COST SENSITIVE LEARNING ALGORITHMS

GÜLAY KÖSE

IŞIK UNIVERSITY
2008

COST SENSITIVE LEARNING ALGORITHMS

GÜLAY KÖSE

B.S., Computer Engineering, Işık University, 2005

Submitted to the Graduate School of Science and Engineering
in partial fulfillment of the requirements for the degree of
Master of Science
in
Computer Engineering

IŞIK UNIVERSITY
2008

IŞIK UNIVERSITY

GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

COST SENSITIVE LEARNING ALGORITHMS

GÜLAY KÖSE

APPROVED BY:

Assist. Prof. Olcay Taner Yıldız (Işık University) _____
(Thesis Supervisor)

Assist. Prof. Taner Eskil (Işık University) _____

Assist. Prof. Uluğ Bayazıt (Istanbul Teknik University) _____

APPROVAL DATE :

COST SENSITIVE LEARNING ALGORITHMS

Abstract

This thesis studies the cost sensitive learning algorithms that calculate the class learning algorithms errors and costs. Data mining is the automated extraction of hidden predictive information from databases that can be applied to predict and diagnose many illnesses. Specifically, accurate classification of illnesses is a very important issue for the treatment of illnesses. The goal of classification is to build a set of models that can correctly predict the class of the different objects. Some algorithms produce better results than others. It is necessary to analyze systematically the performance of classifiers using a variety of datasets.

In this thesis, many features were explored and 10 datasets were classified by using 5 classification algorithms. Logistic Discrimination Algorithm (LD), K-Nearest Neighbor Algorithm (KNN), Multilayer Perceptron Algorithm (MLP) and Nearest Mean algorithm and Decision Tree (C4.5) algorithms have been used for classification. These methods are applied to large and small datasets and then a large number of experiment results were obtained. The results show that there is no single algorithm that performs well in all domains. K-Nearest Neighbor Algorithm (KNN), Multilayer Perceptron (MLP), and Decision Tree (C4.5) algorithms had three steps: train, validate and test. Nearest Mean and Logistic Discrimination algorithms only had train and test steps. In these algorithms, each set had different percentage of data and had equal percentage of classes. The algorithms errors and costs were calculated for each dataset. The error rate is calculated based on the misclassified classes. The algorithms' classification performance is quantified by their error rate. In many applications, not all misclassifications have the same value. Within this thesis, multi-class weighting cost methods are also discussed. Cost models are used for composing cost matrix and experiments. Class Frequency, MaxCost and AvgCost methods were used to calculate costs.

MALİYET DUYARLILIK OĞRENME ALGORITMALARI

Özet

Bu tez, maliyet duyarlılık öğrenme algoritmalarını içermektedir. Algoritmalar kullanılarak sınıf öğrenme hataları ve yanlış tahmin edilen sınıfların maliyetleri hesaplanmaktadır. Veri madenciliğinde elimizde hangi sınıfta olduğunu bildiğimiz verileri kullanarak, hangi sınıfta olduğunu bilmediğimiz verinin hangi sınıftan olduğunu algoritmaları kullanarak tahmin edebiliriz. Sınıflandırmadaki amaç farklı sınıflardan oluşan bilgileri doğru sınıflandırmak için doğru modeller kurmak. Verinin dağılımına gore bir model bulunur. Bulunan model, başarımı belirlendikten sonra niteliğin gelecekteki ya da bilinmeyen değerini tahmin etmek için kullanılır. Bazı veri grupları için iyi sınıflandırma sağlayan algoritma başka veri grupları için iyi sınıf tahmin edemeyebilir. Hangi tip algoritma hangi tip verilerde sınıf tahmin etme hatası düşük onları elde ettik.

Tezde değişik sayıda özellikleri, sınıfları ve veri grupları kullanıldı. Bu veri grupları 5 değişik algoritma kullanılarak eğitildi, doğrulandı ve test edildi. Sınıflandırma için kullanılan algoritmalar Logistic Discrimination, K-Nearest Neighbor, Multilayer Perceptron, C4.5 Decision Tree ve Nearest Mean algoritması. Bu methodlar çok büyük, orta derecede büyük sayıda ve küçük sayıda veri gruplarına uygulandı. Deneylerden pekçok sonuçlar elde edildi. Grafikler çizildi. Bu sonuçlar gösteriyorki her durumda en iyi sonucu veren algoritma yok. Degişik algoritmalar değişik veri gruplarının sınıflarını iyi tahmin edebiliyor. Algoritmalarda değişik yüzdelerde veri kullanıldı ve yüzdesine göre eşit sayıda sınıflar kullanıldı. Herbir veri grupları için hatalar ve maliyetler hesaplandı. Algoritmaların sınıflandırılma performansı hata oranlarına göre değerlendirildi. Pekçok uygulamada yanlış sınıflandırma aynı değerde değil. Bunun için çok sınıflı ağırlık maliyet algoritmaları kullanıldı. Maliyet modelleri maliyet matrikslerini oluşturmak için kullanıldı. Maliyet hesaplamak icin kullanılan maliyet algoritmaları Class Frequency, MaxCost ve AvgCost.

Acknowledgements

I would like to thank my thesis advisor Assistant Prof. Dr. Olcay Taner Yıldız for his supervision, advice and guidance throughout the work.

I am very grateful to my parents Hasan Hüseyin and Nahide Köse, my sister Esmeray and my uncle Murat Köse who always loved me, encouraged and supported my education.

Table of Contents

List of Figures

List of Tables

List of Symbols

class_column	: Class Column of All Data Records
ClassCount	: Number of Classes of All Data Records
column	: Number of Columns of All Data Records
HiddenUnit	: Number of Hidden units :
m	: Mean
r	: Required output
row	: Number of Rows of All Data Records
s	: Variance
TestPercentage	: Percentage of Test Data of All Data Records
TrainPercentage	: Percentage of Train Data of All Data Records
ValidatePercentage	: Percentage of Validate Data of All Data Records
w	: weights
X	: Matrix of Train data(inputs)
XTest	: Matrix of Test data (inputs)
XValidate	: Matrix of Validate data(inputs)
y	: output
z	: Bias of the hidden layer

Chapter 1
Introduction

Machine learning is concerned with the design and development of algorithms and techniques that allow computers to learn. Machine learning that addresses non-uniform cost is known as cost sensitive learning. All misclassifications are not the same value and thus, the purpose is to decrease the cost. Machine learning methods extract rules and patterns out of large data sets. Application of machine learning methods to large databases is called data mining. Data mining is one of the most actively researched areas in information science with important real world applications. Data Mining is the process of automatically searching large volumes of data for patterns using tools such as classification, association rule mining, clustering, etc.

There are two major types of learning: supervised learning and unsupervised learning.

Supervised learning is a machine learning technique for creating a function from training data. The training data consist of pairs of input objects (typically vectors), and desired outputs. Output can predict a class label of the input object. Classification is the process of finding a set of models which describe and distinguish data classes. It uses a model to predict the class of objects whose class is not known.

Unsupervised learning has inputs and the aim is to find regularities in the input. Class labels of the training samples are not known, and the number or set of classes to be learned may not be known in advance. One form of unsupervised learning is clustering.

One of the examples of data mining tools that are used in Turkey is Clementine program, which is especially used in the banks. Clementine helps organizations to improve customer and citizen relationships through an in-depth understanding of data. Clementine offers many modeling techniques, such as

prediction, classification, segmentation, and association detection algorithms. There are many algorithm options in this program such as Nearest Mean Algorithm, QUEST Algorithms, Kohonen Algorithms, Apriori Algorithms, Logistic Regression Algorithm, Carma Algorithms, RBF and C5.0 Algorithms.

This thesis is about cost sensitive learning algorithms. Logistic Discrimination, KNN (K-Nearest Neighbor Algorithm), MLP (Multilayer Perceptron), C4.5 (Decision Tree) and Nearest Mean algorithms were written. These algorithms were written with C program and can be used on different kind of data. All algorithms need normalization without C4.5 (Decision Tree) algorithm. Data was divided to three parts such as Train Set, Test Set and Validate Set. In the algorithms, each set has different percentage of data and each has equal percentage of classes based on their percentage and shuffled rows.

K-Nearest Neighbor Algorithm (KNN), Multilayer Perceptron (MLP), and Decision Tree (C4.5) Algorithms have three steps train, validate and test. Nearest Mean and Logistic Discrimination algorithms only have train and test steps. At the end error and cost were calculated.

This thesis is composed of five parts. In the first chapter, introduction is given. The second chapter describes the classification algorithms' structure and pseudo codes. The third chapter discusses cost sensitive learning methods and cost models. The fourth chapter reports the experiment results and includes graphic illustrations of cost and errors. Results are compared with different algorithms on several datasets.

Chapter 2
Classification Algorithms

The purpose of the classification is to compose models that can predict the class of datasets. The classification performance of the algorithms is quantified by their error rate. Training and testing is needed for good performance [1].

2.1 K-Nearest Neighbor (KNN) Algorithm

KNN is good for classification decision and class is predicted according to neighbor similar classes. K-Nearest Neighbor algorithm is famous in some areas such as recognition of handwriting, satellite image and EKG patterns [1]. Distance is calculated between two instances. Euclidean distance formula is used for distance calculation; Euclidean Distance is the most frequently used distance measure. Euclidean distance is the root of square differences between coordinates of a pair of objects:

$$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + ... + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2} \quad (2.1)$$

Euclidean distance function handles every dimension equally, but data must be normalized. Normalization reduces the misclassification rate. If the normalization is not applied, distance will be very different. For example, if a feature has values between 0.5 – 4.5 and another feature has values between 1000-5000, this situation causes bad performance because all big valued features control distance comparisons [1].

