46.29	0.90	13	52.59	2.32	13	45.92	4.24	13	56.40	1.67	13
ECSDT-GA-8ell	55.55	0.44	17	53.70	0.46	17	53.70	2.31	17	51.48	4.19	17	58.85	0.97	17
ECSDT-GA-10ell	57.03	0.43	21	55.92	0.44	21	49.25	0.51	21	45.55	0.54	21	61.11	0.39	21

Table Apx-C-03: Heart Cost-based results

Algorithm	Haberman Cost-based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	73.529	0.26	5	73.529	0.26	5	73.529	0.26	5	73.529	0.26	5	73.529	0.26	5	27.777	0.75	5
MetaCost+J48	73.529	0.26	1	73.529	0.26	1	73.529	0.26	1	73.529	0.26	1	73.529	0.26	1	29.738	0.76	21
C.S.C+NBTree	73.529	0.26	3	73.529	0.26	3	73.529	0.26	3	73.529	0.26	3	73.529	0.26	3	26.47	0.74	3
MetaCost+NBTree	73.529	0.26	1	73.529	0.26	1	73.529	0.26	1	73.529	0.26	1	73.529	0.26	1	26.47	0.74	1
ECSDT-PSO-2ell	72.22	0.57	5	72.55	0.75	5	70.92	1.58	5	71.57	3.55	5	71.24	6.82	5	26.47	1.06	5
ECSDT-PSO-4ell	71.9	0.46	9	72.88	0.59	9	72.88	1.24	9	74.84	3.51	9	73.53	3.53	9	28.1	0.87	9
ECSDT-PSO-6ell	74.51	0.31	13	77.12	0.39	13	74.84	0.90	13	72.88	1.90	13	75.16	3.51	13	33.66	0.75	13
ECSDT-PSO-8ell	75.49	0.25	17	76.46	0.24	17	76.8	0.23	17	73.86	0.26	17	76.8	0.23	17	38.56	0.67	17
ECSDT-PSO-10ell	77.12	0.23	21	77.78	0.22	21	77.12	0.23	21	74.84	0.25	21	77.78	0.22	21	34.97	0.65	21
ECSDT-GA-2ell	72.55	0.51	5	70.92	0.93	5	72.88	1.57	5	70.92	3.55	5	73.53	6.79	5	26.8	1.08	5
ECSDT-GA-4ell	72.88	0.39	9	71.24	0.61	9	71.9	0.93	9	73.2	3.53	9	74.84	3.52	9	28.76	0.83	9
ECSDT-GA-6ell	75.16	0.31	13	72.88	0.43	13	74.51	0.25	13	74.18	1.89	13	76.47	3.50	13	34.97	0.74	13
ECSDT-GA-8ell	74.51	0.25	17	73.53	0.26	17	73.2	0.59	17	75.16	0.25	17	77.12	0.23	17	37.91	0.68	17
ECSDT-GA-10ell	77.12	0.23	21	76.14	0.26	21	76.47	0.24	21	75.82	0.24	21	77.12	0.23	21	36.6	0.63	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Ratio - 10			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	26.47	0.90	5	26.47	1.06	5	26.47	2.37	5	26.47	4.00	5	50.13	1.04	5
MetaCost+J48	26.47	0.74	1	26.47	0.74	1	26.47	0.74	1	26.47	0.74	1	50.33	0.50	3
C.S.C+NBTree	26.47	0.74	3	26.47	0.74	3	26.47	0.74	3	26.47	0.74	3	50.00	0.50	3
MetaCost+NBTree	26.47	0.74	1	26.47	0.74	1	26.47	0.74	1	26.47	0.74	1	50.00	0.50	1
ECSDT-PSO-2ell	26.14	1.70	5	25.82	2.03	5	26.8	7.25	5	26.8	7.26	5	49.05	3.26	5
ECSDT-PSO-4ell	26.8	1.37	9	27.12	1.70	9	27.45	5.62	9	28.76	3.98	9	50.43	2.29	9
ECSDT-PSO-6ell	33.99	0.98	13	33.99	1.31	13	31.37	3.95	13	32.68	3.94	13	54.02	1.79	13
ECSDT-PSO-8ell	34.97	0.81	17	36.6	0.96	17	34.97	1.95	17	32.03	0.68	17	55.65	0.63	17
ECSDT-PSO-10ell	35.62	0.64	21	35.95	0.64	21	35.29	0.65	21	33.33	0.67	21	55.98	0.44	21
ECSDT-GA-2ell	25.82	1.54	5	26.47	1.71	5	25.82	5.63	5	26.8	7.26	5	49.25	3.06	5
ECSDT-GA-4ell	26.8	1.21	9	26.8	1.70	9	27.45	3.99	9	28.1	3.98	9	50.20	2.07	9
ECSDT-GA-6ell	32.35	0.84	13	32.68	1.00	13	33.33	2.30	13	35.29	3.91	13	54.18	1.52	13
ECSDT-GA-8ell	34.64	0.81	17	35.62	0.97	17	36.93	2.26	17	32.68	0.67	17	55.13	0.70	17
ECSDT-GA-10ell	36.93	0.63	21	35.29	0.65	21	34.97	0.65	21	35.62	0.64	21	56.21	0.44	21

Table Apx-C-04: Haberman Cost-based results

Algorithm	Diabetes Cost-based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	65.10	0.60	39	64.71	1.31	39	64.71	2.29	39	64.71	10.10	39	64.71	19.86	39	57.16	0.83	39
MetaCost+J48	70.05	0.44	19	65.10	0.35	1	65.10	0.35	1	65.10	0.35	1	65.10	0.35	1	62.11	0.67	57
C.S.C+NBTree	68.62	0.44	1	65.63	0.41	1	65.10	0.35	1	65.10	0.35	1	65.10	0.35	1	59.90	0.62	1
MetaCost+NBTree	65.63	0.34	7	65.10	0.35	1	65.10	0.35	1	65.10	0.35	1	65.10	0.35	1	55.86	0.58	3
ECSDT-PSO-2ell	61.72	0.69	5	62.89	1.46	5	62.90	2.56	5	62.89	12.72	5	62.50	28.99	5	45.31	0.66	5
ECSDT-PSO-4ell	60.94	0.43	9	62.50	0.63	9	66.41	1.37	9	64.84	2.95	9	65.63	6.85	9	52.34	0.55	9
ECSDT-PSO-6ell	66.80	0.36	13	68.75	0.38	13	67.96	0.58	13	70.31	1.60	13	65.23	2.95	13	57.81	0.45	13
ECSDT-PSO-8ell	70.31	0.30	17	68.75	0.31	17	66.41	0.34	17	69.14	0.31	17	66.80	0.33	17	63.28	0.37	17
ECSDT-PSO-10ell	68.36	0.32	21	67.58	0.32	21	66.02	0.34	21	67.58	0.32	21	66.80	0.33	21	59.77	0.40	21
ECSDT-GA-2ell	60.16	0.84	5	62.89	1.46	5	62.89	2.56	5	62.89	12.72	5	62.50	28.99	5	46.09	0.64	5
ECSDT-GA-4ell	63.67	0.49	9	63.28	0.62	9	64.45	1.13	9	64.97	2.95	9	64.45	6.86	9	51.95	0.54	9
ECSDT-GA-6ell	65.23	0.37	13	66.41	0.40	13	66.41	0.46	13	64.06	1.66	13	67.19	1.63	13	59.77	0.43	13
ECSDT-GA-8ell	69.53	0.30	17	68.36	0.32	17	66.80	0.33	17	68.48	0.32	17	64.84	0.35	17	62.89	0.37	17
ECSDT-GA-10ell	67.58	0.32	21	69.14	0.31	21	67.58	0.32	21	67.97	0.32	21	67.97	0.32	21	66.80	0.33	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Ratio - 10			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	41.02	1.55	39	39.45	2.54	39	39.45	10.35	39	39.45	20.12	39	54.05	6.95	39
MetaCost+J48	44.40	0.62	15	36.20	0.64	1	34.90	0.65	1	34.90	0.65	1	54.30	0.51	9.8
C.S.C+NBTree	44.14	0.75	1	40.76	0.72	1	35.81	0.64	1	34.90	0.65	1	54.50	0.53	1
MetaCost+NBTree	38.15	0.62	3	34.90	0.65	1	34.90	0.65	1	34.90	0.65	1	52.47	0.49	2
ECSDT-PSO-2ell	33.33	1.62	5	34.11	1.95	5	34.11	7.16	5	33.33	18.88	5	49.31	7.67	5
ECSDT-PSO-4ell	38.80	1.25	9	36.85	1.28	9	35.68	3.89	9	36.46	4.54	9	52.05	2.37	9
ECSDT-PSO-6ell	40.23	0.79	13	39.19	0.87	13	38.80	1.91	13	40.63	1.89	13	55.57	1.18	13
ECSDT-PSO-8ell	41.80	0.71	17	44.53	0.68	17	38.02	0.62	17	42.19	0.58	17	57.12	0.45	17
ECSDT-PSO-10ell	46.88	0.53	21	45.70	0.54	21	42.19	0.58	21	37.63	0.62	21	56.85	0.43	21
ECSDT-GA-2ell	32.55	1.50	5	34.51	1.82	5	33.72	6.51	5	34.11	12.37	5	49.23	6.94	5
ECSDT-GA-4ell	39.19	1.05	9	37.24	1.27	9	35.29	3.25	9	37.63	4.53	9	52.21	2.27	9
ECSDT-GA-6ell	42.58	0.77	13	40.23	0.86	13	37.24	1.28	13	38.80	1.91	13	54.79	0.98	13
ECSDT-GA-8ell	44.14	0.62	17	45.31	0.68	17	42.58	0.57	17	43.62	0.56	17	57.66	0.44	17
ECSDT-GA-10ell	47.66	0.52	21	43.75	0.56	21	42.97	0.57	21	38.41	0.62	21	57.98	0.42	21

Table Apx-C-05: Diabetes Cost-based results

Algorithm	WDBC Cost-based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	92.97	0.31	25	92.26	1.28	25	66.43	2.42	25	45.69	8.44	25	45.69	16.34	25	92.97	0.40	25
MetaCost+J48	90.68	0.19	23	88.57	0.63	23	86.64	1.00	27	37.60	0.62	1	37.26	0.63	1	92.97	0.12	17
C.S.C+NBTree	92.26	0.35	23	87.87	1.24	23	78.20	1.78	23	71.88	6.42	23	72.06	10.81	23	93.14	0.31	23
MetaCost+NBTree	89.45	0.20	23	89.10	0.71	27	88.40	0.81	7	44.81	3.18	19	39.37	0.61	5	91.73	0.10	19
ECSDT-PSO-2ell	73.64	0.50	5	58.88	1.70	5	53.60	2.90	5	73.64	3.74	5	58.88	1.70	5	65.20	0.66	5
ECSDT-PSO-4ell	76.80	0.39	9	70.47	1.33	9	60.98	2.13	9	77.33	2.49	9	70.47	1.33	9	74.17	0.51	9
ECSDT-PSO-6ell	83.13	0.29	13	79.44	1.07	13	68.89	1.70	13	83.66	1.38	13	79.96	0.98	13	83.13	0.26	13
ECSDT-PSO-8ell	86.82	0.20	17	86.29	0.65	17	75.22	1.29	17	90.51	0.96	17	88.4	0.72	17	88.93	0.16	17
ECSDT-PSO-10ell	88.93	0.17	21	88.40	0.46	21	83.66	0.69	21	92.09	0.60	21	90.51	0.61	21	89.46	0.11	21
ECSDT-GA-2ell	73.64	0.50	5	58.88	1.70	5	55.18	2.71	5	73.64	3.74	5	53.60	2.90	5	65.20	0.66	5
ECSDT-GA-4ell	76.80	0.39	9	70.47	1.33	9	60.89	2.13	9	78.91	2.65	9	60.98	2.13	9	75.75	0.37	9
ECSDT-GA-6ell	85.76	0.27	13	79.96	0.98	13	69.42	1.52	13	84.71	1.54	13	68.89	1.70	13	82.60	0.25	13
ECSDT-GA-8ell	84.18	0.19	17	88.40	0.72	17	75.75	1.11	17	89.46	0.80	17	75.22	1.29	17	88.40	0.15	17
ECSDT-GA-10ell	88.93	0.17	21	90.51	0.61	21	83.66	0.69	21	92.09	0.60	21	83.66	0.69	21	88.93	0.11	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Ratio - 10			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	92.61	1.80	25	92.61	3.55	25	92.61	17.61	25	92.61	35.19	25	88.31	1.63	25
MetaCost+J48	92.97	0.33	21	92.09	0.60	23	73.28	3.43	19	62.74	2.13	15	90.65	0.48	22.3
C.S.C+NBTree	88.92	1.14	23	79.78	1.59	23	72.40	4.66	23	72.58	7.30	23	86.7	1.07	23
MetaCost+NBTree	89.98	0.53	17	85.76	0.84	5	66.60	0.33	3	63.09	0.37	15	89.07	0.53	16.3
ECSDT-PSO-2ell	70.47	2.02	5	73.64	3.74	5	73.64	0.50	5	73.64	3.74	5	65.91	1.92	5
ECSDT-PSO-4ell	76.80	1.70	9	78.91	2.65	9	76.80	0.39	9	77.33	2.49	9	73.02	1.45	9
ECSDT-PSO-6ell	83.13	1.03	13	84.71	1.54	13	83.13	0.29	13	83.66	1.38	13	80.41	0.98	13
ECSDT-PSO-8ell	87.87	0.64	17	89.46	0.80	17	86.82	0.20	17	90.51	0.96	17	85.77	0.62	17
ECSDT-PSO-10ell	91.04	0.35	21	92.09	0.60	21	88.93	0.17	21	92.09	0.60	21	88.93	0.40	21
ECSDT-GA-2ell	70.47	2.02	5	73.64	3.74	5	73.64	0.50	5	73.64	3.74	5	66.17	1.89	5
ECSDT-GA-4ell	76.80	1.70	9	77.33	2.49	9	76.80	0.39	9	78.91	2.65	9	73.01	1.40	9
ECSDT-GA-6ell	82.07	0.95	13	83.66	1.38	13	85.76	0.27	13	84.71	1.54	13	80.58	0.89	13
ECSDT-GA-8ell	89.98	0.53	17	90.51	0.96	17	84.18	0.19	17	89.46	0.80	17	86.2	0.61	17
ECSDT-GA-10ell	91.04	0.35	21	92.09	0.60	21	88.93	0.17	21	92.09	0.60	21	89.19	0.42	21