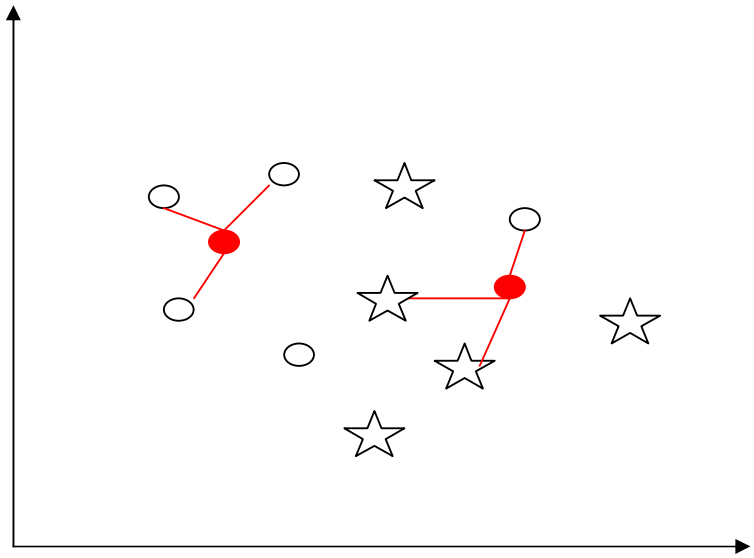
KNN has three steps; train, validate and test. Firstly, k number is decided and *k* number is taken from 1 to 10 in KNN algorithm. In this classification, k nearest neighbors of each training data to validate data is computed firstly.

```
1 For i= 0,…,ValidateRow
2     For  j=0,…,N
3        Index_ij=j;
4         Distance_ij ← Euclidean(XValidate_i, X^j)
5   quickSort(toplam,index,TrainRow)
6 For k= 1,…,10
7     HighestVotedClass ← VotingKClosestDistance(distance^j):
8     Errork← CalculateError(HighestVotedClass, XValidate_i);
9     BestK←SmallestError(Error^k)
10    k ← BestK
11 For i= 0,…,TestRow
12   For j=0,…,N
13     Distance_ij ← Euclidean(XTest_i, X^j)
14     HighestVotedClass ← VotingKClosestDistance(distance^j):
15     Error← CalculateError(HighestVotedClass, XTest_i)
```

Figure 2.1 KNN Algorithm

Sort the distance in increasing order and determine k nearest neighbors according to minimum distance. At *k* minimum distance each class numbers are counted. The class with the maximum number of voters among the *k* neighbor is chosen. The class type of validate data and class with the highest vote at Training Data were compared. If they are the same class, it is correctly classified; if they are not equal it is misclassified.

This *BestK* was used for testing. The distance between each Test data to all the training data by using *BestK* was calculated and was calculated error of K-NN algorithm as shown above. The same process was done. By applying ten fold, 10 different errors were obtained from the KNN algorithm and the average error was also calculated after ten fold. Average confusion matrix was obtained. Diagonal Confusion Matrix shows the numbers of classes that were correctly classified. Other elements of matrix show the number of classes that is misclassified.



k=3

Figure 2.2 KNN Example

Figure 2.2 shows an example of 2 class KNN algorithm. One of the classes is star shape and the other is oval shape. Red ovals are test data and the others are train data. Red and black ovals are in the same class. If the number k is 3, 3, the closest data to each test class are found as seen above. At the left side red oval has 3 closest classes and the class is the same with it, so it is correctly classified. At the right side red class has 2 star shapes and 1 oval shape closest to it, 2 is greater than 1 so it is misclassified. It's not in the same class with maximum number closest class.

The advantage of KNN is more predictable if the training data is in large quantity. The disadvantage is the need to decide the value of parameter k (number of nearest neighbors). Computation cost is quite high because of distance calculations and large memory is required. When data is prepared, features that are not important can be removed or each feature can be weighted differently for decreasing computation time [1].

2.2 Nearest Mean Algorithm

Nearest mean is an algorithm to classify instances into K number of group. In this algorithm, Train and Test data were used. Class means were calculated. Each class type data column average of Train data was calculated and obtained the class means.

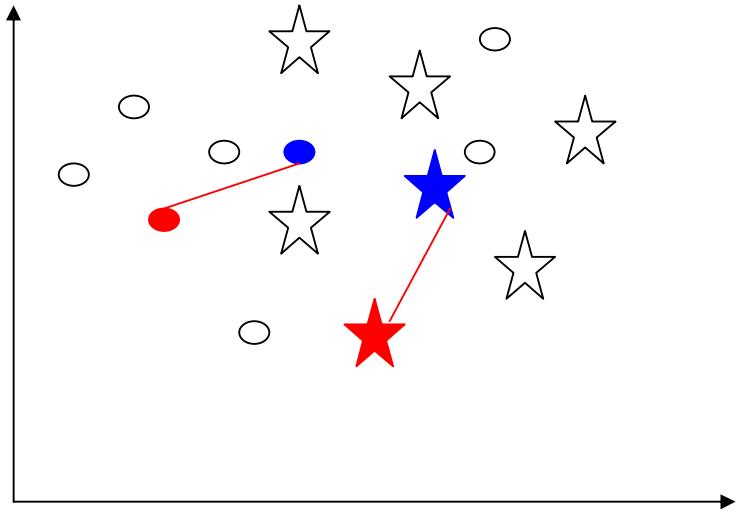


Figure 2.3 Nearest Mean Example

Figure 2.3 shows an example of 2 classes Nearest Mean Example algorithm. One of the classes is star shape and the other is oval shape. Red shapes are test data, blue data are centers of each train class's centers and the others are train data. For each test data the closest center is found. At the left side red oval test data is closest to the blue oval center and they are in the same class, so it is correctly classified. At the right side, red star test data is closest to blue star and they are in the same class, so it is correctly classified. If they weren't in the same class, they would be misclassified.

```
1 X_Avg ← CalculateEachClassColumnAverage(X^i)
2 For i=0,…,TestRow
3 ClassOfMinimumDistanceEachClassColumnAverage ←
Euclidean(XTest_i, X_Avg)
4 If(ClassOfMinimumDistanceEachClassColumnAverage=
XTest_i)
5 TruePredicted
6 Else
7 FalsePredicted
```

Figure 2.4 Nearest Mean Algorithm

The Euclidean distance of each Test data to the class means is calculated. Nearest Mean algorithms only had train and test steps. Each class means distance to

test data is calculated. If the minimum distance is the same class with Test data, it is correctly classified. By applying ten folds, 10 different errors were obtained from Nearest Mean algorithm and got average error after ten fold. Average confusion matrix was obtained.

2.3 Logistic Discrimination

Logistic discrimination is a well established method for classifying observations to two or more groups. The logistic discrimination technique can be regarded as a partially parametric approach to pattern recognition. This method is general and robust because it doesn't make assumptions on the underlying distribution of data.

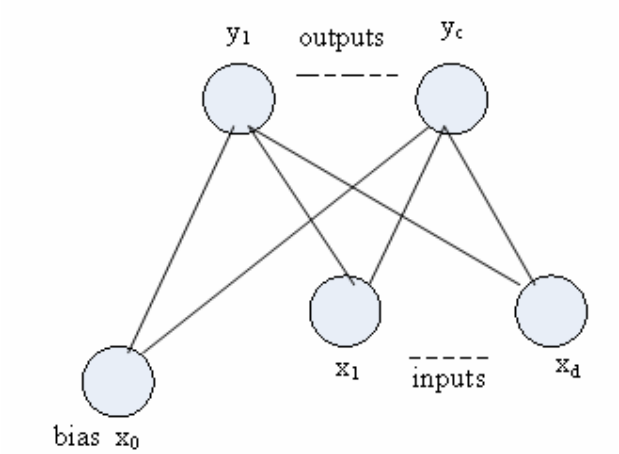


Figure 2.5 The structure of a Logistic Discrimination

LD algorithm only has train and test steps.

```
For i=1,…K, For j=0,…,d, w_ij ← rand(-0.01,0.01)
Repeat
    For(i=1,…K, For j=0,…d, Δw_ij←0
    For t =1,…,N
      For i=1,…,K
        o_i ←0
        For j=0,…,d
            o_i← o_i +w_ij x^t_j
      For i=1,…,K
        y_i ← exp(o_i)/Σ_k exp(o_k)
      For i=1,…, K
         For j=0,…,d
            Δw_ij← Δw_ij + (r^t_i - yi)x^t_j)
    For i=1,… ,K
      For j=0,…,d
         Δw_ij← Δw_ij + η Δw_ij
Until Convergence
```

Figure 2.6 LD Algorithm ([4])

$$\Delta w_{j0} = \eta \sum_t (r_j^t - y_j^t) \tag{2.2}$$

w_{ij} was initialized with random values between -0.01 and 0.01. r^t_i contains the actual output. Weights are updated at Train section. The weights that learned at train section used in the test section.

Update of weights is done in each epoch. Epoch number and learning factor changes from dataset to dataset. In our experiments, we found the best epoch number and learning factor for each dataset.

Magnitude of the update depends on the learning factor η. If it is too large, updates depend too much on recent instances. If this factor is small, many updates may be needed for convergence. Learning factor is generally taken between 0.0 and 1.0, mostly less than or equal to 0.2

Initially all the weights are close to 0 and thus have little effect. As training continues, the most important weights start moving away from 0 and are utilized.

By applying ten fold, 10 different errors were obtained from LD Algorithm and got average error after ten fold. Average confusion matrix was obtained.

Logistic discrimination is better for small training sets. Logistic Discrimination is popular with data analysts, machine learning researchers, statisticians, and with econometricians [1].

2.4 Multilayer Perceptron (MLP)

The multi-layer perceptron network composed of a network or nodes arranged in layers. It needs three or more layers of processing nodes: an input layer which accepts the input variables. Data from an input is presented at the input layer and the network nodes do calculations in the successive layers until an output value is computed at each of the output nodes. Output shows the suitable class for the input data. We expect to have a high output value on the correct class node. The training set is continued iteratively to the network until a stable set of weights is obtained and the error function is decreased to an acceptable level. The MLP is the supervised neural networks.

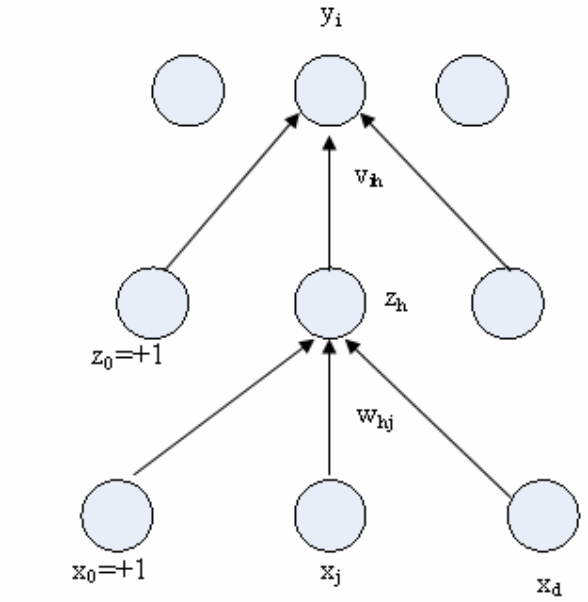


Figure 2.7 The structure of a multilayer perceptron.

As you see at Figure 2.7 x_j, j= 0,…,d are the inputs, z_h , h=1,…,H are the hidden units where H is the dimensionality of this hidden space. z_0 is the bias of the hidden layer. y_i, i = 1,…, K are the output units. w_{hj} are weights in the first layer, and v_{ih} are the weights in the second layer.

$$\Delta v_{ih} = \eta \sum_t (r_i^t - y_i^t) z_h^t \tag{2.3}$$

$$\Delta w_{hj} = \eta \sum_t \left[\sum_i (r_i^t - y_i^t) v_{ih} \right] z_h^t (1 - z_h^t) x_j^t \tag{2.4}$$

```
Initialize all v_ih and w_hj to rand(-0.01,0.01)
Repeat
   For all (x^t, r^t) ε X in random order
        For h=1,…,H
               z_h ← sigmoid(w^T_h x^t )
        For i=1,…,K
               y_i = v^T_i z
        For i=1,…,K
              Δv_i ← η(r^t_i - y^t_i) z
        For h=1,…,H
              Δw_h ← η( Σ_i (r^t_i - y^t_i) v_ih) z_h (1- z_h) x^t
        For i=1,…,K
               v_i ← v_i + Δv_i
        For h=1,…,H
               w_i ← w_i + Δw_i
Until Convergence
```

Figure 2.8 MLP Algorithm ([4])

MLP (Multilayer Perceptron Algorithms) has three steps train, validate and test. Hidden units are a parameter and the hidden unit that gives the least error at train part, is used at validate part. At the MlpVariables text file, the apoch number and learning factor of MLP algorithm exists. If numbers of epochs are increased, the error on the training set decreases, but the error on the validation set starts to increase. Initially all the weights are close to 0 and thus have a little effect. As training continues the most important weights start moving away from 0 and are utilized [4].

The hidden number for MLP was increased which has more than 2 classes datasets. If class numbers increase, hidden number also should be increased for better results.

2.5 C4.5 Algorithm

Classification is the subject of the data mining and purpose of it is to compose a model or classify pre – classified examples [2]. Among the numerous approaches are regressions, Bayesian models, neural networks and classification trees just to name a few. Decision tree is especially an useful option when the tasks are to classify or predict outcomes and to produce easy-to-interpret rules.

The top-down induction of decision trees (TDIDT) is a method (supervised learning) in the building of classification trees [2]. Purpose of the top down induction of C4.5 is to create a decision tree that is as small as possible and suitable for the data [3].

Another key aspect in building decision trees is the decision on when to stop growing the tree. By using prepruning techniques, the growing process is stopped the tree for better future prediction.

The decision tree is built by recursively dividing the data set of examples into subsets according to some splitting criterion (splitting test). The splitting criterion is very important in the process of building the tree, because it determines if we must attach a node or a leaf as next element in the tree.

C4.5 Algorithms has three steps train, validate and test.

```
 C4_5_Algorithm() {
      RecursiveC4_5_Algorithm()
      PruneValidateAndCalculateError()
      CostCurrentNode← SearchAndTestAndCalculateError()
      Return CostCurrentNode
      }
RecursiveC4_5_Algorithm() {
      bestRowSplit ← BestSplit(currentNode)
      if((bestRowSplit != -1)&&(bestRowSplit>=0)){
      RecursiveC4_5_Algorithm(currentNode->left)
      RecursiveC4_5_Algorithm(currentNode->right)
      else
      currentNode->left ← NULL;
      currentNode->right ← NULL;
      currentNode ->bestColumn ← -1;
      }
PruneValidateAndCalculateError(rootNode) {
      ErrorCurrentNode ← SearchAndTestAndCalculateError(rootNode,i)
      currentNode ->oldBestColumn =currentNode ->bestColumn;
      currentNode ->bestColumn =-1;
      PruneErrorCurrentNode←SearchAndTestAndCalculateError(rootNode,i);
      if(PruneErrorCurrentNode > ErrorCurrentNode)

            currentNode ->bestColumn = currentNode ->oldBestColumn;
            if(currentNode->left->bestColumn != -1)
            PruneValidateAndCalculateError(rootNode, currentNode->left)

            if(currentNode->right->bestColumn != -1)
            PruneValidateAndCalculateError(rootNode,currentNode->right);
      }
```

Figure 2.9 C4.5 Algorithm

If all branches of a root node are in the same class, they are pure. If not, they are impure. The branch is checked if it is pure or not. If it is not pure, it is split again. The tree is split until the leaf is pure, and the process is not continued any further. Tree construction continues recursively for all branches that are not pure. If a given branch has a higher error rate than a simple leaf would, the branch is replaced with a leaf. By applying this heuristic from the bottom to the top of the tree, it is possible to prune back the tree for better future prediction. It is possible to grow a very large tree. The quality of a test is measured by the impurity.

Entropy is the function that is used to measure impurity. Current root is divided to two parts according to best entropy. Minimum entropy (I) is the best entropy.

$$I = \sum_{i=1}^{k} P_L^i \log_2 P_L^i + P_R^i \log_2 P_R^i \qquad (2.5)$$

Examining the split that minimizes impurity after the split allows generation of the smallest tree. If the subsets after the split are closer to pure, fewer splits will be needed afterward. There is no guarantee of finding the smallest decision tree [4].

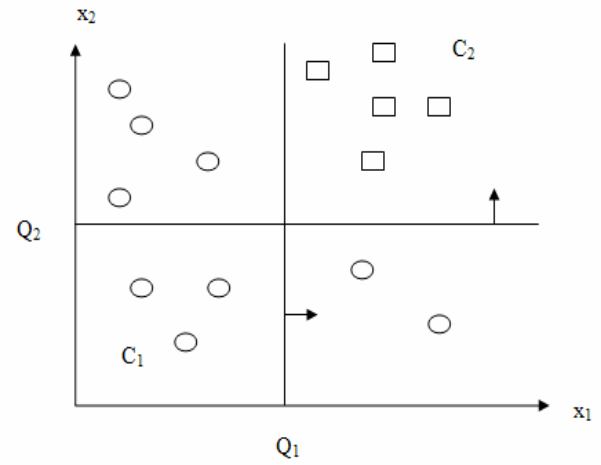


Figure 2.10 Example of dataset

Figure 2.10 shows that input is divided to local regions. According to best entropy root is divided to two parts.

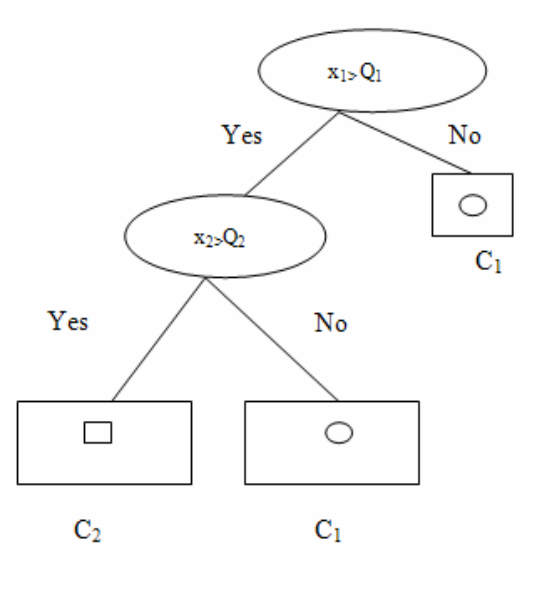


Figure 2.11 Decision Tree

Figure 2.1 shows that oval nodes are the decision nodes and rectangles are leaf nodes. The tree is divided until the leaf is pure.

Chapter 3
Cost Sensitive Learning

During decision making process misclassification errors result in additional cost. To prevent the classification errors balance of the negative examples can be increased [5].

Highly non-uniform misclassification costs are very common for solving real-life data mining problems, such as fraud detection, medical diagnosis and various problems in business decision-making. Model becomes useless when the cost is ignored. Because model classifies each example of the most frequent class [6].

The cost of performing a certain test on a given patient may be conditional. For example, the cost of a blood test is conditional on the patient's age; the cost of an exercise stress test on a patient may be conditional on whether the patient has heart disease or not. Because stress test could cause heart failure, which adds to the total cost of the test [7].

3.1 Misclassification Cost

In general, misclassification costs may be described by an arbitrary cost matrix C, with C(i, j) being the cost of prediction that an example belongs to class i, while, in reality it belongs to class j [8].