Table Apx-C-6: WDBC Cost-based results

Algorithm	WPBC Cost-based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	72.72	1.41	21	29.79	3.67	21	29.79	6.70	21	29.79	3.67	21	29.79	6.70	21	73.23	1.18	21
MetaCost+J48	44.95	1.14	25	37.87	2.85	31	29.80	3.70	9	37.87	2.85	1	29.80	3.70	1	76.76	0.28	1
C.S.C+NBTree	47.98	1.25	11	23.23	1.01	11	23.23	1.27	11	23.23	1.01	11	23.23	1.27	11	75.76	0.33	11
MetaCost+NBTree	39.39	1.29	11	24.75	0.75	1	23.73	0.76	1	24.75	0.75	1	23.73	0.76	1	75.76	0.38	1
ECSDT-PSO-2ell	28.28	2.08	5	24.75	4.46	5	22.22	6.78	5	24.75	4.46	5	22.22	6.78	5	51.52	2.71	5
ECSDT-PSO-4ell	30.81	1.46	9	27.78	2.95	9	26.77	5.73	9	27.78	2.95	9	26.77	5.73	9	58.59	2.19	9
ECSDT-PSO-6ell	34.85	1.11	13	30.81	1.93	13	31.80	5.68	13	30.81	1.93	13	31.80	5.68	13	62.63	1.60	13
ECSDT-PSO-8ell	40.40	1.01	17	32.32	1.67	17	31.31	2.69	17	32.32	1.67	17	31.31	2.69	17	71.72	0.69	17
ECSDT-PSO-10ell	43.43	0.93	21	34.85	1.15	21	33.33	1.17	21	34.85	1.15	21	33.33	1.17	21	76.77	0.28	21
ECSDT-GA-2ell	28.28	2.08	5	24.75	4.46	5	22.22	6.78	5	24.75	4.46	5	22.22	6.78	5	51.52	2.71	5
ECSDT-GA-4ell	31.31	1.46	9	27.78	2.95	9	26.80	5.73	9	27.78	2.95	9	26.80	5.73	9	57.07	2.16	9
ECSDT-GA-6ell	35.86	1.05	13	29.80	2.19	13	29.29	4.21	13	29.80	2.19	13	29.29	4.21	13	63.13	1.55	13
ECSDT-GA-8ell	39.90	0.96	17	32.83	1.41	17	31.31	2.69	17	32.83	1.41	17	31.31	2.69	17	71.72	0.69	17
ECSDT-GA-10ell	44.44	0.96	21	35.35	0.89	21	33.84	1.16	21	35.35	0.89	21	33.84	1.16	21	76.76	0.28	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Ratio - 10			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	73.23	4.47	21	73.23	8.77	21	73.23	1.18	21	73.23	4.47	21	58.67	4.37	11.5
MetaCost+J48	75.76	0.49	1	76.26	0.24	1	76.76	0.28	1	75.76	0.49	1	56.9	1.45	12.8
C.S.C+NBTree	75.76	0.49	11	75.76	0.74	11	75.76	0.33	11	75.76	0.49	11	53.62	0.85	8.5
MetaCost+NBTree	76.26	0.24	1	76.26	0.24	1	75.76	0.38	1	76.26	0.24	1	52.69	0.61	6.17
ECSDT-PSO-2ell	51.52	12.61	5	55.56	22.44	5	51.52	2.71	5	51.52	12.61	5	38.98	8.52	5
ECSDT-PSO-4ell	55.05	10.84	9	57.07	19.43	9	58.59	2.19	9	55.05	10.84	9	42.68	7.10	9
ECSDT-PSO-6ell	64.14	6.30	13	65.66	10.84	13	62.63	1.60	13	64.14	6.30	13	48.32	4.58	13
ECSDT-PSO-8ell	69.19	3.77	17	73.74	3.76	17	71.72	0.69	17	69.19	3.77	17	53.11	2.26	17
ECSDT-PSO-10ell	74.24	1.25	21	75.25	1.25	21	76.77	0.28	21	74.24	1.25	21	56.31	1.00	21
ECSDT-GA-2ell	51.52	12.61	5	55.56	22.44	5	51.52	2.71	5	51.52	12.61	5	38.98	8.52	5
ECSDT-GA-4ell	54.55	11.10	9	57.07	19.43	9	57.07	2.16	9	54.55	11.10	9	42.43	7.14	9
ECSDT-GA-6ell	64.65	6.05	13	65.15	10.85	13	63.13	1.55	13	64.65	6.05	13	47.98	4.31	13
ECSDT-GA-8ell	69.19	3.77	17	73.74	3.76	17	71.72	0.69	17	69.19	3.77	17	53.12	2.22	17
ECSDT-GA-10ell	76.26	0.73	21	75.76	0.74	21	76.76	0.28	21	76.26	0.73	21	57.07	0.80	21

Table Apx-C-07: WPBC Cost-based results

Algorithm	IRIS Cost-based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	96.00	1.54	9	96.00	0.82	9	69.33	1.03	9	93.33	0.41	11	94.67	0.14	5	90.00	0.31	7
MetaCost+J48	93.33	1.75	4	94.00	0.84	7	74.00	0.38	3	94.67	0.28	7	93.33	0.18	7	93.33	0.09	9
C.S.C+NBTree	84.00	2.44	9	80.67	0.55	9	84.67	0.15	9	94.67	0.40	9	95.33	0.13	9	93.33	0.13	9
MetaCost+NBTree	84.00	2.62	11	85.33	0.27	9	68.67	1.21	5	91.33	0.51	9	90.67	0.15	1	90.00	0.19	9
ECSDT-PSO-3ell	82.66	1.73	7	86.00	0.50	7	74.00	1.58	7	92.00	0.57	7	90.00	0.13	7	88.00	0.24	7
ECSDT-PSO-4ell	90.66	0.93	9	84.67	0.39	9	86.66	0.13	9	77.33	0.23	9	94.00	0.09	9	94.00	0.12	9
ECSDT-PSO-5ell	90.66	0.93	13	84.67	0.39	13	89.33	0.77	13	72.00	0.28	13	94.00	0.09	13	88.66	0.17	13
ECSDT-PSO-6ell	85.33	1.47	17	90.00	0.22	17	90.00	0.10	17	77.33	0.23	17	96.66	0.06	17	93.33	0.07	17
ECSDT-PSO-7ell	89.33	1.07	21	87.33	0.25	21	86.00	0.80	21	73.33	0.27	21	90.66	0.12	21	91.33	0.15	21
ECSDT-GA-3ell	82.66	1.73	7	86.00	0.50	7	74.00	1.58	7	93.33	0.47	7	90.00	0.13	7	88.00	0.24	7
ECSDT-GA-4ell	85.33	1.07	9	87.33	0.31	9	89.33	0.77	9	80.00	0.20	9	96.00	0.07	9	90.66	0.09	9
ECSDT-GA-5ell	90.66	0.93	13	88.66	0.23	13	89.33	0.77	13	77.33	0.23	13	94.00	0.09	13	90.00	0.16	13
ECSDT-GA-6ell	85.33	1.07	17	88.66	0.23	17	90.00	0.10	17	80.66	0.19	17	96.66	0.06	17	93.33	0.07	17
ECSDT-GA-7ell	86.66	1.33	21	86.00	0.33	21	86.66	0.13	21	72.00	0.28	21	94.00	0.09	21	90.00	0.16	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	96.00	8.33	7	92.67	12.67	7	94.66	10.33	9	91.41	3.95	8.11
MetaCost+J48	94.67	11.33	7	95.33	8.33	7	94.00	11.33	7	91.85	3.84	6.44
C.S.C+NBTree	94.67	12.00	9	94.67	9.33	9	94.66	10.67	9	90.74	3.98	9
MetaCost+NBTree	92.67	16.33	9	92.67	12.67	9	93.33	12.00	7	87.63	5.11	7.67
ECSDT-PSO-3ell	90.00	21.33	7	89.33	17.33	7	90.66	16.67	7	86.96	6.68	7
ECSDT-PSO-4ell	94.00	8.33	9	96.66	5.00	9	96.66	7.00	9	90.52	2.47	9
ECSDT-PSO-5ell	95.33	9.67	13	96.00	7.00	13	96.66	7.00	13	89.70	2.92	13
ECSDT-PSO-6ell	96.66	7.00	17	96.66	5.00	17	95.33	7.67	17	91.26	2.42	17
ECSDT-PSO-7ell	94.00	8.33	21	96.00	7.00	21	95.33	7.67	21	89.26	2.85	21
ECSDT-GA-3ell	92.66	16.33	7	92.00	15.33	7	88.66	18.33	7	87.48	6.07	7
ECSDT-GA-4ell	96.66	7.33	9	96.66	5.00	9	96.66	7.00	9	90.96	2.43	9
ECSDT-GA-5ell	94.00	8.33	13	95.33	8.00	13	96.00	7.33	13	90.59	2.90	13
ECSDT-GA-6ell	96.66	7.33	17	96.66	5.00	17	96.66	7.00	17	91.62	2.34	17
ECSDT-GA-7ell	93.33	8.67	21	96.66	6.00	21	94.66	10.33	21	88.89	3.04	21

Table Apx-C-8: IRIS Cost-based results

Algorithm	Hays Cost-based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	84.38	0.16	23	74.38	0.37	23	81.88	1.03	23	84.38	0.16	23	74.38	0.37	23	81.88	1.03	23
MetaCost+J48	84.38	0.16	17	77.50	0.23	21	76.25	1.20	21	84.38	0.16	17	77.50	0.23	23	76.25	1.20	19
C.S.C+NBTree	54.38	1.13	13	57.50	0.80	13	64.38	1.10	1	54.38	1.13	13	57.50	0.80	13	64.38	1.10	5
MetaCost+NBTree	59.38	0.41	1	55.00	0.65	1	44.38	2.06	1	59.38	0.41	1	55.00	0.65	1	44.38	2.06	3
ECSDT-PSO-3ell	45.63	2.91	7	43.75	5.40	7	42.50	4.57	7	48.13	1.36	7	48.75	1.98	7	43.75	2.69	7
ECSDT-PSO-6ell	48.13	1.53	13	51.88	2.51	13	46.25	4.19	13	53.13	0.98	13	56.25	1.16	13	52.50	1.88	13
ECSDT-PSO-9ell	51.25	1.16	19	52.50	2.45	19	54.38	3.38	19	60.00	0.76	19	58.13	0.84	19	55.63	1.57	19
ECSDT-PSO-12ell	56.25	0.72	25	58.75	1.14	25	57.50	2.51	25	64.38	0.56	25	64.38	0.54	25	61.25	1.35	25
ECSDT-PSO-15ell	63.13	0.37	31	63.13	1.04	31	68.75	1.38	31	69.38	0.38	31	69.38	0.36	31	62.50	1.10	31
ECSDT-GA-3ell	44.38	2.47	7	45.00	5.14	7	42.50	4.57	7	50.00	1.33	7	48.75	1.98	7	43.75	2.69	7
ECSDT-GA-6ell	48.13	1.53	13	51.88	2.45	13	46.30	4.19	13	53.13	0.95	13	56.25	1.16	13	52.50	1.88	13
ECSDT-GA-9ell	52.50	1.09	19	52.50	2.28	19	55.63	3.26	19	58.75	0.86	19	57.50	0.78	19	56.25	1.51	19
ECSDT-GA-12ell	54.38	0.91	25	59.38	1.08	25	57.50	2.51	25	63.13	0.56	25	64.38	0.54	25	61.25	1.35	25
ECSDT-GA-15ell	64.38	0.36	31	64.38	0.98	31	69.38	1.32	31	70.63	0.29	31	68.13	0.32	31	62.50	1.10	31

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	83.13	28.44	23	83.13	34.69	23	83.13	37.19	23	81.67	16.98	23
MetaCost+J48	80.63	31.56	21	81.88	36.56	27	80.00	55.31	21	80.1	20.84	21.3
C.S.C+NBTree	59.38	60.94	1	62.50	68.75	13	64.38	71.25	13	60.42	33.99	9.67
MetaCost+NBTree	63.13	61.88	1	55.63	75.31	1	63.75	74.06	1	56.88	35.73	1.33
ECSDT-PSO-3ell	45.00	92.81	7	45.63	92.19	7	43.75	102.81	7	45.84	48.97	7
ECSDT-PSO-6ell	47.50	86.56	13	46.25	87.81	13	48.13	94.06	13	50.63	45.41	13
ECSDT-PSO-9ell	48.75	80.00	19	52.50	79.06	19	50.63	89.06	19	54.27	41.88	19
ECSDT-PSO-12ell	52.50	74.69	25	56.88	66.56	25	55.63	79.06	25	59.17	37.13	25
ECSDT-PSO-15ell	59.38	60.94	31	62.50	56.25	31	57.50	75.31	31	63.44	32.39	31
ECSDT-GA-3ell	45.00	92.81	7	44.38	94.69	7	43.75	102.81	7	45.94	49.38	7
ECSDT-GA-6ell	47.50	86.56	13	48.75	82.81	13	46.88	96.56	13	50.84	44.99	13
ECSDT-GA-9ell	47.50	85.31	19	51.88	76.56	19	53.13	84.06	19	54.17	41.51	19
ECSDT-GA-12ell	51.88	75.63	25	56.88	66.56	25	55.63	79.06	25	58.86	37.28	25
ECSDT-GA-15ell	58.13	63.44	31	61.88	56.88	31	56.88	76.56	31	63.03	33.1	31