In image or text retrieval, the cost of not displaying a relevant item may be lower or higher than the cost of displaying an irrelevant item. In medicine, the cost of prescribing a drug to an allergic patient can be much higher than the cost of not prescribing the drug to a non-allergic patient, if alternative treatments are available. In many applications, not all misclassifications have the same value. For example, in medical diagnosis, classifying a diabetic sick patient is often far more costly than labeling a healthy patient as sick.

3.2 Cost Models

Cost models are used for composing cost matrix and experiments. Each experiment involves testing several different cost matrixes. These matrixes were generated randomly based on eight different cost models which are shown in Table 3.1. The diagonal elements of cost matrix are not zero for cost models M7 and M8 while it is zero for other six cost models. In cost model M5, for example, the cost of mislabeling an example from class j as belonging to class i is determined by the ratio of the number of examples in class i to the number of examples in class j. In particular, when class i is very common, class j is very rare, which make this mistake (on the average) very expensive, because $P_{(i)}/P_{(j)}$ will be very large number. $P_{(i)}$ is the probability of class i and $P_{(j)}$ is probability of the class j. In the case of the cost model M6, we reversed the relationship, so that the least expensive errors are those that mislabel a rare class j as belonging to a common class i. Finally, model M8 is similar to model M2 in most part, with the exception of none zero costs on the diagonal of C of M8 [8].

When a dataset has equal number of classes, model M5 and M6 will function similar to M3, however they can not have the same cost matrix, since they are composed randomly. For each dataset, cost matrixes of all cost models were generated randomly.

Table 3.1 Cost Models from M1 to M8

Cost Model	C(i,j) i≠j	C(i,i)
M1	Unif[0,10]	0
M2	Unif[0,100]	0
M3	Unif[0,1000]	0
M4	Unif[0,10000]	0
M5	Unif[0,1000 x P(i)/P(j)]	0
M6	Unif[0,1000 x P(j)/P(i)]	0
M7	Unif[0,10000]	Unif[0,1000]
M8	Unif[0,100]	Unif[0,10]

3.3 Cost Methods

3.3.1 Class Frequency Method

In this method, class frequencies are calculated for each class in the dataset. For example, $ClassNumber_{[i]}$ has the number of examples belonging to class i. ($ClassNumber_{[i]}$ * $costWeight_{[i]}$) is constant for all classes which gives higher weight to classes that are less frequent. Finally number of data which belongs this class is calculated.

```
1 For i=0,…,ClassCount
2 costWeightTotal=0.0;
3    For j=0,…,ClassCount
4       if(i!=j)
5       costWeightTotal←costWeightTotal+
(double)ClassNumber_i/ClassNumber_j
6       costWeight_i  ←    (double)ClassCount/(1.0+
costWeightTotal)
```

Figure 3.1 Class Frequency Method

After calculation of the weights, results are normalized. Data ($ClassNumber_{[K]}$/ $costWeight_{[K]}$) is randomly chosen for each class and then ten fold is applied.

3.3.2 MaxCost Method

Each weight of the class is computed as the maximum corresponding column. The purpose of the maximum value within a column is the worst case cost of misclassifying an example whose true class corresponds to that column.

```
1   For j=0,…,ClassCount
2      max  ←  0.0;
3      For i=0,…,ClassCount
4         if(max < CostMatrix_ij)
5         max  ←  CostMatrix_ij
6         costWeight_j ←  max;
```

Figure 3.2 MaxCost Method

Similar to Class Frequency Method, results of calculated weights are normalized and data as ($ClassNumber_{[k]}$/ $costWeight_{[k]}$) is randomly chosen for each class and then ten fold is applied.

3.3.3 AvgCost Method

In this method, each weight of the class is computed as the mean of the off-diagonal elements in the corresponding column.

```
1  For j=0,…,ClassCount
2     sum=0.0;
3         For i=0,…,ClassCount
4           if(i!=j)
5                 sum←sum+CostMatrix[i][j];
6          costWeight[j] ← sum/(ClassCount -1)
```

Figure 3.3 AvgCost Method

Again remaining steps are similar to two other previous methods; results of calculated weights are normalized and data as ($ClassNumber_{[k]}$/ $costWeight_{[k]}$) is randomly chosen for each class and then ten fold is applied.

Chapter 4
Experiments

4.1 Experiment Setup

All data sets were taken from UCI repository [9]. For each dataset, two text files were made. One of these file keeps data itself the other one holds data information which has number of rows, number of columns, number of classes, class column of all data records, fold number, percentage of train, and class values. For example, putting class column to first column is needed for using these algorithms. For example Breast cancer data has two classes which are benign and malignant and were used 2.0 instead of benign and 4.0 instead of malignant. Ten fold were used for all data sets, however the fold number can also be changed in the data information. Our program also works well for different folds and various percentages. The train percentage was used as %63 for all dataset. Ten percent of data was used for Test purpose. %27 of the data was used to validate the data. Rows were randomly shuffled 5 times to prevent redundancy in the test, train or validate data.

Table 4.1 Ten data sets

Dataset	# of Instances	# of Features	Number of Classes	Source
Diabetic	44	19	2	Çapa Hospital
Iris	150	5	3	UCI
Dermatology	366	35	6	UCI
Breast Cancer	699	10	2	UCI
Pima Indian	768	9	2	UCI
Yeast	1484	9	10	UCI
Segmentation	2310	20	7	UCI
WaveForm	5000	22	3	UCI
PenDigits	7494	17	10	UCI
Letter	20000	17	26	UCI

Table 4.2 shows the datasets sorted in ascending by number of data and the data properties such as Number of instances, Number of Features, Number of classes of each data set, and the source of the data sets.

4.1.1. Normalization

Before using the data a normalization step, the process of removing statistical error in repeated measured data, is performed for all algorithms except Decision Tree (C4.5) algorithm.

The formula of the normalization is

$$Z = (X-u) /s \quad (4.1)$$

where Z is the normalization of the data, X is data records, u is the mean of each column and s is the standard deviation. We will obtain a vector Z that has normal distribution with zero mean and unit variance

The standard deviation(s) can be calculated by using the formula:

$$s=\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})^2} \quad (4.2)$$

Where s is the standard deviation, n is the number of rows, and x is data record. Here, to remove the effect of total number of data, we divide the sum of square deviation with the number of data to produce average of sum of square deviation.

```
1 //Calculation of sum of each column
2  for i=0,…, column
3   for j=0,…, row
4    sum_i ←  sum_i+ DataRecords_ji
5 //Mean of each column
6 for i=0,…, column
7 mean_i ←  (double)sum_i/row
8  for i=0,…, column
9    for j=0,…, row
10   stdrow_i←stdrow_i+((DataRecords_ji-
ean_i)*(DataRecords_ji-mean_i))
11 //Calculation of the standard deviation
12 for i=0,…, column
13      std_i ← (double)sqrt(stdrowi/(row-1))
14 for i=0,…, column
15      for j=0,…, row
16   if(std_[i]!=0)
17            DataRecords_ji ← (DataRecords_ji-mean_i)
/ std_i;
18    else
19   DataRecords_ji ← 0;
```

Figure 4.1 Normalization Data

The code to calculate the normalization is given in the Figure 4.2. Each data was normalized for the Logistic Discrimination Algorithm (LD), K-Nearest Neighbor Algorithm (KNN), Multilayer Perceptron Algorithm (MLP) and Nearest Mean algorithm; data was not normalized for Decision Tree Algorithm (C4.5) algorithm. During the normalization process first column was omitted. Because, the first column contains class values which are supported constant.

4.1.2 Dividing Data to Train, Test and Validate

Each data was divided to three parts such as Train Set, Test Set and Validate Set and rows of the data were shuffled. In the algorithms, each set have different percentage of data and each have equal percentage of classes according to their percentage.

If we give an example of our life for train, test and validate set, the example problems that the instructor solves in class while teaching a subject form the training set; exam questions are the validation set; and the problems we solve in our later, professional life are the test set.

The data was trained with train set. Validation was used to choose the best model, and it has effectively become a part of the training set. Validation set was used at K Nearest Neighbor algorithm for finding the best (k) and at MLP algorithm for finding the best hidden unit (h). The Best *k* was found and tested with test data at K Nearest Neighbor algorithm. Best hidden unit number was found and tested with test data at MLP algorithm. The outputs of the test set are only used to evaluate the performance of the model and are not used during training step.

4.2 Experiment Results of Classification Learning Algorithms Errors

In data mining applications, it is important to develop evaluation methods for selecting quality and profitable rules. For instance data needs to be ranked which is sorting the data in ascending order. When it is ranked in ascending order, the lower values are given, the numerically lower rank. For Example, the lower the score is better. Therefore lower values are assigned lower numerical rankings. With the costs {52, 98, 65, 99}, the value 52, the lower score in the list, is given a rank of 1 and 99, the highest score in the list, is given a 4 of rank.

Table 4. 3 Classification Learning Algorithms Error Results with standard deviation[1]

	LD	**KNN**	**MLP**	**N. MEAN**	**C4.5**
Diab.	35,00 ± 21,0	30,00 ± 28,38	35,00 ± 17,48	27,50 ± 24,86	55,00 ± 10,54
Iris	4,00 ± 5,62	6,00 ± 6,62	4,00 ± 5,62	13,33 ± 5,44	10,66 ± 7,16
Der.	2,57 ± 2,84	4,57 ± 5,25	8,57 ± 4,46	3,42 ± 2,95	10,57 ± 6,46
Breast	3,47 ± 3,49	3,62 ± 3,07	3,62 ± 3,36	3,76 ± 3,29	*17,24 ± 13,79*
Pima	22,76 ± 4,11	28,55 ± 4,34	23,02 ± 4,12	26,31 ± 4,60	34,21 ± 0,00
Yeast	41,77 ± 3,73	44,41 ± 5,08	44,48 ± 3,66	49,86 ± 2,87	66,62 ± 3,50
Seg.	8,70 ± 5,25	7,79±7,09	12,29±6,30	16,75 ± 8,56	19,82 ± 6,78
Wave.	13,29 ± 1,57	17,65 ± 2,05	13,13 ± 1,67	18,93 ± 1,02	*57,02 ± 10,00*
Pen.	4,09 ± 0,94	0,79 ± 0,34	11,15 ± 2,32	15,70 ± 1,45	*70,14 ± 7,92*
Letter	23,09 ± 1,47	5,66 ± 0,47	*42,34±1,61*	43,02 ± 1,65	*91,39 ±0,41*

[1] There can be a problem with C4.5 Algorithm implementation for breast cancer, waveform and letter datasets. There can be also a problem with MLP Algorithm implementation for letter dataset.

Table 4.4 Ranking of Classification Algorithms Errors

	LD	KNN	MLP	N. MEAN	C4.5
Diabetic	3	2	3	1	5
Iris	1	2	1	5	4
Dermatology	1	3	4	2	5
Breast Cancer	1	2	2	3	5
Pima Indian	1	4	2	3	5
Yeast	3	1	2	4	5
Segmentation	2	1	3	4	5
WaveForm	2	3	1	4	5
PenDigits	2	1	3	4	5
Letter	2	1	3	4	5
AVERAGE	1,80	2,00	2,40	3,40	4,90

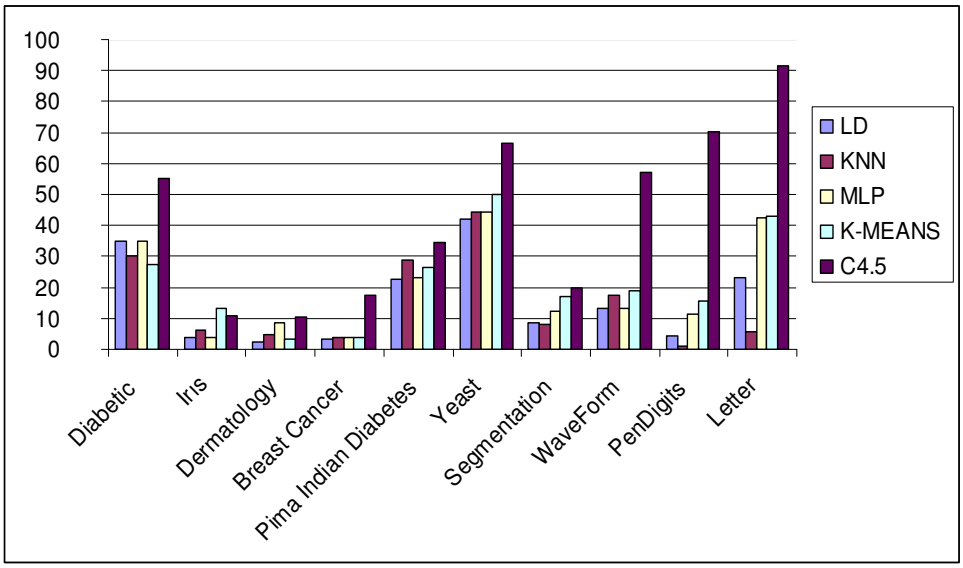


Figure 4.3 Graphic of the Classification Learning Algorithms Error Results

As shown in Table 4.2, Table 4.3, Figure 4.2 Logistic Discrimination Algorithm is much better because it has the smallest error among 5 algorithms in all datasets. LD algorithm is also better at the datasets which has smaller instances and classes. KNN is much better at datasets which have biggest datasets and number of classes. The worst algorithm is C4.5 Algorithm which has the biggest error in almost all datasets.

4.3 Experiment Results of Cost Sensitive Learning Algorithms

Ten fold is applied to all algorithms and results of each fold were written to a text file.

Table 4.5 Best costs among Classification Algorithms

	LD	KNN	MLP	N. MEAN	C4.5
M1	2,03	1,77	2,73	3,93	4,47
M2	2	1,83	3,1	4,13	3,93
M3	2,2	1,47	3,03	4,03	4,23
M4	2,13	1,73	2,9	4,07	4,13
M5	2,03	2,07	3,37	3,37	4,13
M6	2,07	1,9	3,2	3,83	3,97
M7	1,97	1,57	3	4,07	4,3
M8	2,03	1,77	3,17	4,1	3,9

Comparison of M1, M2, M3 and M4 cost models

Table 4.6 shows that among Logistic Discrimination Algorithm results, the best one is M2 because it has least cost among all four cost models. If one compares K-Nearest Neighbor algorithms, the best one is M3 Cost model. Among the Multilayer Perceptron Algorithm, the best one is M1. Among Nearest Mean Algorithm, the best one is M1. Among C4.5 Algorithm, the cost model result, M2 is the best one. Lower cost results make the model better.

Comparison of M5 and M6 cost models

Between Logistic Discrimination Algorithm results, the best one is M5 because it has least cost between two cost models. If we compare K-Nearest Neighbor algorithms, the best one is M6 Cost model. Between Multilayer Perceptron Algorithm, the best one is M6. When compared with Nearest Mean, the best one is M5. Between C4.5 Algorithm cost model results M6 is best one.

Comparison of M7 and M8 cost models

Between Logistic Discrimination Algorithm results, the best one is M7 because it has least cost between two cost models. If we compare K-Nearest Neighbor algorithms, the best one is M7 Cost model. Between Multilayer Perceptron Algorithm, the best one is M7. When compared with Nearest Mean, the best one is M7. Between C4.5 Algorithm cost model results M8 is the best one.

Table 4.7 Average of the best costs among Classification Algorithms

	LD	**KNN**	**MLP**	**N. MEAN**	**C4.5**
AVERAGE	2,06	1,76	3,06	3,94	4,13

Table 4.5 shows that KNN is the best algorithm among all other Classification algorithms due to its cost.

Table 4.8 Best Costs among Cost Algorithms

	CW	**MW**	**AW**
M1	1,86	2	2,1
M2	2,1	1,84	2,06
M3	2,28	1,92	1,82
M4	2,18	1,96	1,88
M5	1,8	2,08	2,1
M6	1,98	2,04	1,96
M7	2,06	1,78	2,14
M8	2,06	2,04	1,9

Comparison of M1, M2, M3 and M4 cost models

Table 4.6 shows that among ClassFrequencyWeight Algorithm results, the best one is M1 because it has least cost among four cost models. If we compare

MaxCostWeight algorithm best one is M2 Cost model. Among AvgCostWeight best one is M3.

Comparison of M5 and M6 cost models

Between ClassFrequencyWeight Algorithm results, the best one is M5. Because it has least cost between two models. When compared with MaxCostWeight algorithm, the best one is M6 Cost model. Between AvgCostWeight the best one is M6.

Comparison M7 and M8 cost models

Between ClassFrequencyWeight Algorithm results two results are the same. If we compare MaxCostWeight algorithm, the best one is M7 Cost model. Between AvgCostWeight, the best one is M8.

Table 4.9 Average of Best Costs among Cost Algorithms

	CW	**MW**	**AW**
AVERAGE	2,04	1,9575	1,995

Table 4.7 shows that in all Cost algorithms Max Cost Weight is the best one because it has least cost. There is no much difference at the results.

Table 4.10 Average of Cost Algorithms versus classification algorithms ranking

	LD	**KNN**	**MLP**	**NEAREST MEAN**	**C4.5**
CW	2,15	1,8125	2,7	4,2	4,05
MW	2	1,6375	3,2125	3,95	4,1875
AW	2,08	1,8	3,2375	3,675	4,1625

Again KNN Algorithm seems better at all cost algorithms. Max Cost Weight Algorithm is more successful at Table 4.11.

Table 4.12 Data sets versus classification algorithms Ranks for ClassFrequencyWeight Algorithm

		LD	KNN	MLP	N. MEAN	C4.5
CW	**Diabetic**	2,00	2,13	2,25	5,00	2,75
CW	**Iris**	2,00	1,75	2,88	5,00	3,25
CW	**Dermatology**	1,00	2,13	4,38	3,25	4,25
CW	**Breast Cancer**	1,50	2,63	2,38	4,38	4,13
CW	**Pima Indian Diabetes**	3,88	2,50	1,38	4,50	2,75
CW	**Yeast**	3,00	1,13	2,50	4,25	4,13
CW	**Segmentation**	1,88	1,13	3,00	4,00	5,00
CW	**WaveForm**	2,25	2,00	2,13	3,88	4,75
CW	**PenDigits**	1,88	1,13	3,25	4,25	4,50
CW	**Letter**	2,13	1,63	2,88	3,50	5,00

Table 4.9 shows that LD is better at Diabetic, Dermatology, and Breast Cancer datasets. KNN is better at Iris, Yeast, Segmentation, PenDigits and Letter datasets. Data sets versus classification algorithms Ranks for ClassFrequencyWeight Algorithm are more successful at KNN.

Table 4.13 Data sets versus classification algorithms Ranks for MaxCostWeight Algorithm

		LD	KNN	MLP	N. MEAN	C4.5
MW	**Diabetic**	2,13	2,25	1,75	3,88	5,00
MW	**Iris**	1,63	2,75	3,88	4,88	1,75
MW	**Dermatology**	1,00	3,00	5,00	3,50	2,50
MW	**Breast Cancer**	2,25	1,50	2,25	4,38	4,63
MW	**Pima Indian Diabetes**	2,88	1,00	3,25	3,00	4,88
MW	**Yeast**	2,38	1,63	3,00	3,25	4,75
MW	**Segmentation**	1,88	1,13	3,38	4,88	3,75
MW	**WaveForm**	1,88	1,13	3,13	4,00	4,88
MW	**PenDigits**	2,00	1,00	3,00	4,25	4,75
MW	**Letter**	2,00	1,00	3,50	3,50	5,00

Table 4.10 shows that LD is better at Diabetic, Iris, and Dermatology datasets. KNN is better at Breast Cancer, Pima Indian Diabetes, Yeast, Segmentation, WaveForm, PenDigits, and Letter datasets. Data sets versus classification algorithms Ranks for MaxCostWeight Algorithm is more successful at KNN.