Table Apx-C-09: Hays Cost-based results

Algorithm	Seeds Cost-based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	89.05	3.02	15	62.86	2.54	15	91.43	1.41	15	89.05	0.40	15	91.90	0.58	15	91.43	0.34	15
MetaCost+J48	86.19	0.74	11	77.62	2.75	13	81.43	1.60	15	86.67	0.26	17	86.19	0.89	13	86.19	0.41	15
C.S.C+NBTree	70.95	0.59	7	70.47	1.75	7	86.19	3.44	7	87.62	0.28	7	92.38	0.37	7	88.57	0.54	7
MetaCost+NBTree	67.14	0.46	7	61.90	2.48	7	70.95	3.38	15	87.62	0.22	1	86.19	0.67	3	88.10	0.40	5
ECSDT-PSO-3ell	60.48	3.40	7	54.76	3.02	7	56.19	3.44	7	59.05	1.05	7	60.95	1.06	7	61.90	1.10	7
ECSDT-PSO-6ell	73.81	1.93	13	66.67	2.13	13	66.19	2.52	13	68.57	0.75	13	65.24	1.02	13	65.24	0.92	13
ECSDT-PSO-9ell	76.19	1.31	19	73.33	1.81	19	77.14	1.86	19	75.24	0.52	19	71.90	0.81	19	70.95	0.65	19
ECSDT-PSO-12ell	80.00	0.71	25	77.14	1.43	25	82.86	1.29	25	78.57	0.40	25	77.14	0.51	25	80.95	0.32	25
ECSDT-PSO-15ell	82.86	0.39	31	78.57	1.40	31	85.71	1.21	31	87.14	0.21	31	81.43	0.32	31	83.33	0.30	31
ECSDT-GA-3ell	60.48	3.40	7	55.71	3.14	7	58.10	3.50	7	60.00	1.00	7	60.95	1.06	7	61.90	1.10	7
ECSDT-GA-6ell	72.86	1.85	13	66.67	2.13	13	65.24	2.58	13	68.10	0.80	13	64.29	1.05	13	65.24	0.92	13
ECSDT-GA-9ell	75.71	1.31	19	74.76	1.88	19	77.10	1.86	19	73.81	0.60	19	70.48	0.86	19	71.43	0.63	19
ECSDT-GA-12ell	80.48	0.75	25	76.19	1.49	25	83.33	1.24	25	79.05	0.38	25	75.71	0.54	25	79.52	0.30	25
ECSDT-GA-15ell	83.33	0.47	31	80.95	1.50	31	85.70	1.21	31	87.14	0.21	31	80.95	0.33	31	82.86	0.30	31

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	91.90	15.48	15	91.90	17.86	15	91.90	15.24	15	87.44	6.32	15
MetaCost+J48	87.62	24.76	15	90.00	20.95	15	88.10	21.19	19	85.98	8.17	14
C.S.C+NBTree	90.95	20.71	7	91.43	18.10	7	91.43	18.10	7	85.18	7.10	8
MetaCost+NBTree	87.14	25.00	7	89.05	23.33	7	87.14	24.29	7	81.06	8.91	7
ECSDT-PSO-3ell	55.24	75.24	7	52.38	80.48	7	56.19	74.29	7	60.9	27.01	7
ECSDT-PSO-6ell	61.90	63.10	13	57.62	70.71	13	61.90	59.52	13	64.6	22.51	13
ECSDT-PSO-9ell	67.62	50.48	19	65.71	57.14	19	71.90	44.05	19	71.11	17.63	19
ECSDT-PSO-12ell	74.76	39.29	25	76.19	37.14	25	79.52	30.71	25	77.72	12.42	25
ECSDT-PSO-15ell	81.90	27.14	31	84.76	22.86	31	87.61	19.52	31	82.8	8.15	31
ECSDT-GA-3ell	55.24	75.24	7	52.38	80.48	7	56.19	74.29	7	61.37	27.02	7
ECSDT-GA-6ell	63.81	60.24	13	58.57	69.29	13	60.48	62.38	13	64.55	22.36	13
ECSDT-GA-9ell	68.57	49.05	19	63.81	60.00	19	70.48	44.29	19	70.68	17.83	19
ECSDT-GA-12ell	72.86	40.71	25	78.10	34.29	25	79.52	30.71	25	77.3	12.27	25
ECSDT-GA-15ell	80.95	30.48	31	84.76	22.86	31	88.09	18.81	31	82.91	8.46	31

Table Apx-C-010: Seeds Cost-based results

Algorithm	Tae Cost-based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	54.30	1.85	67	54.96	1.82	67	50.99	2.10	67	57.61	83.44	67	54.96	1.82	67	56.95	82.78	67
MetaCost+J48	46.35	2.55	33	51.65	1.56	35	45.03	2.03	45	59.60	78.81	33	51.65	1.56	37	54.30	85.43	35
C.S.C+NBTree	40.39	1.91	7	47.01	1.49	7	46.35	1.83	7	56.29	87.42	7	47.01	1.49	7	56.29	86.09	7
MetaCost+NBTree	36.42	2.18	11	45.03	1.66	3	40.39	1.97	11	50.33	97.02	11	45.03	1.66	7	53.64	85.76	3
ECSDT-PSO-3ell	28.47	3.76	7	30.46	2.81	7	27.80	2.89	7	32.50	123.84	7	30.46	2.81	7	30.50	127.48	7
ECSDT-PSO-6ell	32.45	2.74	13	33.77	2.58	13	33.11	2.60	13	36.40	114.24	13	33.77	2.58	13	34.40	114.57	13
ECSDT-PSO-9ell	35.10	2.44	19	42.38	1.89	19	35.76	2.23	19	44.40	97.68	19	42.38	1.89	19	43.00	97.35	19
ECSDT-PSO-12ell	40.40	1.68	25	43.71	1.49	25	40.40	1.60	25	55.60	73.18	25	43.71	1.49	25	53.60	76.16	25
ECSDT-PSO-15ell	50.33	1.19	31	50.33	1.16	31	46.36	1.30	31	61.60	62.25	31	50.33	1.16	31	56.30	70.86	31
ECSDT-GA-3ell	28.47	3.76	7	31.13	2.78	7	27.80	2.89	7	33.80	121.85	7	31.13	2.78	7	32.50	124.50	7
ECSDT-GA-6ell	33.77	2.67	13	33.77	2.58	13	31.13	2.73	13	36.40	114.24	13	33.77	2.58	13	35.80	112.25	13
ECSDT-GA-9ell	34.44	2.45	19	42.38	1.89	19	34.44	2.30	19	44.40	97.68	19	42.38	1.89	19	43.00	97.35	19
ECSDT-GA-12ell	40.40	1.68	25	45.70	1.42	25	40.40	1.60	25	54.30	75.17	25	45.70	1.42	25	53.60	76.16	25
ECSDT-GA-15ell	50.33	1.19	31	50.33	1.16	31	45.03	1.37	31	61.60	62.25	31	50.33	1.16	31	56.30	70.86	31

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	57.61	83.44	67	56.95	82.78	67	60.92	79.80	67	55.96	41.97	67
MetaCost+J48	59.60	78.81	51	54.30	85.43	45	55.62	90.07	51	52.09	43.41	42
C.S.C+NBTree	56.29	87.42	7	56.29	86.09	7	58.94	78.48	7	50.88	42.87	7
MetaCost+NBTree	50.33	97.02	11	53.64	85.76	11	54.96	87.09	7	46.8	45.94	8.33
ECSDT-PSO-3ell	32.50	123.84	7	30.50	127.48	7	32.45	117.88	7	30.36	63.11	7
ECSDT-PSO-6ell	36.40	114.24	13	34.40	114.57	13	38.41	105.96	13	34.76	57.11	13
ECSDT-PSO-9ell	44.40	97.68	19	43.00	97.35	19	45.00	92.72	19	40.94	49.05	19
ECSDT-PSO-12ell	55.60	73.18	25	53.60	76.16	25	57.00	68.87	25	48.45	37.16	25
ECSDT-PSO-15ell	61.60	62.25	31	56.30	70.86	31	63.60	57.95	31	54.75	32.45	31
ECSDT-GA-3ell	33.80	121.85	7	32.50	124.50	7	32.50	117.88	7	31.03	62.28	7
ECSDT-GA-6ell	36.40	114.24	13	35.80	112.25	13	39.74	104.97	13	35.1	56.57	13
ECSDT-GA-9ell	44.40	97.68	19	43.00	97.35	19	44.40	94.04	19	40.51	49.28	19
ECSDT-GA-12ell	54.30	75.17	25	53.60	76.16	25	56.30	70.20	25	48.45	37.70	25
ECSDT-GA-15ell	61.60	62.25	31	56.30	70.86	31	63.60	57.95	31	54.53	32.46	31

Table Apx-C-011: Tae Cost-based results

Algorithm	Thyroid Cost-based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	93.02	0.27	17	91.16	0.47	17	92.09	0.40	17	92.09	15.12	17	92.09	16.74	17	92.09	15.58	17
MetaCost+J48	89.30	0.24	11	88.83	0.47	11	91.16	0.43	11	91.16	16.51	17	91.16	18.14	11	90.70	18.84	13
C.S.C+NBTree	90.69	0.44	7	92.56	0.25	7	93.02	0.33	7	93.02	14.88	7	93.48	12.33	7	93.02	11.16	7
MetaCost+NBTree	92.09	0.33	1	89.77	0.34	1	92.09	0.39	1	90.23	20.23	9	90.70	17.44	1	90.23	20.00	9
ECSDT-PSO-3ell	78.60	0.70	7	82.79	0.67	7	82.33	0.67	7	83.72	28.84	7	84.19	29.30	7	82.33	28.37	7
ECSDT-PSO-6ell	85.58	0.43	13	86.98	0.52	13	86.98	0.49	13	86.05	23.72	13	87.44	20.00	13	85.58	21.86	13
ECSDT-PSO-9ell	88.37	0.35	19	91.62	0.42	19	91.62	0.32	19	92.55	11.63	19	90.23	15.35	19	91.16	13.26	19
ECSDT-PSO-12ell	88.37	0.27	25	88.84	0.22	25	90.69	0.27	25	94.40	8.84	25	92.09	11.86	25	93.95	9.07	25
ECSDT-PSO-15ell	93.95	0.17	31	92.55	0.19	31	94.88	0.22	31	94.40	8.37	31	93.95	9.07	31	93.95	9.07	31
ECSDT-GA-3ell	78.60	0.70	7	82.79	0.67	7	82.33	0.67	7	83.72	28.84	7	87.44	20.00	7	82.33	28.37	7
ECSDT-GA-6ell	87.91	0.44	13	88.37	0.47	13	85.58	0.52	13	86.51	23.49	13	90.23	15.35	13	86.51	22.33	13
ECSDT-GA-9ell	89.77	0.30	19	88.37	0.32	19	90.23	0.30	19	93.95	10.47	19	92.09	11.86	19	92.56	11.16	19
ECSDT-GA-12ell	90.69	0.22	25	90.69	0.24	25	91.62	0.24	25	93.95	9.07	25	93.95	9.07	25	93.95	9.07	25
ECSDT-GA-15ell	95.81	0.20	31	93.95	0.17	31	93.95	0.19	31	94.88	7.67	31	92.55	11.16	31	94.42	8.37	31

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	91.16	0.47	17	92.09	0.40	17	92.09	15.12	17	92.09	8.10	17
MetaCost+J48	88.83	0.47	17	91.16	0.43	13	91.16	16.51	17	90.39	9.11	15
C.S.C+NBTree	92.56	0.25	7	93.02	0.33	7	93.02	14.88	7	92.63	6.57	7
MetaCost+NBTree	89.77	0.34	9	92.09	0.39	9	90.23	20.23	11	90.85	9.79	8
ECSDT-PSO-3ell	82.79	0.67	7	82.33	0.67	7	83.72	28.84	7	82.33	14.76	7
ECSDT-PSO-6ell	86.98	0.52	13	86.98	0.49	13	86.05	23.72	13	86.44	11.17	13
ECSDT-PSO-9ell	91.62	0.42	19	91.62	0.32	19	92.55	11.63	19	90.93	6.89	19
ECSDT-PSO-12ell	88.84	0.22	25	90.69	0.27	25	94.40	8.84	25	91.39	5.09	25
ECSDT-PSO-15ell	92.55	0.19	31	94.88	0.22	31	94.40	8.37	31	93.95	4.51	31
ECSDT-GA-3ell	82.79	0.67	7	82.33	0.67	7	83.72	28.84	7	82.87	13.21	7
ECSDT-GA-6ell	88.37	0.47	13	85.58	0.52	13	86.51	23.49	13	87.52	10.43	13
ECSDT-GA-9ell	88.37	0.32	19	90.23	0.30	19	93.95	10.47	19	91.16	5.73	19
ECSDT-GA-12ell	90.69	0.24	25	91.62	0.24	25	93.95	9.07	25	92.48	4.65	25
ECSDT-GA-15ell	93.95	0.17	31	93.95	0.19	31	94.88	7.67	31	94.26	4.63	31