Table 4.14 Data sets versus classification algorithms Ranks for AvgCostWeight Algorithm

		LD	KNN	MLP	N. MEAN	C4.5
AW	**Diabetic**	1,38	3,25	3,75	1,88	4,38
AW	**Iris**	1,50	2,63	3,88	4,88	2,00
AW	**Dermatology**	1,25	2,63	4,88	3,00	3,25
AW	**Breast Cancer**	3,13	1,75	1,88	3,38	4,88
AW	**Pima Indian Diabetes**	3,00	1,13	3,50	3,88	3,50
AW	**Yeast**	2,25	1,63	2,88	4,13	4,13
AW	**Segmentation**	1,88	1,25	2,88	4,00	5,00
AW	**WaveForm**	2,13	1,50	2,38	4,00	5,00
AW	**PenDigits**	2,00	1,00	3,25	4,25	4,50
AW	**Letter**	2,25	1,25	3,13	3,38	5,00

LD is better at Diabetic, Iris, and Dermatology datasets. KNN is better at Breast Cancer, Pima Indian Diabetes, Yeast, Segmentation, WaveForm, PenDigits, and Letter datasets. Data sets versus classification algorithms Ranks for AvgCostWeight Algorithm is more successful at KNN.

For each cost algorithm (Class Frequency, Maximum cost, and Average cost weight) of M1 model cost results graphic were drawn. The difference of cost results are very high, so logarithmic scale graphics were drawn. When data is spread on an extremely large area the graph will be very compact you may miss some sharp drops in values, so logarithmic graphs were used.

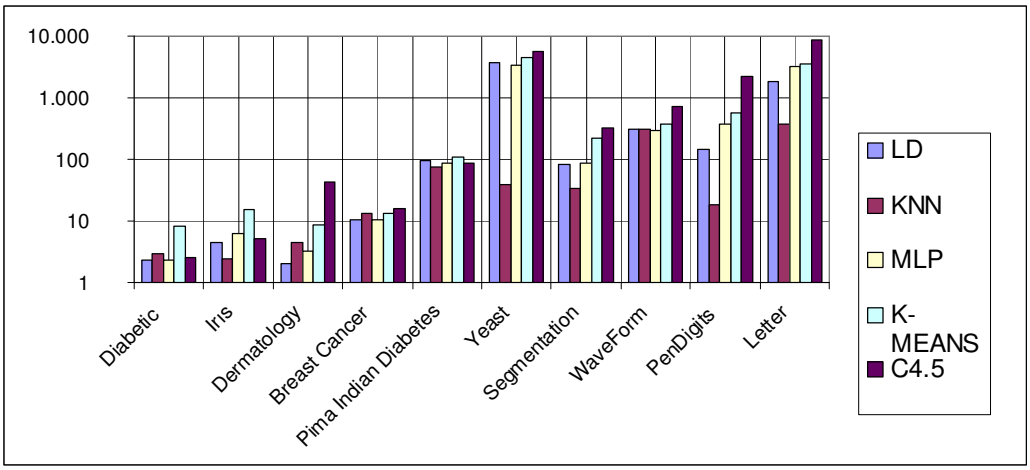


Figure 4.4 Class Frequency weight cost algorithm cost result of M1 model

Table 4.15 Class Frequency weight cost algorithm cost result Ranking of M1 model

	LD	KNN	MLP	N. MEAN	C4.5
Diabetic	1	4	1	5	3
Iris	2	1	4	5	3
Dermatology	1	3	2	4	5
Breast Cancer	2	4	1	3	5
Pima Indian	4	1	2	5	3
Yeast	3	1	2	4	5
Segmentation	2	1	3	4	5
Waveform	3	2	1	4	5
Pen Digits	2	1	3	4	5
Letter	2	1	3	4	5
AVERAGE	**2,20**	**1,90**	**2,20**	**4,20**	**4,40**

Figure 4.3 and Table 4.12 shows that KNN is better algorithm in almost all datasets according to cost. It is usually has least cost at M1 model when Class Frequency weight cost algorithm was used.

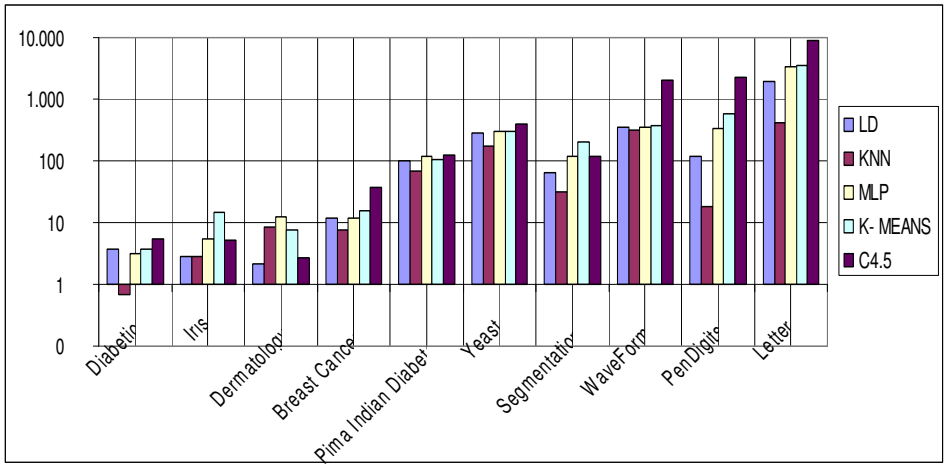


Figure 4.5 Maximum cost weight cost algorithm cost result of M1 Model

Table 4.16 Maximum cost weight cost algorithm cost result Ranking of M1 model

	LD	KNN	MLP	N. MEAN	C4.5
Diabetic	3	1	2	4	5
Iris	1	1	4	5	3
Dermatology	1	4	5	3	2
Breast Cancer	3	1	2	4	5
Pima Indian	2	1	4	3	5
Yeast	2	1	3	4	5
Segmentation	2	1	3	5	4
WaveForm	2	1	3	4	5
PenDigits	2	1	3	4	5
Letter	2	1	3	4	5
AVERAGE	**2,00**	**1,30**	**3,20**	**4,00**	**4,40**

Figure 4.6 and Table 4.17 KNN shows that in Maximum cost weight cost algorithm is better algorithm in almost all datasets according to cost. It is usually has least cost at M1 model when Maximum cost weight cost algorithm was used. Worst one is again C4.5.

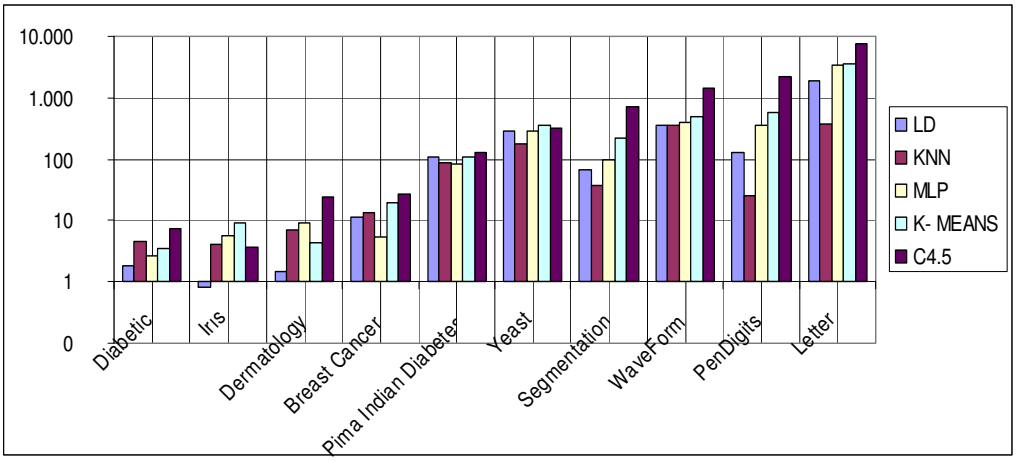


Figure 4.7 Average cost weight cost Algorithm cost result of M1 model

Table 4.14 Maximum cost weight cost algorithm cost result Ranking of M1 model

	LD	KNN	MLP	N. MEAN	C4.5
Diabetic	1	4	2	3	5
Iris	1	3	4	5	2
Dermatology	1	3	4	2	5
Breast Cancer	2	3	1	4	5
Pima Indian	3	2	1	4	5
Yeast	2	1	3	5	4
Segmentation	2	1	3	4	5
WaveForm	2	1	3	4	5
PenDigits	2	1	3	4	5
Letter	4	1	2	3	5
AVERAGE	**2,00**	**2,00**	**2,60**	**3,80**	**4,60**

Figure 4.8 and Table 4.14 shows that KNN and LD are better algorithms in almost all datasets according to cost. It is usually has least cost at M1 model when Average cost weight cost algorithm was used.

Chapter 5
Conclusion

Logistic Discrimination (LD), K-Nearest Neighbor (KNN), Multilayer Perceptron (MLP), Decision Tree (C4.5) and Nearest Mean classification algorithms were studied for calculating class cost and error. For all algorithms, datasets needed to be normalized except Decision Tree (C4.5) algorithm. In this study, datasets were taken from UCI repository and Çapa hospital Istanbul, Turkey. Therefore, this study did not involve the data acquisition step. Data has different kinds of classes and features and therefore datasets were cleaned and formed in suitable format for to be read from the file.

K-Nearest Neighbor Algorithm (KNN), Multilayer Perceptron (MLP), and Decision Tree (C4.5) Algorithms had three steps; train, validate and test. Nearest Mean and Logistic Discrimination algorithms had only train and test steps. Different kinds of data sets, such as Iris, Dermatology, Diabetic, and Breast Cancer were used. Although these algorithms can be used for different kinds of data applications, medical records data were mostly used in this study.