Table Apx-C-12: Thyroid Cost-based results

Algorithm	Glass Cost-based Results																				
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6			Average		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Acc	Acc	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	96.73	12.67	11	96.73	10.29	11	96.73	8.27	11	96.73	3.33	11	96.73	10.14	11	96.73	9.64	11	96.73	9.06	11
MetaCost+J48	95.32	4.45	11	95.79	6.68	11	97.66	7.34	11	93.93	4.64	11	93.46	6.52	11	96.26	2.72	11	95.40	5.39	11
C.S.C+NBTree	50.47	22.75	7	64.02	11.58	7	82.71	12.50	7	88.32	4.96	7	81.31	5.98	7	55.61	14.74	7	70.40	12.09	7
MetaCost+NBTree	34.58	13.41	11	55.14	11.29	9	80.37	24.71	5	85.51	7.44	7	80.84	10.74	9	50.47	14.39	13	64.49	13.66	9
ECSDT-PSO-6ell	52.34	12.83	13	50.47	16.68	13	54.70	87.37	13	53.27	46.98	13	52.80	19.62	13	51.40	9.06	13	52.50	32.09	13
ECSDT-PSO-8ell	53.74	5.40	17	51.87	5.78	17	55.60	74.75	17	53.74	38.10	17	55.61	16.82	17	54.67	7.43	17	54.21	24.71	17
ECSDT-PSO-10ell	58.88	3.48	21	56.07	4.60	21	60.75	44.38	21	59.81	32.96	21	61.21	10.98	21	58.88	6.07	21	59.27	17.08	21
ECSDT-PSO-12ell	65.42	1.78	25	64.95	3.25	25	65.42	32.23	25	64.95	16.60	25	62.62	10.28	25	61.68	5.23	25	64.17	11.56	25
ECSDT-PSO-14ell	69.63	0.71	29	71.96	2.06	29	72.43	8.36	29	69.63	11.93	29	69.16	6.54	29	67.29	3.73	29	70.02	5.55	29
ECSDT-GA-6ell	52.34	12.83	13	50.47	16.68	13	53.74	79.39	13	52.34	47.91	13	52.80	19.62	13	53.27	8.50	13	52.49	30.82	13
ECSDT-GA-8ell	54.21	5.30	17	52.34	5.31	17	55.60	74.75	17	54.21	35.76	17	55.14	17.05	17	55.14	7.19	17	54.44	24.23	17
ECSDT-GA-10ell	59.81	3.38	21	56.07	4.60	21	61.68	41.57	21	58.88	33.89	21	58.41	12.38	21	59.35	5.84	21	59.03	16.94	21
ECSDT-GA-12ell	66.82	1.89	25	63.55	3.27	25	69.16	28.49	25	64.95	16.60	25	63.55	9.81	25	63.55	5.98	25	65.26	11.01	25
ECSDT-GA-14ell	71.50	0.81	29	69.16	1.46	29	75.23	7.46	29	70.56	11.00	29	69.16	6.54	29	70.09	4.20	29	70.95	5.24	29

Table Apx-C-013: Glass Cost-based results

Algorithm	Ecoli Cost-based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	31.85	31.19	41	66.07	41.95	41	78.27	57.06	41	77.67	32.95	41	75.30	22.48	41	76.19	50.99	41
MetaCost+J48	32.14	19.55	21	65.18	40.94	31	78.87	33.52	17	78.27	20.02	29	74.40	17.81	25	68.45	42.50	41
C.S.C+NBTree	26.49	20.07	13	63.99	31.42	13	74.70	44.68	13	72.62	28.77	13	66.67	24.49	13	55.06	37.28	13
MetaCost+NBTree	25.00	19.19	9	61.01	34.99	23	76.79	37.24	13	72.62	28.80	13	65.18	21.83	5	58.63	35.16	9
ECSDT-PSO-6ell	21.13	13.61	13	50.30	30.12	13	60.42	42.09	13	59.52	24.95	13	56.55	24.74	13	51.49	48.32	13
ECSDT-PSO-8ell	22.92	11.96	17	55.65	29.33	17	64.88	36.70	17	62.20	21.15	17	58.33	19.79	17	52.08	42.09	17
ECSDT-PSO-10ell	24.11	11.06	21	56.85	27.10	21	66.67	35.51	21	66.37	18.74	21	61.90	15.70	21	54.76	36.10	21
ECSDT-PSO-12ell	23.81	10.62	25	61.31	24.67	25	69.35	30.39	25	71.13	16.87	25	68.15	11.78	25	63.69	30.57	25
ECSDT-PSO-14ell	26.79	10.30	29	60.12	24.81	29	72.92	35.99	29	75.60	14.62	29	72.02	13.11	29	70.83	27.75	29
ECSDT-GA-6ell	22.02	13.45	13	51.19	29.52	13	60.40	40.57	13	60.12	24.50	13	56.55	24.74	13	52.08	47.87	13
ECSDT-GA-8ell	23.21	12.59	17	55.95	29.03	17	65.77	36.02	17	64.58	20.69	17	58.63	18.84	17	54.17	42.41	17
ECSDT-GA-10ell	24.11	11.07	21	58.63	25.72	21	69.05	31.29	21	70.24	18.24	21	63.99	16.71	21	58.63	35.21	21
ECSDT-GA-12ell	24.70	10.36	25	60.42	24.07	25	67.56	30.07	25	72.92	16.63	25	71.13	13.59	25	64.88	30.06	25
ECSDT-GA-14ell	25.89	10.27	29	62.80	23.92	29	73.81	34.50	29	74.70	14.69	29	69.94	12.59	29	69.05	27.93	29

Algorithm	Ratio - 7			Ratio - 8			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	69.35	40.40	41	57.44	22.56	41	66.52	37.45	41
MetaCost+J48	62.50	31.65	43	60.71	16.78	45	65.07	27.85	32
C.S.C+NBTree	47.32	24.06	13	38.39	19.14	13	55.65	28.74	13
MetaCost+NBTree	50.60	27.60	11	14.88	32.12	19	53.09	29.62	13
ECSDT-PSO-6ell	44.35	35.61	13	43.75	22.17	13	48.44	30.20	13
ECSDT-PSO-8ell	48.51	28.40	17	46.70	20.81	17	51.41	26.28	17
ECSDT-PSO-10ell	50.60	28.02	21	48.81	14.41	21	53.76	23.33	21
ECSDT-PSO-12ell	55.95	23.85	25	56.85	11.75	25	58.78	20.06	25
ECSDT-PSO-14ell	63.99	20.50	29	61.61	11.56	29	62.99	19.83	29
ECSDT-GA-6ell	45.24	35.28	13	43.80	22.17	13	48.93	29.76	13
ECSDT-GA-8ell	47.92	28.45	17	44.35	21.72	17	51.82	26.22	17
ECSDT-GA-10ell	55.65	24.60	21	49.70	14.34	21	56.25	22.15	21
ECSDT-GA-12ell	56.85	23.81	25	54.17	11.91	25	59.08	20.06	25
ECSDT-GA-14ell	62.80	25.01	29	60.12	11.62	29	62.39	20.07	29

Table Apx-C-014: Ecoli Cost-based results

APPENDIX – D: Results Of the Empirical Comparison Based on Accuracy and Cost

This appendix presents the results obtained by the ECSDT when adopting the third aspect which considers both accuracy and cost as a multi-objective function for the evaluation.

Algorithm	Bupa Accuracy + Cost based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio - 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	46.67	1.08	51	43.77	20.81	51	43.77	20.81	51	43.77	20.81	51	43.77	41.10	51	66.96	0.72	51
MetaCost+J48	50.14	0.81	37	42.03	0.58	1	42.03	0.58	1	42.03	0.58	1	42.03	0.58	1	61.74	0.49	23
C.S.C+NBTree	42.32	0.66	11	42.03	0.58	11	42.03	0.58	11	42.03	0.58	11	42.03	0.58	11	57.68	0.48	11
MetaCost+NBTree	43.19	0.57	1	42.03	0.58	1	42.03	0.58	1	42.03	0.58	1	42.03	0.58	1	57.97	0.42	1
ECSDT-PSO-2ell	44.06	0.74	5	42.60	1.14	5	42.89	1.43	5	41.44	4.92	5	43.76	12.14	5	51.30	1.09	5
ECSDT-PSO-4ell	48.69	0.59	9	46.66	0.79	9	47.24	1.39	9	44.63	3.45	9	48.11	9.21	9	65.79	0.55	9
ECSDT-PSO-6ell	49.11	0.63	13	45.88	0.82	13	48.40	0.80	13	47.24	3.42	13	46.95	6.32	13	66.95	0.54	13
ECSDT-PSO-8ell	50.72	0.60	17	47.24	0.81	17	50.14	0.79	17	49.56	3.40	17	51.01	6.28	17	68.69	0.44	17
ECSDT-PSO-10ell	54.78	0.56	21	49.56	0.65	21	52.46	0.76	21	49.85	1.95	21	50.43	3.39	21	70.43	0.40	21
ECSDT-GA-2ell	45.21	0.73	5	41.76	1.31	5	43.76	1.42	5	42.89	7.81	5	43.76	12.14	5	54.49	0.85	5
ECSDT-GA-4ell	48.52	0.64	9	46.08	0.82	9	49.27	0.79	9	46.96	4.87	9	47.82	9.21	9	65.50	0.58	9
ECSDT-GA-6ell	53.23	0.59	13	48.53	1.08	13	47.24	0.81	13	50.43	4.83	13	48.69	9.20	13	63.76	0.52	13
ECSDT-GA-8ell	53.04	0.60	17	47.64	0.66	17	49.56	0.79	17	48.98	4.85	17	50.14	6.29	17	69.27	0.46	17
ECSDT-GA-10ell	55.65	0.57	21	51.30	0.63	21	52.75	0.76	21	51.30	1.93	21	52.46	3.37	21	71.30	0.39	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Ratio - 10			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	63.80	2.21	51	63.77	4.09	51	63.77	19.17	51	63.77	38.01	51	54.378	13.43	51
MetaCost+J48	58.00	0.56	7	57.97	0.42	1	57.97	0.42	1	57.97	0.42	1	51.275	0.56	7.4
C.S.C+NBTree	57.70	0.57	11	57.97	0.42	11	57.97	0.42	11	57.97	0.42	11	50.028	0.53	11
MetaCost+NBTree	58.00	0.42	1	57.97	0.42	1	57.97	0.42	1	57.97	0.42	1	50.115	0.50	1
ECSDT-PSO-2ell	53.33	2.74	5	54.20	5.05	5	52.75	26.51	5	54.78	38.10	5	48.11	9.39	5
ECSDT-PSO-4ell	63.76	0.79	9	61.44	1.25	9	64.63	7.59	9	65.50	14.82	9	55.65	4.04	9
ECSDT-PSO-6ell	64.34	0.78	13	63.76	0.94	13	64.92	4.69	13	66.08	9.03	13	56.36	2.80	13
ECSDT-PSO-8ell	66.66	0.62	17	64.29	0.92	17	66.37	2.94	17	63.76	6.15	17	57.84	2.30	17
ECSDT-PSO-10ell	68.98	0.45	21	66.66	0.91	21	68.40	1.76	21	67.53	3.22	21	59.91	1.40	21
ECSDT-GA-2ell	55.65	2.15	5	54.49	4.76	5	57.97	22.12	5	55.07	29.41	5	49.51	8.27	5
ECSDT-GA-4ell	64.92	0.78	9	64.05	0.93	9	65.21	4.69	9	65.21	9.03	9	56.35	3.23	9
ECSDT-GA-6ell	62.89	0.66	13	63.76	0.94	13	63.67	6.15	13	65.79	11.92	13	56.80	3.67	13
ECSDT-GA-8ell	68.40	0.46	17	65.50	0.92	17	68.40	1.76	17	65.79	3.24	17	58.67	2.00	17
ECSDT-GA-10ell	70.72	0.43	21	68.40	0.60	21	70.43	1.74	21	66.66	3.23	21	61.10	1.37	21

Table Apx-D-01: Bupa Accuracy + Cost based results

Algorithm	Hepatitis Accuracy + Cost based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	79.49	0.55	9	64.1	1.62	5	58.97	1.68	5	19.23	0.81	1	19.23	0.81	1	80.77	0.19	1
MetaCost+J48	70.51	0.64	13	60.26	0.40	9	32.05	1.95	1	19.23	0.81	1	19.23	0.81	1	82.05	0.18	1
C.S.C+NBTree	70.51	0.64	5	39.74	1.23	5	28.21	0.72	5	25.64	0.74	5	19.23	0.81	5	80.77	0.31	5
MetaCost+NBTree	73.08	0.73	5	50	0.50	5	26.92	0.73	5	19.23	0.81	1	19.23	0.81	3	80.77	0.31	1
ECSDT-PSO-2ell	62.82	0.83	5	57.69	2.31	5	62.82	2.91	5	58.97	13.21	5	56.41	26.05	5	78.20	0.56	5
ECSDT-PSO-4ell	74.35	0.60	9	69.23	1.56	9	69.23	2.85	9	62.82	13.17	9	66.66	25.95	9	82.05	0.41	9
ECSDT-PSO-6ell	80.76	0.42	13	78.20	0.85	13	75.64	1.51	13	64.10	6.76	13	76.92	13.04	13	88.46	0.23	13
ECSDT-PSO-8ell	85.89	0.37	17	78.20	0.85	17	78.20	1.49	17	80.76	6.59	17	73.07	13.08	17	84.61	0.27	17
ECSDT-PSO-10ell	80.76	0.42	21	80.76	0.82	21	78.20	1.49	21	67.94	6.72	21	78.20	13.03	21	88.46	0.23	21
ECSDT-GA-2ell	67.94	0.67	5	65.38	1.60	5	61.53	2.92	5	55.12	13.24	5	58.97	26.03	5	80.76	0.54	5
ECSDT-GA-4ell	71.79	0.63	9	70.51	1.55	9	71.79	1.55	9	70.51	13.09	9	65.38	25.96	9	84.61	0.38	9
ECSDT-GA-6ell	82.05	0.41	13	78.20	0.85	13	78.20	1.49	13	70.51	13.09	13	74.35	13.06	13	91.02	0.21	13
ECSDT-GA-8ell	80.76	0.42	17	80.76	0.82	17	78.20	1.49	17	78.20	6.62	17	74.35	13.06	17	85.89	0.26	17
ECSDT-GA-10ell	85.89	0.37	21	82.05	0.81	21	75.64	1.51	21	75.64	6.64	21	76.92	13.04	21	84.61	0.27	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Ratio - 10			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	80.77	0.19	1	80.77	0.19	1	80.77	0.19	1	80.77	0.19	1	64.49	0.64	2.6
MetaCost+J48	80.77	0.19	1	80.77	0.19	1	80.77	0.19	1	80.77	0.19	1	60.64	0.56	3
C.S.C+NBTree	80.77	0.19	5	80.77	0.19	5	80.77	0.19	5	80.77	0.19	5	58.72	0.52	5
MetaCost+NBTree	80.77	0.19	1	80.77	0.19	1	80.77	0.19	1	80.77	0.19	1	59.23	0.47	2.4
ECSDT-PSO-2ell	80.76	2.08	5	76.92	4.04	5	78.20	13.01	5	78.20	25.83	5	69.099	9.08	5
ECSDT-PSO-4ell	82.05	1.44	9	80.76	2.73	9	82.05	6.58	9	82.05	12.99	9	75.125	6.83	9
ECSDT-PSO-6ell	84.61	0.78	13	87.17	1.40	13	82.05	6.58	13	84.61	12.96	13	80.252	4.45	13
ECSDT-PSO-8ell	87.17	0.76	17	87.17	1.40	17	80.76	6.59	17	82.05	12.99	17	81.788	4.44	17
ECSDT-PSO-10ell	87.17	0.76	21	85.89	1.41	21	87.17	6.53	21	85.89	12.95	21	82.044	4.43	21
ECSDT-GA-2ell	80.76	2.08	5	78.02	4.03	5	78.20	13.01	5	80.76	25.81	5	70.744	8.99	5
ECSDT-GA-4ell	80.76	1.45	9	82.05	2.72	9	82.05	6.58	9	82.05	12.99	9	76.15	6.69	9
ECSDT-GA-6ell	84.61	0.78	13	84.61	1.42	13	84.61	6.55	13	88.46	12.92	13	81.662	5.08	13
ECSDT-GA-8ell	85.89	0.77	17	84.61	1.42	17	87.17	6.53	17	87.17	12.94	17	82.3	4.43	17
ECSDT-GA-10ell	91.02	0.72	21	87.17	1.40	21	84.61	6.55	21	82.05	12.99	21	82.56	4.43	21

Table Apx-D-02: Hepatitis Accuracy + Cost based results

Algorithm	Heart Accuracy + Cost based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	68.51	1.11	39	60.00	2.40	39	60.00	4.43	39	60.00	20.73	39	60.00	41.10	39	67.40	0.96	39
MetaCost+J48	66.29	0.64	31	61.11	0.39	15	55.55	0.44	1	55.55	0.44	1	55.55	0.44	1	71.11	0.52	29
C.S.C+NBTree	74.44	0.39	17	67.03	0.51	17	60.37	0.40	17	58.88	0.41	17	55.55	0.44	17	66.29	0.60	17
MetaCost+NBTree	70.00	0.50	11	56.66	0.43	1	55.55	0.44	1	55.55	0.44	1	55.55	0.44	1	62.96	0.57	11
ECSDT-PSO-2ell	54.44	1.62	5	57.03	2.61	5	58.89	7.74	5	56.66	30.00	5	53.33	30.07	5	57.41	1.26	5
ECSDT-PSO-4ell	65.18	0.61	9	63.33	1.46	9	62.96	4.04	9	63.33	11.46	9	56.29	22.64	9	64.07	0.76	9
ECSDT-PSO-6ell	71.11	0.49	13	68.51	0.68	13	65.93	2.17	13	65.55	4.04	13	65.55	7.74	13	67.41	0.53	13
ECSDT-PSO-8ell	74.81	0.45	17	72.96	0.45	17	70.33	1.76	17	68.88	4.01	17	67.03	7.73	17	73.33	0.47	17
ECSDT-PSO-10ell	77.03	0.36	21	75.92	0.42	21	70.33	1.03	21	72.96	2.12	21	69.26	4.01	21	76.66	0.40	21
ECSDT-GA-2ell	58.52	1.58	5	58.14	2.78	5	57.41	7.03	5	58.89	26.29	5	55.55	26.34	5	60.74	1.06	5
ECSDT-GA-4ell	63.70	0.70	9	65.55	1.43	9	62.96	3.30	9	65.93	9.59	9	59.63	15.20	9	65.18	0.68	9
ECSDT-GA-6ell	71.48	0.49	13	68.51	0.68	13	67.41	1.79	13	65.93	5.89	13	65.93	7.74	13	70.00	0.50	13
ECSDT-GA-8ell	74.81	0.45	17	74.44	0.44	17	72.22	0.64	17	70.37	2.14	17	63.70	7.76	17	73.33	0.43	17
ECSDT-GA-10ell	77.77	0.36	21	76.66	0.41	21	70.33	0.66	21	71.85	2.13	21	67.03	4.03	21	77.77	0.39	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Ratio - 10			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	50.74	1.76	39	48.88	3.08	39	48.88	13.45	39	48.88	26.41	39	57.33	11.54	39
MetaCost+J48	52.22	0.66	1	44.44	0.56	1	44.44	0.56	1	44.44	0.56	1	55.07	0.52	8.2
C.S.C+NBTree	52.22	1.02	17	48.14	1.25	17	45.55	2.39	17	45.18	0.55	17	57.37	0.80	17
MetaCost+NBTree	44.44	0.56	1	44.44	0.56	1	44.44	0.56	1	44.44	0.56	1	53.41	0.51	3
ECSDT-PSO-2ell	49.63	2.32	5	41.85	3.51	5	42.96	15.36	5	42.59	30.17	5	51.48	12.47	5
ECSDT-PSO-4ell	53.33	1.56	9	51.11	1.96	9	47.77	7.91	9	50.37	15.30	9	57.77	6.77	9
ECSDT-PSO-6ell	58.15	1.14	13	59.25	1.14	13	55.92	4.14	13	57.78	7.82	13	63.52	2.99	13
ECSDT-PSO-8ell	65.92	0.52	17	63.33	0.73	17	60.37	2.24	17	58.15	4.12	17	67.51	2.25	17
ECSDT-PSO-10ell	63.70	0.73	21	65.18	0.71	21	65.92	2.19	21	61.48	4.09	21	69.84	1.61	21
ECSDT-GA-2ell	50.74	2.13	5	42.59	3.14	5	43.70	13.50	5	44.44	26.46	5	53.07	11.03	5
ECSDT-GA-4ell	53.33	1.37	9	53.70	1.56	9	50.74	7.89	9	52.59	15.27	9	59.33	5.70	9
ECSDT-GA-6ell	59.25	0.95	13	60.00	0.77	13	54.07	4.16	13	58.52	4.11	13	64.11	2.71	13
ECSDT-GA-8ell	64.81	0.50	17	63.33	0.73	17	61.11	2.24	17	60.00	3.73	17	67.81	1.91	17
ECSDT-GA-10ell	63.70	0.73	21	64.44	0.72	21	63.33	2.21	21	62.96	3.74	21	69.58	1.54	21

Table Apx-D-3: Heart Accuracy + Cost based results

Algorithm	Haberman Accuracy + Cost based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	73.529	0.26	5	73.529	0.26	5	73.529	0.26	5	73.529	0.26	5	73.529	0.26	5	27.777	0.75	5
MetaCost+J48	73.529	0.26	1	73.529	0.26	1	73.529	0.26	1	73.529	0.26	1	73.529	0.26	1	29.738	0.76	21
C.S.C+NBTree	73.529	0.26	3	73.529	0.26	3	73.529	0.26	3	73.529	0.26	3	73.529	0.26	3	26.47	0.74	3
MetaCost+NBTree	73.529	0.26	1	73.529	0.26	1	73.529	0.26	1	73.529	0.26	1	73.529	0.26	1	26.47	0.74	1
ECSDT-PSO-2ell	71.25	0.58	5	72.22	1.24	5	71.90	2.22	5	73.53	5.16	5	73.53	10.08	5	26.80	1.08	5
ECSDT-PSO-4ell	74.51	0.49	9	74.84	0.89	9	74.51	1.55	9	76.47	3.50	9	76.14	6.77	9	30.72	0.93	9
ECSDT-PSO-6ell	77.77	0.40	13	76.80	0.55	13	74.18	0.91	13	76.47	3.50	13	75.82	6.77	13	36.60	0.81	13
ECSDT-PSO-8ell	80.72	0.31	17	79.74	0.52	17	78.76	0.53	17	80.72	1.82	17	78.48	3.48	17	39.87	0.72	17
ECSDT-PSO-10ell	83.33	0.25	21	81.70	0.34	21	82.35	0.50	21	78.76	1.84	21	82.35	3.44	21	40.20	0.69	21
ECSDT-GA-2ell	73.53	0.56	5	72.22	1.40	5	73.20	1.89	5	72.55	5.17	5	73.86	10.06	5	27.45	1.05	5
ECSDT-GA-4ell	75.82	0.45	9	73.86	0.90	9	74.51	1.23	9	76.47	5.13	9	75.82	6.77	9	29.74	0.94	9
ECSDT-GA-6ell	78.76	0.33	13	74.18	0.42	13	76.80	0.88	13	78.76	3.47	13	77.45	6.75	13	37.25	0.77	13
ECSDT-GA-8ell	82.68	0.26	17	76.80	0.39	17	76.80	0.56	17	78.76	1.84	17	78.48	3.48	17	41.18	0.71	17
ECSDT-GA-10ell	83.33	0.25	21	82.35	0.34	21	83.33	0.49	21	81.70	1.81	21	83.01	3.43	21	40.85	0.68	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Ratio - 10			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	26.47	0.90	5	26.47	1.06	5	26.47	2.37	5	26.47	4.00	5	50.13	1.04	5
MetaCost+J48	26.47	0.74	1	26.47	0.74	1	26.47	0.74	1	26.47	0.74	1	50.33	0.50	3
C.S.C+NBTree	26.47	0.74	3	26.47	0.74	3	26.47	0.74	3	26.47	0.74	3	50.00	0.50	3
MetaCost+NBTree	26.47	0.74	1	26.47	0.74	1	26.47	0.74	1	26.47	0.74	1	50.00	0.50	1
ECSDT-PSO-2ell	25.16	2.03	5	26.47	2.68	5	25.49	7.27	5	26.47	13.79	5	49.28	4.61	5
ECSDT-PSO-4ell	28.76	1.67	9	27.78	2.02	9	28.76	5.60	9	27.45	10.52	9	51.99	3.39	9
ECSDT-PSO-6ell	36.27	1.28	13	37.91	1.59	13	34.97	3.91	13	32.68	7.20	13	55.95	2.69	13
ECSDT-PSO-8ell	38.89	0.93	17	39.22	1.25	17	38.24	2.25	17	40.20	3.86	17	59.48	1.57	17
ECSDT-PSO-10ell	41.50	0.75	21	41.83	0.91	21	41.18	2.22	21	38.89	3.88	21	61.21	1.48	21
ECSDT-GA-2ell	26.80	1.69	5	26.80	2.67	5	25.49	7.27	5	26.47	10.53	5	49.84	4.23	5
ECSDT-GA-4ell	28.76	1.67	9	28.10	2.01	9	27.45	5.62	9	29.41	7.24	9	51.99	3.20	9
ECSDT-GA-6ell	38.89	1.09	13	35.95	1.29	13	36.60	3.90	13	37.25	7.16	13	57.19	2.61	13
ECSDT-GA-8ell	36.93	0.95	17	40.85	0.92	17	40.52	2.23	17	40.52	3.86	17	59.35	1.52	17
ECSDT-GA-10ell	42.16	0.74	21	38.56	0.94	21	38.89	2.24	21	42.16	3.84	21	61.63	1.48	21