In many applications, not all misclassifications have the same value. There may be a significant difference between the problems caused by a false negative and those caused by a false positive. For example, in medical diagnosis, classifying a diabetic sick patient is often far more costly than labeling a healthy patient as sick.

Cost models are used for composing cost matrix and experiments. Each experiment involves testing several different costs Matrix. These matrixes were randomly generated based on eight different cost models. ClassFrequencyWeight, MaxCostWeight and AvgCostWeight cost algorithms were used for dataset weights. Data ($ClassNumber_{[K]}$/ $costWeight_{[K]}$) is randomly chosen for each class and then ten fold is applied. costWeight is obtained by using cost algorithms. ClassNumber is the number of the classes at the datasets.

In general, Logistic Discrimination Algorithm is the best algorithm, because it has the smallest error among 5 algorithms in all datasets. LD algorithm is better at the datasets which has smaller instances and classes. KNN is better at datasets which have biggest datasets and number of classes. Worst algorithm is C4.5 Algorithm which has the biggest error in almost all datasets.

M1, M2, M3 and M4 cost models were compared for classification algorithms. Among Logistic Discrimination Algorithm results, the best one is M2 because it has least cost among four cost models. If one compares K-Nearest Neighbor algorithms, the best one is M3 Cost model. Among Multilayer Perceptron Algorithm, the best one is M1. Among Nearest Mean Algorithm, the best one is M1. Among C4.5 Algorithm, the cost model results, M2 is best one. Lower cost results make the model better.

M5 and M6 cost models were compared for classification algorithms. Between Logistic Discrimination Algorithm results, the best one is M5 because it has least cost between two cost models. If we compare K-Nearest Neighbor algorithms, the best one is M6 Cost model. Between Multilayer Perceptron Algorithm, the best one is M6. When compared with Nearest Mean, the best one is M5. Between C4.5 Algorithm cost model results M6 is best one.

M7 and M8 cost models were compared for classification algorithms. Between Logistic Discrimination Algorithm results, the best one is M7 because it has the least cost between two cost models. If we compare K-Nearest Neighbor algorithms, the best one is M7 Cost model. Between Multilayer Perceptron Algorithm, the best one is M7. When compared with Nearest Mean, the best one is M7. Between C4.5 Algorithm cost model results M8 is best one.

M1, M2, M3 and M4 cost models were compared for cost algorithms. Among ClassFrequencyWeight Algorithm results, the best one is M1 because it has least cost among four cost models. If we compare MaxCostWeight algorithm, the best one is M2 Cost model. Among AvgCostWeight, the best one is M3.

M5 and M6 cost models were compared for cost algorithms. Between ClassFrequencyWeight Algorithm results, the best one is M5 because it has least cost between two models. When compared with MaxCostWeight algorithm, the best one is M6 Cost model. Between AvgCostWeight, the best one is M6.

M7 and M8 cost models were compared for cost algorithms. Between ClassFrequencyWeight Algorithm results two results are the same. If we compare

MaxCostWeight algorithm, the best one is M7 Cost model. Between AvgCostWeight, the best one is M8.

In ClassFrequencyWeight LD is better at Diabetic, Dermatology, and Breast Cancer datasets. KNN is better at Iris, Yeast, Segmentation, PenDigits and Letter datasets. Data sets versus classification algorithms Ranks for ClassFrequencyWeight Algorithm are more successful at KNN.

In MaxCostWeight Algorithm LD is better at Diabetic, Iris, and Dermatology datasets. KNN is better at Breast Cancer, Pima Indian Diabetes, Yeast, Segmentation, WaveForm, PenDigits, and Letter datasets. Data sets versus classification algorithms Ranks for MaxCostWeight Algorithm are more successful at KNN.

In AvgCostWeight Algorithm LD is better at Diabetic, Iris, and Dermatology datasets. KNN is better at Breast Cancer, Pima Indian Diabetes, Yeast, Segmentation, WaveForm, PenDigits, and Letter datasets. Data set versus classification algorithms Ranks for AvgCostWeight Algorithm is more successful at KNN.

Appendix A

In the appendix all costs of 8 cost models exist. In each cost model ten datasets's cost results according to LD, KNN, MLP, NEAREST MEAN, and C4.5 algorithms exist.

Table A.1 Cost M1 Model Results of Algorithms

		LD	KNN	MLP	N.MEAN	C4.5
Diabetic	**CW**	2,3	2,9	2,3	8,3	2,6
	MW	3,7	0,7	3,2	3,8	5,4
	AW	1,8	4,5	2,6	3,4	7,4
Iris	**CW**	4,4	2,4	6,3	15,6	5,2
	MW	2,8	2,8	5,5	14,8	5,2
	AW	0,8	4	5,6	9,2	3,6
Dermatology	**CW**	2	4,6	3,2	8,8	43,1
	MW	2,1	8,3	12,5	7,4	2,7
	AW	1,5	6,9	9,3	4,4	23,8
Breast	**CW**	10,7	13,5	10,3	13,2	16
	MW	12	7,7	11,9	15,7	36,9
	AW	11,5	13,3	5,3	19,7	27,5
Pima Indian	**CW**	94,3	76,1	85,7	107,4	88,5
	MW	99,7	67,5	118,1	107	122,4
	AW	106,8	87,6	83,5	107	127,1
Yeast	**CW**	3.779,50	39,8	3.347,80	4.585,50	5.796,00
	MW	282	170	297,8	306,2	397,1
	AW	286,2	177,4	289	354	318,9
Segmentation	**CW**	83,2	33,9	85,1	218,6	321,1
	MW	64	31	118,8	199,7	119,5
	AW	68,1	36,4	98,7	220,1	706,8
WaveForm	**CW**	311	304,9	294,9	380,1	709,9
	MW	349,1	313,2	362	366	2.003,90
	AW	358,3	351,8	398,5	486,3	1.454,00
PenDigits	**CW**	144,6	18,8	374,3	555,9	2.267,70
	MW	119,2	18,2	330,7	571,8	2.271,40
	AW	124,8	26	353,6	592,5	2.235,80
Letter	**CW**	1.833,00	377,6	3.290,10	3.584,00	8.702,70
	MW	1.899,00	406,8	3.361,10	3.556,90	8.724,60
	AW	1.890,60	373,2	3.381,40	3.531,70	7.600,20

Table A.2 Cost M2 Model Results of Algorithms

		LD	KNN	MLP	N. MEAN	C4.5
Diabetic	**CW**	26	23	34	61	28
	MW	18	23	8	37	56
	AW	28	33	28	50	67
Iris	**CW**	62	42	67	211	71
	MW	35	50	58	170	31
	AW	37	41	82	150	37
Dermatology	**CW**	25	58	67	67	712
	MW	16	70	171	81	56
	AW	36	63	194	62	22
Breast	**CW**	107	117	145	167	132
	MW	48	54	83	124	112
	AW	100	52	70	145	143
Pima Indian	**CW**	613	595	512	1.487	522
	MW	1.262	644	1.219	970	2.171
	AW	1.075	755	1.207	1.052	1.562
Yeast	**CW**	19.940	340	19.509	44.930	25.055
	MW	2.265	1.560	2.244	2.985	2.848
	AW	2.187	1.664	2.147	3.385	3.026
Segmentation	**CW**	689	387	800	1.733	1.844
	MW	516	318	904	1.597	1.457
	AW	494	285	822	1.675	2.353
WaveForm	**CW**	3.975	4.097	4.170	7.484	7.478
	MW	4.197	3.893	4.523	7.286	16.898
	AW	4.107	4.242	4.424	6.319	8.357
PenDigits	**CW**	1.351	251	3.572	6.848	37.179
	MW	1.155	343	3.975	6.734	38.493
	AW	1.276	469	3.988	6.669	26.308
Letter	**CW**	21.928	4.334	41.000	39.197	86.450
	MW	23.098	4.516	40.350	39.012	78.449
	AW	22.865	4.610	41.320	39.157	90.230

Table A.3 Cost M3 Model Results of Algorithms

		LD	KNN	MLP	N. MEAN	C4.5
Diabetic	**CW**	328	280	328	770	304
	MW	46	280	0	303	586
	AW	210	256	304	163	657
Iris	**CW**	267	160	311	756	222
	MW	222	122	431	588	54
	AW	54	15	202	428	92
Dermatology	**CW**	212	487	10.580	678	5.384
	MW	234	788	8.489	608	547
	AW	256	588	7.856	636	1.168
Breast	**CW**	1.193	1.379	1.431	1.711	1.588
	MW	958	746	937	1.583	3.151
	AW	1.845	887	1.078	1.563	2.195
Pima Indian	**CW**	8.230	7.332	7.176	14.773	7.362
	MW	10.111	7.200	10.576	10.940	13.089
	AW	9.336	7.851	9.697	12.393	10.258
Yeast	**CW**	395.971	303.287	338.206	468.038	678.912
	MW	34.081	33.414	30.592	36.187	44.393
	AW	33.417	32.478	32.731	36.807	57.321
Segmentation	**CW**	5.933	3.028	7.572	17.406	20.331
	MW	4.963	2.615	8.972	17.678	6.387
	AW	5.278	2.801	7.615	15.932	22.661
WaveForm	**CW**	24.859	24.910	25.589	37.812	56.571
	MW	24.456	24.356	26.251	31.962	61.728
	AW	24.422	22.077	25.048	32.583	60.454
PenDigits	**CW**	14.935	2.411	44.334	63.256	320.140
	MW	13.204	2.977	50.916	65.434	328.830
	AW	17.257	2.604	42.524	65.137	387.173
Letter	**CW**	212.489	40.536	368.495	427.628	984.220
	MW	222.251	42.477	382.618	425.575	918.851
	AW	226.278	40.312	385.057	433.012	748.557