Table Apx-D-4: Haberman Accuracy + Cost based results

Algorithm	Diabetes Accuracy + Cost based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	65.10	0.60	39	64.71	1.31	39	64.71	2.29	39	64.71	10.10	39	64.71	19.86	39	57.16	0.83	39
MetaCost+J48	70.05	0.44	19	65.10	0.35	1	65.10	0.35	1	65.10	0.35	1	65.10	0.35	1	62.11	0.67	57
C.S.C+NBTree	68.62	0.44	1	65.63	0.41	1	65.10	0.35	1	65.10	0.35	1	65.10	0.35	1	59.90	0.62	1
MetaCost+NBTree	65.63	0.34	7	65.10	0.35	1	65.10	0.35	1	65.10	0.35	1	65.10	0.35	1	55.86	0.58	3
ECSDT-PSO-2ell	60.55	0.80	5	62.50	1.65	5	63.28	2.69	5	62.50	13.37	5	62.50	28.99	5	47.66	0.78	5
ECSDT-PSO-4ell	66.80	0.51	9	65.63	0.73	9	65.23	1.39	9	65.63	4.24	9	66.02	10.75	9	55.47	0.62	9
ECSDT-PSO-6ell	71.88	0.40	13	72.66	0.53	13	71.88	0.80	13	67.97	2.92	13	68.36	5.52	13	64.45	0.47	13
ECSDT-PSO-8ell	74.61	0.32	17	74.61	0.38	17	73.44	0.52	17	73.83	0.91	17	71.88	2.88	17	68.75	0.41	17
ECSDT-PSO-10ell	70.70	0.33	21	71.48	0.35	21	74.61	0.38	21	73.83	0.91	21	73.44	1.57	21	64.06	0.43	21
ECSDT-GA-2ell	61.72	0.73	5	62.11	1.85	5	62.89	2.82	5	63.67	11.41	5	62.50	28.99	5	46.88	0.77	5
ECSDT-GA-4ell	67.58	0.49	9	66.41	0.72	9	66.41	1.37	9	65.63	4.24	9	65.23	10.75	9	56.25	0.60	9
ECSDT-GA-6ell	70.70	0.39	13	70.31	0.55	13	69.53	0.69	13	68.75	2.91	13	70.31	4.28	13	66.80	0.43	13
ECSDT-GA-8ell	74.22	0.32	17	73.83	0.33	17	74.61	0.51	17	72.66	0.92	17	72.27	2.88	17	70.31	0.40	17
ECSDT-GA-10ell	71.09	0.32	21	74.22	0.32	21	72.27	0.41	21	74.60	0.90	21	73.44	1.57	21	68.75	0.42	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Ratio - 10			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	41.02	1.55	39	39.45	2.54	39	39.45	10.35	39	39.45	20.12	39	54.05	6.95	39
MetaCost+J48	44.40	0.62	15	36.20	0.64	1	34.90	0.65	1	34.90	0.65	1	54.30	0.51	9.8
C.S.C+NBTree	44.14	0.75	1	40.76	0.72	1	35.81	0.64	1	34.90	0.65	1	54.50	0.53	1
MetaCost+NBTree	38.15	0.62	3	34.90	0.65	1	34.90	0.65	1	34.90	0.65	1	52.47	0.49	2
ECSDT-PSO-2ell	35.68	2.56	5	35.68	2.71	5	33.33	10.41	5	33.33	18.88	5	49.70	8.28	5
ECSDT-PSO-4ell	41.80	1.67	9	39.45	1.64	9	38.41	5.16	9	41.41	7.09	9	54.59	3.38	9
ECSDT-PSO-6ell	46.88	1.08	13	44.92	1.07	13	43.75	3.16	13	46.48	4.44	13	59.92	2.04	13
ECSDT-PSO-8ell	49.61	0.82	17	47.66	0.78	17	47.66	1.82	17	47.66	3.13	17	62.97	1.20	17
ECSDT-PSO-10ell	51.56	0.61	21	50.39	0.63	21	38.80	0.61	21	40.63	1.89	21	60.95	0.77	21
ECSDT-GA-2ell	35.29	2.24	5	35.68	2.58	5	32.94	9.77	5	34.51	14.96	5	49.82	7.61	5
ECSDT-GA-4ell	43.75	1.46	9	41.02	1.75	9	40.23	5.15	9	42.19	7.08	9	55.47	3.36	9
ECSDT-GA-6ell	47.66	0.91	13	45.70	0.93	13	46.09	2.49	13	48.44	4.42	13	60.43	1.80	13
ECSDT-GA-8ell	50.00	0.82	17	47.27	0.79	17	48.05	1.17	17	48.83	3.11	17	63.21	1.12	17
ECSDT-GA-10ell	52.73	0.60	21	51.17	0.62	21	37.24	0.63	21	41.80	1.88	21	61.73	0.77	21

Table Apx-D-05: Diabetes Accuracy + Cost based results

Algorithm	WDBC Accuracy + Cost based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	92.97	0.31	25	92.26	1.28	25	66.43	2.42	25	45.69	8.44	25	45.69	16.34	25	92.97	0.40	25
MetaCost+J48	90.68	0.19	23	88.57	0.63	23	86.64	1.00	27	37.60	0.62	1	37.26	0.63	1	92.97	0.12	17
C.S.C+NBTree	92.26	0.35	23	87.87	1.24	23	78.20	1.78	23	71.88	6.42	23	72.06	10.81	23	93.14	0.31	23
MetaCost+NBTree	89.45	0.20	23	89.10	0.71	27	88.40	0.81	7	44.81	3.18	19	39.37	0.61	5	91.73	0.10	19
ECSDT-PSO-2ell	75.75	0.56	5	60.98	2.28	5	55.71	3.57	5	67.31	0.72	5	73.64	2.85	5	75.22	4.60	5
ECSDT-PSO-4ell	78.91	0.45	9	72.58	1.82	9	63.62	3.15	9	76.8	0.55	9	78.91	2.36	9	81.55	3.66	9
ECSDT-PSO-6ell	85.24	0.38	13	81.55	1.48	13	70.47	2.73	13	85.76	0.32	13	85.76	1.52	13	86.82	2.74	13
ECSDT-PSO-8ell	88.93	0.30	17	89.46	0.97	17	77.86	1.96	17	91.56	0.21	17	90.51	0.96	17	91.04	1.48	17
ECSDT-PSO-10ell	91.56	0.21	21	90.51	0.61	21	86.29	1.01	21	93.67	0.16	21	93.67	0.58	21	94.2	0.75	21
ECSDT-GA-2ell	75.75	0.56	5	62.57	2.10	5	57.82	3.38	5	66.78	0.74	5	73.64	2.85	5	76.8	4.41	5
ECSDT-GA-4ell	77.33	0.51	9	72.58	1.82	9	61.51	3.17	9	77.33	0.51	9	77.86	2.46	9	80.49	3.67	9
ECSDT-GA-6ell	87.35	0.36	13	82.6	1.38	13	70.47	2.73	13	85.76	0.32	13	84.18	1.63	13	85.76	2.75	13
ECSDT-GA-8ell	88.4	0.32	17	88.93	0.97	17	75.22	2.16	17	90.51	0.22	17	91.04	0.86	17	93.67	1.28	17
ECSDT-GA-10ell	91.56	0.21	21	90.51	0.61	21	85.24	1.02	21	94.73	0.12	21	93.67	0.58	21	92.09	0.78	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Ratio - 10			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	92.61	1.80	25	92.61	3.55	25	92.61	17.61	25	92.61	35.19	25	88.31	1.63	25
MetaCost+J48	92.97	0.33	21	92.09	0.60	23	73.28	3.43	19	62.74	2.13	15	90.65	0.48	22.3
C.S.C+NBTree	88.92	1.14	23	79.78	1.59	23	72.40	4.66	23	72.58	7.30	23	86.7	1.07	23
MetaCost+NBTree	89.98	0.53	17	85.76	0.84	5	66.60	0.33	3	63.09	0.37	15	89.07	0.53	16.3
ECSDT-PSO-2ell	73.64	0.50	5	58.88	1.70	5	55.71	3.57	5	73.64	2.85	5	68.1	2.43	5
ECSDT-PSO-4ell	76.8	0.39	9	70.47	1.33	9	63.62	3.15	9	78.91	2.36	9	75.4	2.00	9
ECSDT-PSO-6ell	83.13	0.29	13	79.44	1.07	13	70.47	2.73	13	85.76	1.52	13	82.6	1.53	13
ECSDT-PSO-8ell	86.82	0.20	17	86.29	0.65	17	77.86	1.96	17	90.51	0.96	17	88.23	0.98	17
ECSDT-PSO-10ell	88.93	0.17	21	88.4	0.46	21	86.29	1.01	21	93.67	0.58	21	91.65	0.55	21
ECSDT-GA-2ell	73.64	0.50	5	58.88	1.70	5	57.82	3.38	5	73.64	2.85	5	68.89	2.34	5
ECSDT-GA-4ell	76.8	0.39	9	70.47	1.33	9	61.51	3.17	9	77.86	2.46	9	74.52	2.02	9
ECSDT-GA-6ell	85.76	0.27	13	79.96	0.98	13	70.47	2.73	13	84.18	1.63	13	82.69	1.53	13
ECSDT-GA-8ell	84.18	0.19	17	88.4	0.72	17	75.22	2.16	17	91.04	0.86	17	87.96	0.97	17
ECSDT-GA-10ell	88.93	0.17	21	90.51	0.61	21	85.24	1.02	21	93.67	0.58	21	91.3	0.55	21

Table Apx-D-06: WDBC Accuracy + Cost based results

Algorithm	WPBC Accuracy + Cost based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	72.72	1.41	21	29.79	3.67	21	29.79	6.70	21	29.79	3.67	21	29.79	6.70	21	73.23	1.18	21
MetaCost+J48	44.95	1.14	25	37.87	2.85	31	29.80	3.70	9	37.87	2.85	1	29.80	3.70	1	76.76	0.28	1
C.S.C+NBTree	47.98	1.25	11	23.23	1.01	11	23.23	1.27	11	23.23	1.01	11	23.23	1.27	11	75.76	0.33	11
MetaCost+NBTree	39.39	1.29	11	24.75	0.75	1	23.73	0.76	1	24.75	0.75	1	23.73	0.76	1	75.76	0.38	1
ECSDT-PSO-2ell	28.28	2.08	5	25.76	4.95	5	23.70	8.76	5	51.52	2.71	5	53.54	12.34	5	53.54	22.96	5
ECSDT-PSO-4ell	33.84	1.66	9	29.29	3.68	9	28.79	7.71	9	57.58	2.24	9	56.57	11.32	9	57.07	19.43	9
ECSDT-PSO-6ell	37.88	1.30	13	33.84	2.64	13	31.82	5.68	13	65.66	1.62	13	67.17	6.52	13	67.17	9.33	13
ECSDT-PSO-8ell	43.43	1.11	17	33.33	1.90	17	33.33	3.67	17	73.23	0.86	17	73.23	3.48	17	71.72	4.78	17
ECSDT-PSO-10ell	45.96	0.99	21	37.88	1.11	21	36.36	2.14	21	77.78	0.31	21	78.79	1.70	21	77.78	1.22	21
ECSDT-GA-2ell	29.80	2.02	5	25.76	4.95	5	24.75	8.25	5	53.54	2.60	5	53.54	12.34	5	55.56	22.44	5
ECSDT-GA-4ell	33.84	1.66	9	28.79	3.68	9	27.78	7.72	9	58.59	2.19	9	57.58	11.07	9	58.08	18.92	9
ECSDT-GA-6ell	38.89	1.25	13	33.33	2.65	13	32.32	5.18	13	64.14	1.68	13	66.16	6.53	13	65.66	11.34	13
ECSDT-GA-8ell	42.93	1.12	17	35.35	1.73	17	35.86	3.14	17	73.23	0.86	17	74.75	3.22	17	73.74	4.26	17
ECSDT-GA-10ell	46.46	0.99	21	37.88	1.36	21	36.87	1.63	21	78.79	0.30	21	78.79	1.20	21	78.28	1.22	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Ratio - 10			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	73.23	4.47	21	73.23	8.77	21	73.23	1.18	21	73.23	4.47	21	58.67	4.37	11.5
MetaCost+J48	75.76	0.49	1	76.26	0.24	1	76.76	0.28	1	75.76	0.49	1	56.9	1.45	12.8
C.S.C+NBTree	75.76	0.49	11	75.76	0.74	11	75.76	0.33	11	75.76	0.49	11	53.62	0.85	8.5
MetaCost+NBTree	76.26	0.24	1	76.26	0.24	1	75.76	0.38	1	76.26	0.24	1	52.69	0.61	6.17
ECSDT-PSO-2ell	28.28	2.08	5	24.75	4.46	5	51.52	2.71	5	53.54	12.34	5	39.39	8.97	5
ECSDT-PSO-4ell	30.81	1.46	9	27.78	2.95	9	57.58	2.24	9	56.57	11.32	9	43.86	7.67	9
ECSDT-PSO-6ell	34.85	1.11	13	30.81	1.93	13	65.66	1.62	13	67.17	6.52	13	50.59	4.51	13
ECSDT-PSO-8ell	40.40	1.01	17	32.32	1.67	17	73.23	0.86	17	73.23	3.48	17	54.71	2.63	17
ECSDT-PSO-10ell	43.43	0.93	21	34.85	1.15	21	77.78	0.31	21	78.79	1.70	21	59.09	1.25	21
ECSDT-GA-2ell	28.28	2.08	5	24.75	4.46	5	53.54	2.60	5	53.54	12.34	5	40.49	8.77	5
ECSDT-GA-4ell	31.31	1.46	9	27.78	2.95	9	58.59	2.19	9	57.58	11.07	9	44.11	7.54	9
ECSDT-GA-6ell	35.86	1.05	13	29.80	2.19	13	64.14	1.68	13	66.16	6.53	13	50.08	4.77	13
ECSDT-GA-8ell	39.90	0.96	17	32.83	1.41	17	73.23	0.86	17	74.75	3.22	17	55.98	2.39	17
ECSDT-GA-10ell	44.44	0.96	21	35.35	0.89	21	78.79	0.30	21	78.79	1.20	21	59.51	1.12	21

Table Apx-D-07: WPBC Accuracy + Cost based results

Algorithm	IRIS Accuracy + Cost based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	96.00	1.54	9	96.00	0.82	9	69.33	1.03	9	93.33	0.41	11	94.67	0.14	5	90.00	0.31	7
MetaCost+J48	93.33	1.75	4	94.00	0.84	7	74.00	0.38	3	94.67	0.28	7	93.33	0.18	7	93.33	0.09	9
C.S.C+NBTree	84.00	2.44	9	80.67	0.55	9	84.67	0.15	9	94.67	0.40	9	95.33	0.13	9	93.33	0.13	9
MetaCost+NBTree	84.00	2.62	11	85.33	0.27	9	68.67	1.21	5	91.33	0.51	9	90.67	0.15	1	90.00	0.19	9
ECSDT-PSO-3ell	90.00	2.80	7	90.67	0.69	7	72.66	2.25	7	86.66	0.73	7	91.33	0.19	7	91.33	0.33	7
ECSDT-PSO-4ell	96.66	1.53	9	96.00	0.28	9	92.00	1.40	9	95.33	0.37	9	96.66	0.14	9	93.33	0.19	9
ECSDT-PSO-5ell	94.00	1.80	13	91.33	0.87	13	86.00	1.46	13	95.33	0.37	13	91.33	0.19	13	94.00	0.24	13
ECSDT-PSO-6ell	93.33	1.87	17	94.66	0.47	17	92.66	0.73	17	90.66	0.57	17	94.00	0.17	17	94.66	0.17	17
ECSDT-PSO-7ell	94.00	1.80	21	90.00	0.88	21	87.33	1.45	21	94.00	0.40	21	93.33	0.17	21	93.33	0.19	21
ECSDT-GA-3ell	92.66	2.53	7	92.66	0.61	7	74.66	2.23	7	86.66	0.73	7	91.33	0.19	7	92.00	0.26	7
ECSDT-GA-4ell	96.66	1.53	9	96.66	0.21	9	91.33	1.41	9	96.00	0.33	9	96.66	0.14	9	94.00	0.24	9
ECSDT-GA-5ell	93.33	1.87	13	92.66	0.73	13	88.66	1.43	13	96.66	0.27	13	93.33	0.17	13	91.33	0.21	13
ECSDT-GA-6ell	96.66	1.53	17	96.00	0.28	17	92.66	0.73	17	94.66	0.27	17	94.66	0.16	17	94.66	0.17	17
ECSDT-GA-7ell	94.00	1.80	21	92.66	0.61	21	90.00	0.76	21	95.33	0.33	21	94.66	0.16	21	93.33	0.19	21

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	96.00	8.33	7	92.67	12.67	7	94.66	10.33	9	91.41	3.95	8.11
MetaCost+J48	94.67	11.33	7	95.33	8.33	7	94.00	11.33	7	91.85	3.84	6.44
C.S.C+NBTree	94.67	12.00	9	94.67	9.33	9	94.66	10.67	9	90.74	3.98	9
MetaCost+NBTree	92.67	16.33	9	92.67	12.67	9	93.33	12.00	7	87.63	5.11	7.67
ECSDT-PSO-3ell	89.33	23.33	7	90.00	18.33	7	90.66	16.67	7	88.07	7.26	7
ECSDT-PSO-4ell	96.00	9.00	9	96.66	6.00	9	95.33	9.67	9	95.33	3.17	9
ECSDT-PSO-5ell	94.00	13.33	13	94.66	9.00	13	96.66	7.00	13	93.03	3.81	13
ECSDT-PSO-6ell	96.66	7.67	17	96.66	6.00	17	94.00	12.33	17	94.14	3.33	17
ECSDT-PSO-7ell	96.00	9.00	21	95.33	8.00	21	96.00	8.00	21	93.26	3.32	21
ECSDT-GA-3ell	89.33	24.00	7	91.33	16.00	7	90.66	16.67	7	89.03	7.03	7
ECSDT-GA-4ell	95.33	10.67	9	96.66	6.00	9	96.00	8.00	9	95.48	3.17	9
ECSDT-GA-5ell	96.00	9.00	13	96.00	7.00	13	96.66	7.00	13	93.85	3.08	13
ECSDT-GA-6ell	96.00	9.00	17	96.66	6.00	17	96.66	7.00	17	95.40	2.79	17
ECSDT-GA-7ell	94.66	11.67	21	95.33	8.00	21	95.33	9.67	21	93.92	3.69	21

Table Apx-D-08: IRIS Accuracy + Cost based results

Algorithm	Hays Accuracy + Cost based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	84.38	0.16	23	74.38	0.37	23	81.88	1.03	23	84.38	0.16	23	74.38	0.37	23	81.88	1.03	23
MetaCost+J48	84.38	0.16	17	77.50	0.23	21	76.25	1.20	21	84.38	0.16	17	77.50	0.23	23	76.25	1.20	19
C.S.C+NBTree	54.38	1.13	13	57.50	0.80	13	64.38	1.10	1	54.38	1.13	13	57.50	0.80	13	64.38	1.10	5
MetaCost+NBTree	59.38	0.41	1	55.00	0.65	1	44.38	2.06	1	59.38	0.41	1	55.00	0.65	1	44.38	2.06	3
ECSDT-PSO-3ell	52.50	1.79	7	48.75	1.98	7	43.75	2.69	7	45.00	92.81	7	45.63	92.19	7	44.38	119.69	7
ECSDT-PSO-6ell	59.38	1.46	13	58.75	1.43	13	54.38	1.94	13	47.50	86.56	13	48.75	88.44	13	50.63	104.06	13
ECSDT-PSO-9ell	63.75	1.19	19	64.38	0.99	19	58.75	1.66	19	50.00	81.25	19	52.50	79.06	19	52.50	99.38	19
ECSDT-PSO-12ell	68.75	0.81	25	70.63	0.72	25	58.75	1.54	25	55.63	71.56	25	58.13	68.44	25	57.50	86.88	25
ECSDT-PSO-15ell	75.63	0.50	31	77.50	0.41	31	63.75	1.26	31	62.50	56.88	31	64.38	56.56	31	58.75	83.75	31
ECSDT-GA-3ell	52.50	1.79	7	50.00	1.86	7	45.00	2.56	7	43.75	94.69	7	46.88	89.69	7	43.75	121.25	7
ECSDT-GA-6ell	58.75	1.49	13	58.75	1.43	13	55.00	1.88	13	48.75	84.69	13	48.75	88.44	13	48.13	110.31	13
ECSDT-GA-9ell	63.75	1.19	19	66.25	0.93	19	58.75	1.66	19	51.88	78.13	19	52.50	79.06	19	54.38	94.69	19
ECSDT-GA-12ell	67.50	0.91	25	68.75	0.76	25	61.25	1.35	25	55.63	71.56	25	57.50	67.81	25	57.50	86.88	25
ECSDT-GA-15ell	75.63	0.50	31	77.50	0.41	31	65.63	1.19	31	62.50	50.88	31	66.25	51.50	31	60.00	77.63	31

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	83.13	28.44	23	83.13	34.69	23	83.13	37.19	23	81.67	16.98	23
MetaCost+J48	80.63	31.56	21	81.88	36.56	27	80.00	55.31	21	80.1	20.84	21.3
C.S.C+NBTree	59.38	60.94	1	62.50	68.75	13	64.38	71.25	13	60.42	33.99	9.67
MetaCost+NBTree	63.13	61.88	1	55.63	75.31	1	63.75	74.06	1	56.88	35.73	1.33
ECSDT-PSO-3ell	43.75	2.69	7	45.00	92.81	7	45.63	92.19	7	46.67	51.86	7
ECSDT-PSO-6ell	52.50	1.88	13	47.50	86.56	13	46.25	87.81	13	53.23	47.31	13
ECSDT-PSO-9ell	55.63	1.57	19	48.75	80.00	19	52.50	79.06	19	56.98	43.92	19
ECSDT-PSO-12ell	61.25	1.35	25	52.50	74.69	25	56.88	66.56	25	61.57	38.32	25
ECSDT-PSO-15ell	62.50	1.10	31	59.38	60.94	31	62.50	56.25	31	67.09	33.23	31
ECSDT-GA-3ell	43.75	2.69	7	45.00	92.81	7	44.38	94.69	7	46.98	51.97	7
ECSDT-GA-6ell	52.50	1.88	13	47.50	86.56	13	48.75	82.81	13	53.02	48.04	13
ECSDT-GA-9ell	56.25	1.51	19	47.50	85.31	19	51.88	76.56	19	57.92	42.61	19
ECSDT-GA-12ell	61.25	1.35	25	51.88	75.63	25	56.88	66.56	25	61.36	38.21	25
ECSDT-GA-15ell	62.50	1.10	31	58.13	63.44	31	61.88	50.88	31	67.92	32.85	31

Table Apx-D-09: Hays Accuracy + Cost based results

Algorithm	Seeds Accuracy + Cost based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	89.05	3.02	15	62.86	2.54	15	91.43	1.41	15	89.05	0.40	15	91.90	0.58	15	91.43	0.34	15
MetaCost+J48	86.19	0.74	11	77.62	2.75	13	81.43	1.60	15	86.67	0.26	17	86.19	0.89	13	86.19	0.41	15
C.S.C+NBTree	70.95	0.59	7	70.47	1.75	7	86.19	3.44	7	87.62	0.28	7	92.38	0.37	7	88.57	0.54	7
MetaCost+NBTree	67.14	0.46	7	61.90	2.48	7	70.95	3.38	15	87.62	0.22	1	86.19	0.67	3	88.10	0.40	5
ECSDT-PSO-3ell	62.86	4.57	7	56.19	3.74	7	58.57	4.23	7	62.86	1.10	7	62.86	1.51	7	63.81	1.43	7
ECSDT-PSO-6ell	75.71	3.24	13	69.52	2.83	13	69.52	3.05	13	72.86	0.81	13	68.57	1.14	13	68.57	1.10	13
ECSDT-PSO-9ell	79.52	1.96	19	76.67	2.08	19	78.57	2.27	19	77.62	0.59	19	75.71	1.01	19	76.67	0.81	19
ECSDT-PSO-12ell	86.66	0.90	25	80.48	1.57	25	86.66	1.50	25	84.76	0.45	25	84.76	0.52	25	85.71	0.36	25
ECSDT-PSO-15ell	84.29	0.76	31	82.38	1.76	31	89.52	1.39	31	89.52	0.24	31	92.85	0.38	31	90.95	0.30	31
ECSDT-GA-3ell	62.86	4.57	7	58.10	3.55	7	58.60	4.23	7	62.86	1.10	7	62.86	1.51	7	63.81	1.43	7
ECSDT-GA-6ell	73.81	3.26	13	69.52	2.83	13	70.00	3.00	13	69.50	0.85	13	66.19	1.20	13	68.57	1.10	13
ECSDT-GA-9ell	80.95	1.90	19	78.57	1.50	19	75.71	2.39	19	78.57	0.61	19	77.62	0.96	19	75.71	0.84	19
ECSDT-GA-12ell	84.76	0.84	25	80.48	1.57	25	86.70	1.50	25	85.71	0.42	25	82.86	0.56	25	87.14	0.32	25
ECSDT-GA-15ell	87.61	0.65	31	84.29	1.66	31	88.57	1.44	31	89.52	0.24	31	90.47	0.43	31	90.95	0.30	31

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	91.90	15.48	15	91.90	17.86	15	91.90	15.24	15	87.44	6.32	15
MetaCost+J48	87.62	24.76	15	90.00	20.95	15	88.10	21.19	19	85.98	8.17	14
C.S.C+NBTree	90.95	20.71	7	91.43	18.10	7	91.43	18.10	7	85.18	7.10	8
MetaCost+NBTree	87.14	25.00	7	89.05	23.33	7	87.14	24.29	7	81.06	8.91	7
ECSDT-PSO-3ell	56.67	77.14	7	53.81	80.95	7	56.19	74.29	7	59.31	27.66	7
ECSDT-PSO-6ell	64.29	63.33	13	60.00	70.48	13	62.86	60.48	13	67.99	22.94	13
ECSDT-PSO-9ell	69.52	52.86	19	65.71	59.52	19	71.90	45.00	19	74.65	18.46	19
ECSDT-PSO-12ell	76.19	39.52	25	78.57	36.90	25	80.48	32.14	25	82.70	12.65	25
ECSDT-PSO-15ell	82.86	27.62	31	85.71	24.29	31	87.61	20.48	31	87.30	8.58	31
ECSDT-GA-3ell	55.24	79.29	7	54.76	79.52	7	56.19	74.29	7	59.48	27.72	7
ECSDT-GA-6ell	64.29	63.33	13	60.48	69.29	13	62.86	59.52	13	67.25	22.71	13
ECSDT-GA-9ell	70.00	51.90	19	65.24	60.24	19	72.86	44.52	19	75.03	18.32	19
ECSDT-GA-12ell	74.76	41.67	25	80.48	34.05	25	80.48	31.19	25	82.60	12.46	25
ECSDT-GA-15ell	80.95	30.48	31	85.71	24.29	31	88.09	19.76	31	87.35	8.81	31

Table Apx-D-010: Seeds Accuracy + Cost based results

Algorithm	Tae Accuracy + Cost based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	54.30	1.85	67	54.96	1.82	67	50.99	2.10	67	57.61	83.44	67	54.96	1.82	67	56.95	82.78	67
MetaCost+J48	46.35	2.55	33	51.65	1.56	35	45.03	2.03	45	59.60	78.81	33	51.65	1.56	37	54.30	85.43	35
C.S.C+NBTree	40.39	1.91	7	47.01	1.49	7	46.35	1.83	7	56.29	87.42	7	47.01	1.49	7	56.29	86.09	7
MetaCost+NBTree	36.42	2.18	11	45.03	1.66	3	40.39	1.97	11	50.33	97.02	11	45.03	1.66	7	53.64	85.76	3
ECSDT-PSO-3ell	29.80	4.04	7	32.45	3.05	7	29.80	3.44	7	34.40	124.83	7	32.50	2.81	7	34.40	130.79	7
ECSDT-PSO-6ell	33.77	3.25	13	36.42	2.72	13	33.11	3.15	13	38.40	116.56	13	36.40	2.58	13	40.40	115.89	13
ECSDT-PSO-9ell	37.75	2.64	19	44.37	2.12	19	37.75	2.49	19	45.70	99.01	19	44.40	1.89	19	47.70	97.68	19
ECSDT-PSO-12ell	43.71	2.18	25	46.36	1.85	25	43.71	2.10	25	55.60	73.18	25	56.30	1.49	25	58.90	60.54	25
ECSDT-PSO-15ell	48.30	1.91	31	52.98	1.36	31	48.34	1.54	31	56.30	70.20	31	56.30	1.16	31	62.25	59.93	31
ECSDT-GA-3ell	30.46	3.97	7	32.45	3.05	7	31.13	3.34	7	34.40	124.83	7	34.40	2.78	7	34.40	120.79	7
ECSDT-GA-6ell	33.77	3.25	13	35.76	2.75	13	33.77	3.15	13	37.10	119.21	13	37.10	2.58	13	39.74	116.89	13
ECSDT-GA-9ell	36.42	2.68	19	42.38	2.25	19	38.41	2.42	19	46.40	97.35	19	43.70	1.89	19	47.70	97.68	19
ECSDT-GA-12ell	45.03	2.11	25	48.34	1.72	25	41.72	2.20	25	57.00	71.19	25	55.00	1.42	25	57.60	71.52	25
ECSDT-GA-15ell	48.30	1.91	31	53.64	1.35	31	47.68	1.57	31	58.28	67.22	31	56.30	1.16	31	62.25	59.93	31

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	57.61	83.44	67	56.95	82.78	67	60.92	79.80	67	55.96	41.97	67
MetaCost+J48	59.60	78.81	51	54.30	85.43	45	55.62	90.07	51	52.09	43.41	42
C.S.C+NBTree	56.29	87.42	7	56.29	86.09	7	58.94	78.48	7	50.88	42.87	7
MetaCost+NBTree	50.33	97.02	11	53.64	85.76	11	54.96	87.09	7	46.8	45.94	8.33
ECSDT-PSO-3ell	30.46	127.48	7	30.50	107.48	7	32.50	123.84	7	32.23	66.104	7
ECSDT-PSO-6ell	33.77	114.57	13	34.40	104.57	13	36.40	114.24	13	36.42	60.348	13
ECSDT-PSO-9ell	42.38	97.35	19	43.00	77.35	19	44.40	97.68	19	42.95	51.098	19
ECSDT-PSO-12ell	43.71	76.16	25	53.60	56.16	25	55.60	68.18	25	50.76	37.832	25
ECSDT-PSO-15ell	50.33	70.86	31	56.30	50.86	31	61.60	62.25	31	54.08	35.514	31
ECSDT-GA-3ell	31.13	114.50	7	32.50	104.50	7	33.80	116.85	7	32.87	65.579	7
ECSDT-GA-6ell	33.77	112.25	13	35.80	96.25	13	36.40	104.24	13	36.21	60.795	13
ECSDT-GA-9ell	42.38	97.35	19	43.00	77.35	19	44.40	97.68	19	42.5	51.061	19
ECSDT-GA-12ell	45.70	76.16	25	53.60	56.16	25	54.30	75.17	25	50.78	38.035	25
ECSDT-GA-15ell	50.33	70.86	31	56.30	49.86	31	61.60	62.25	31	32.23	35.022	31

Table Apx-D-11: Tae Accuracy + Cost based results

Algorithm	Thyroid Accuracy + Cost based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	93.02	0.27	17	91.16	0.47	17	92.09	0.40	17	92.09	15.12	17	92.09	16.74	17	92.09	15.58	17
MetaCost+J48	89.30	0.24	11	88.83	0.47	11	91.16	0.43	11	91.16	16.51	17	91.16	18.14	11	90.70	18.84	13
C.S.C+NBTree	90.69	0.44	7	92.56	0.25	7	93.02	0.33	7	93.02	14.88	7	93.48	12.33	7	93.02	11.16	7
MetaCost+NBTree	92.09	0.33	1	89.77	0.34	1	92.09	0.39	1	90.23	20.23	9	90.70	17.44	1	90.23	20.00	9
ECSDT-PSO-3ell	80.47	0.80	7	84.65	0.88	7	84.65	0.81	7	84.19	30.70	7	84.19	29.30	7	83.72	31.40	7
ECSDT-PSO-6ell	88.37	0.56	13	87.91	0.65	13	88.84	0.63	13	87.44	24.42	13	88.84	20.93	13	86.96	26.51	13
ECSDT-PSO-9ell	92.56	0.33	19	91.62	0.42	19	93.95	0.44	19	94.41	13.02	19	92.55	15.58	19	93.95	15.12	19
ECSDT-PSO-12ell	90.23	0.31	25	91.62	0.29	25	92.09	0.37	25	92.55	14.42	25	94.40	10.70	25	95.81	10.47	25
ECSDT-PSO-15ell	95.81	0.21	31	94.88	0.22	31	96.74	0.28	31	95.81	10.47	31	94.40	10.70	31	95.35	11.63	31
ECSDT-GA-3ell	84.19	0.68	7	84.65	0.88	7	84.65	0.81	7	84.65	29.77	7	84.65	28.60	7	84.65	30.00	7
ECSDT-GA-6ell	90.69	0.48	13	88.37	0.63	13	88.84	0.63	13	86.51	25.81	13	86.51	23.49	13	88.84	23.72	13
ECSDT-GA-9ell	92.56	0.33	19	93.02	0.39	19	94.88	0.39	19	95.34	11.63	19	90.69	15.81	19	94.88	13.72	19
ECSDT-GA-12ell	93.48	0.24	25	90.69	0.30	25	94.88	0.34	25	96.27	9.30	25	93.95	11.63	25	94.88	11.86	25
ECSDT-GA-15ell	96.74	0.20	31	95.34	0.21	31	96.74	0.28	31	96.74	8.14	31	95.34	9.30	31	96.28	10.23	31

Algorithm	Ratio - 7			Ratio - 8			Ratio - 9			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	91.16	0.47	17	92.09	0.40	17	92.09	15.12	17	92.09	8.10	17
MetaCost+J48	88.83	0.47	17	91.16	0.43	13	91.16	16.51	17	90.39	9.11	15
C.S.C+NBTree	92.56	0.25	7	93.02	0.33	7	93.02	14.88	7	92.63	6.57	7
MetaCost+NBTree	89.77	0.34	9	92.09	0.39	9	90.23	20.23	11	90.85	9.79	8
ECSDT-PSO-3ell	80.47	0.80	7	84.65	0.88	7	80.47	45.70	7	83.65	15.65	7
ECSDT-PSO-6ell	88.37	0.56	13	87.91	0.65	13	88.37	34.42	13	88.06	12.28	13
ECSDT-PSO-9ell	92.56	0.33	19	91.62	0.42	19	92.56	23.02	19	93.17	7.484	19
ECSDT-PSO-12ell	90.23	0.31	25	91.62	0.29	25	90.23	24.42	25	92.78	6.092	25
ECSDT-PSO-15ell	95.81	0.21	31	94.88	0.22	31	95.81	18.47	31	95.5	5.584	31
ECSDT-GA-3ell	84.19	0.68	7	84.65	0.88	7	84.19	44.77	7	84.57	15.12	7
ECSDT-GA-6ell	90.69	0.48	13	88.37	0.63	13	90.69	35.81	13	88.29	12.46	13
ECSDT-GA-9ell	92.56	0.33	19	93.02	0.39	19	92.56	20.63	19	93.56	7.043	19
ECSDT-GA-12ell	93.48	0.24	25	90.69	0.30	25	93.48	15.30	25	94.03	5.612	25
ECSDT-GA-15ell	96.74	0.20	31	95.34	0.21	31	96.74	12.14	31	96.2	4.729	31

Table Apx-D-012: Thyroid Accuracy + Cost based results

Algorithm	Glass Accuracy + Cost based Results																				
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6			Average		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Acc	Acc	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	96.73	12.67	11	96.73	10.29	11	96.73	8.27	11	96.73	3.33	11	96.73	10.14	11	96.73	9.64	11	96.73	9.06	11
MetaCost+J48	95.32	4.45	11	95.79	6.68	11	97.66	7.34	11	93.93	4.64	11	93.46	6.52	11	96.26	2.72	11	95.40	5.39	11
C.S.C+NBTree	50.47	22.75	7	64.02	11.58	7	82.71	12.50	7	88.32	4.96	7	81.31	5.98	7	55.61	14.74	7	70.40	12.09	7
MetaCost+NBTree	34.58	13.41	11	55.14	11.29	9	80.37	24.71	5	85.51	7.44	7	80.84	10.74	9	50.47	14.39	13	64.49	13.66	9
ECSDT-PSO-6ell	52.34	12.83	13	50.47	16.68	13	54.67	87.37	13	52.34	61.00	13	55.14	23.36	13	53.27	16.58	13	53.04	36.30	13
ECSDT-PSO-8ell	55.61	10.50	17	53.74	13.03	17	57.94	75.22	17	55.61	53.99	17	57.94	21.02	17	56.54	15.70	17	56.23	31.57	17
ECSDT-PSO-10ell	60.75	9.12	21	58.41	11.39	21	63.55	51.34	21	62.15	38.10	21	64.95	16.82	21	62.62	13.22	21	62.07	23.33	21
ECSDT-PSO-12ell	68.22	2.52	25	67.76	6.82	25	69.63	32.93	25	67.29	29.22	25	62.62	17.99	25	66.82	11.12	25	67.06	16.77	25
ECSDT-PSO-14ell	74.77	2.29	29	75.70	3.62	29	77.57	9.63	29	75.70	18.00	29	73.83	7.94	29	71.96	9.25	29	74.92	8.45	29
ECSDT-GA-6ell	52.34	12.83	13	51.40	15.75	13	54.70	87.37	13	52.34	61.00	13	55.14	23.36	13	54.21	16.30	13	53.36	36.10	13
ECSDT-GA-8ell	56.54	10.21	17	53.74	10.69	17	55.61	82.70	17	55.61	53.99	17	58.41	20.56	17	57.48	15.23	17	56.23	32.23	17
ECSDT-GA-10ell	60.75	9.12	21	56.54	11.41	21	65.42	43.86	21	63.08	37.40	21	62.62	17.99	21	61.21	13.55	21	61.60	22.22	21
ECSDT-GA-12ell	70.09	2.42	25	68.69	6.73	25	71.50	24.52	25	66.36	30.15	25	67.76	14.01	25	68.22	10.79	25	68.77	14.77	25
ECSDT-GA-14ell	74.77	2.29	29	74.77	3.67	29	78.50	9.16	29	74.77	18.94	29	74.30	7.71	29	73.36	8.55	29	75.08	8.38	29

Table Apx-D-013: Glass Accuracy + Cost based results

Algorithm	Ecoli Accuracy + Cost based Results																	
	Ratio - 1			Ratio - 2			Ratio - 3			Ratio - 4			Ratio – 5			Ratio - 6		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	31.85	31.19	41	66.07	41.95	41	78.27	57.06	41	77.67	32.95	41	75.30	22.48	41	76.19	50.99	41
MetaCost+J48	32.14	19.55	21	65.18	40.94	31	78.87	33.52	17	78.27	20.02	29	74.40	17.81	25	68.45	42.50	41
C.S.C+NBTree	26.49	20.07	13	63.99	31.42	13	74.70	44.68	13	72.62	28.77	13	66.67	24.49	13	55.06	37.28	13
MetaCost+NBTree	25.00	19.19	9	61.01	34.99	23	76.79	37.24	13	72.62	28.80	13	65.18	21.83	5	58.63	35.16	9
ECSDT-PSO-6ell	23.21	28.32	13	55.65	40.47	13	66.67	52.18	13	64.88	31.11	13	57.74	24.98	13	50.60	48.76	13
ECSDT-PSO-8ell	25.89	23.50	17	59.23	35.38	17	70.24	45.93	17	68.15	27.40	17	62.20	22.15	17	55.95	46.06	17
ECSDT-PSO-10ell	28.57	22.27	21	62.80	31.79	21	73.81	40.10	21	74.70	22.71	21	67.56	18.59	21	60.42	43.66	21
ECSDT-PSO-12ell	26.79	19.31	25	66.67	29.52	25	75.89	32.74	25	77.38	19.74	25	73.81	15.41	25	66.67	36.93	25
ECSDT-PSO-14ell	31.25	17.19	29	64.88	27.46	29	81.24	38.76	29	80.95	18.98	29	77.68	15.15	29	72.62	30.25	29
ECSDT-GA-6ell	24.70	27.35	13	57.44	40.01	13	65.48	53.07	13	66.67	30.97	13	59.63	24.53	13	51.49	48.32	13
ECSDT-GA-8ell	26.49	22.91	17	59.52	34.64	17	70.83	45.04	17	68.45	27.57	17	63.99	21.34	17	57.44	45.96	17
ECSDT-GA-10ell	26.79	22.47	21	64.58	30.72	21	75.30	38.90	21	74.40	23.01	21	66.37	18.95	21	63.69	37.92	21
ECSDT-GA-12ell	30.06	17.68	25	67.26	30.11	25	74.40	32.83	25	78.87	19.51	25	74.70	15.34	25	67.26	36.49	25
ECSDT-GA-14ell	33.63	15.73	29	66.37	27.74	29	80.36	37.63	29	81.85	18.89	29	76.79	15.24	29	74.40	30.21	29

Algorithm	Ratio - 7			Ratio - 8			Averages		
	Acc	Cost	Size	Acc	Cost	Size	Acc	Cost	Size
C.S.C+J48	69.35	40.40	41	57.44	22.56	41	66.52	37.45	41
MetaCost+J48	62.50	31.65	43	60.71	16.78	45	65.07	27.85	32
C.S.C+NBTree	47.32	24.06	13	38.39	19.14	13	55.65	28.74	13
MetaCost+NBTree	50.60	27.60	11	14.88	32.12	19	53.05	25.61	13
ECSDT-PSO-6ell	45.83	37.63	13	41.67	21.90	13	50.78	35.67	13
ECSDT-PSO-8ell	51.19	34.18	17	46.73	20.81	17	54.95	31.93	17
ECSDT-PSO-10ell	56.85	29.34	21	53.27	17.36	21	59.75	28.23	21
ECSDT-PSO-12ell	61.90	25.61	25	59.52	15.43	25	63.58	24.34	25
ECSDT-PSO-14ell	66.37	21.26	29	64.88	17.14	29	67.48	23.27	29
ECSDT-GA-6ell	44.94	37.84	13	40.18	22.01	13	51.32	35.51	13
ECSDT-GA-8ell	51.19	34.18	17	44.94	20.95	17	55.36	31.57	17
ECSDT-GA-10ell	58.33	29.23	21	55.65	17.26	21	60.64	27.31	21
ECSDT-GA-12ell	63.69	25.49	25	58.63	15.59	25	64.36	24.13	25
ECSDT-GA-14ell	65.77	21.31	29	63.99	17.18	29	67.90	22.99	29

Table Apx-D-014: Ecoli Accuracy + Cost based results