Table A.4 Cost M4 Model Results of Algorithms

		LD	KNN	MLP	N. MEAN	C4.5
Diabetic	**CW**	4.667	4.040	4.667	11.125	4.354
	MW	683	4.040	0	4.410	8.338
	AW	3.044	3.727	4.354	2.362	9.334
Iris	**CW**	1.956	2.640	2.280	9.177	3.900
	MW	374	3.423	935	6.079	1.702
	AW	187	2.095	1.719	3.423	0
Dermatology	**CW**	2.229	5.518	83.840	8.331	60.361
	MW	3.260	9.165	137.928	9.235	6.534
	AW	4.041	5.713	128.517	7.231	7.102
Breast	**CW**	17.101	19.832	20.347	24.388	22.876
	MW	16.844	13.173	14.997	23.727	35.680
	AW	26.234	17.784	14.057	20.571	34.628
Pima Indian	**CW**	119.239	105.731	104.186	210.187	106.916
	MW	143.089	100.415	137.550	157.494	163.067
	AW	128.092	115.893	148.954	184.444	146.078
Yeast	**CW**	1.259.413	946.867	1.086.061	3.316.938	2.330.333
	MW	141.809	131.064	159.818	183.727	280.979
	AW	146.012	141.713	146.389	201.278	311.227
Segmentation	**CW**	96.123	43.667	106.701	205.210	257.648
	MW	65.808	36.699	127.554	190.051	83.068
	AW	74.144	35.511	122.683	204.543	419.907
WaveForm	**CW**	347.677	345.019	334.035	313.217	1.008.166
	MW	451.072	417.045	484.992	469.435	1.903.796
	AW	460.538	410.993	441.261	484.530	1.462.276
PenDigits	**CW**	145.723	23.194	393.545	571.100	2.299.670
	MW	124.502	21.883	348.311	572.444	1.937.039
	AW	120.565	32.952	342.079	513.135	1.603.925
Letter	**CW**	2.008.079	396.513	353.659	3.631.987	9.265.075
	MW	2.050.336	404.448	3.610.724	3.649.258	8.716.497
	AW	1.989.848	398.359	3.634.730	3.675.754	8.822.895

Table A.5 Cost M5 Model Results of Algorithms

		LD	KNN	MLP	N. MEAN	C4.5
Diabetic	**CW**	425	448	425	1.261	437
	MW	437	460	708	813	1.307
	AW	625	885	1.084	625	779
Iris	**CW**	337	447	263	1.577	668
	MW	262	763	3.244	1.652	243
	AW	181	310	395	867	162
Dermatology	**CW**	94	303	15.054	545	9.961
	MW	86	309	11.744	494	1.112
	AW	133	988	12.411	335	2.181
Breast	**CW**	2.090	2.806	1.635	2.259	3.387
	MW	2.138	2.985	2.525	3.817	3.387
	AW	4.233	1.287	2.312	1.848	5.230
Pima Indian	**CW**	21.151	161.889	19.609	17.203	20.312
	MW	20.793	14.245	28.119	20.629	21.055
	AW	22.244	15.521	25.182	21.386	16.754
Yeast	**CW**	430.468	359.161	458.654	278.878	821.677
	MW	583.602	423.305	563.733	331.438	870.357
	AW	534.029	444.460	549.613	349.101	791.355
Segmentation	**CW**	9.674	3.746	10.331	19.559	25.654
	MW	6.297	3.328	12.588	19.120	15.165
	AW	8.553	3.174	16.271	16.780	27.896
WaveForm	**CW**	27.740	28.885	29.905	53.249	51.578
	MW	33.377	29.258	37.844	46.815	44.261
	AW	29.725	29.411	30.023	47.708	52.250
PenDigits	**CW**	14.120	2.465	34.200	58.508	250.602
	MW	14.187	2.184	41.340	57.095	415.610
	AW	14.748	3.994	38.356	61.871	264.608
Letter	**CW**	213.699	45.673	390.346	393.967	938.744
	MW	218.064	49.585	404.980	392.932	843.224
	AW	210.648	43.019	414.965	403.496	902.828

Table A.6 Cost M6 Model Results of Algorithms

		LD	KNN	MLP	N.MEAN	C4.5
Diabetic	**CW**	421	362	421	995	391
	MW	60	362	0	392	752
	AW	271	332	391	211	842
Iris	**CW**	441	310	478	1.597	582
	MW	320	406	624	1.198	308
	AW	199	158	1.641	1.033	308
Dermatology	**CW**	203	500	5.450	1.328	5.202
	MW	403	844	22.280	1.593	11.835
	AW	436	977	15.419	1.080	905
Breast	**CW**	2.172	2.299	3.073	3.511	2.566
	MW	1.080	1.200	1.778	2.618	2.279
	AW	1.498	2.559	1.778	2.300	2.857
Pima Indian	**CW**	11.102	11.308	9.056	31.030	9.186
	MW	29.373	12.965	24.221	20.152	47.488
	AW	27.819	13.657	32.680	22.307	24.880
Yeast	**CW**	256.673	176.034	235.994	2.256.105	191.309
	MW	126.232	129.176	148.378	93.020	216.264
	AW	85.612	107.341	102.820	111.252	78.225
Segmentation	**CW**	7.846	3.649	9.440	18.885	25.781
	MW	7.604	3.244	10.451	19.015	13.568
	AW	7.075	3.563	9.407	10.408	35.231
WaveForm	**CW**	29.654	30.384	29.078	40.892	69.866
	MW	30.324	30.452	32.804	45.208	58.766
	AW	32.604	31.057	30.121	47.519	54.334
PenDigits	**CW**	11.836	2.886	37.819	58.862	264.840
	MW	13.642	3.317	36.889	62.936	408.593
	AW	12.054	3.550	42.226	56.350	260.754
Letter	**CW**	211.796	46.703	412.184	414.054	860.152
	MW	229.286	49.679	421.049	420.015	859.849
	AW	229.762	46.813	425.354	428.885	928.457

Table A.7 Cost M7 Model Results of Algorithms

		LD	KNN	MLP	N. MEAN	C4.5
Diabetic	**CW**	6.540	6.179	6.540	12.594	6.359
	MW	6.540	6.831	7.374	9.151	15.527
	AW	7.665	10.165	11.651	7.665	8.860
Iris	**CW**	8.128	8.396	8.314	12.864	8.913
	MW	6.429	8.198	8.587	9.980	7.167
	AW	7.037	7.271	7.062	10.296	7.050
Dermatology	**CW**	29.193	33.527	114.662	37.648	68.774
	MW	18.727	22.488	69.330	27.125	23.069
	AW	21.430	26.638	79.946	24.930	23.746
Breast	**CW**	56.363	58.973	58.753	62.377	61.764
	MW	55.298	52.326	54.064	60.629	81.513
	AW	53.993	51.674	52.397	57.437	81.583
Pima Indian	**CW**	153.658	139.914	140.575	225.274	143.186
	MW	174.886	131.964	166.614	184.547	214.858
	AW	167.597	145.488	175.349	202.450	196.108
Yeast	**CW**	2.120.649	1.841.171	2.011.122	3.003.958	2.577.400
	MW	193.687	189.148	235.030	224.145	236.293
	AW	210.624	204.388	226.329	275.945	245.587
Segmentation	**CW**	205.013	164.291	211.316	281.213	374.689
	MW	204.892	166.825	214.751	268.092	300.560
	AW	209.101	177.202	208.455	356.946	687.640
WaveForm	**CW**	540.686	538.349	535.119	548.664	1.152.081
	MW	605.826	593.541	611.720	627.271	1.137.475
	AW	575.854	589.824	590.077	636.763	991.931
PenDigits	**CW**	530.487	425.012	685.243	962.140	2.747.844
	MW	512.780	417.437	817.451	955.437	2.742.035
	AW	583.032	489.104	837.632	987.435	2.698.903
Letter	**CW**	2.795.068	1.314.065	4.439.082	4.238.319	8.067.282
	MW	2.892.443	1.337.468	4.554.459	4.269.878	8.002.186
	AW	2.952.543	1.334.184	4.721.824	4.409.154	8.771.868

Table A.8 Cost M8 Model Results of Algorithms

		LD	KNN	MLP	N. MEAN	C4.5
Diabetic	**CW**	36	43	36	94	39
	MW	54	27	51	53	68
	AW	54	64	88	77	82
Iris	**CW**	47	36	56	121	50
	MW	35	42	50	85	27
	AW	19	20	35	78	30
Dermatology	**CW**	212	249	770	291	772
	MW	170	246	1.074	245	198
	AW	183	175	1.047	184	278
Breast	**CW**	529	557	521	548	581
	MW	541	503	533	581	774
	AW	595	515	530	549	867
Pima Indian	**CW**	1.362	1.178	1.286	1.395	1.313
	MW	1.376	1.063	1.497	1.390	1.557
	AW	1.332	1.144	1.262	1.438	1.225
Yeast	**CW**	25.812	22.364	26.858	36.044	32.242
	MW	2.424	2.485	2.589	2.842	3.201
	AW	2.653	2.925	2.936	3.069	4.627
Segmentation	**CW**	1.577	1.580	1.776	2.882	3.222
	MW	1.496	1.529	1.689	2.781	1.672
	AW	1.500	1.503	1.634	2.609	3.546
WaveForm	**CW**	3.741	3.696	3.767	5.247	7.888
	MW	3.901	3.718	4.044	4.957	9.559
	AW	3.843	3.710	3.605	4.803	9.095
PenDigits	**CW**	4.191	3.490	5.886	7.884	29.401
	MW	4.298	3.585	5.947	8.299	31.346
	AW	4.378	3.646	5.895	8.446	27.619
Letter	**CW**	29.797	13.702	46.591	49.219	95.847
	MW	31.182	13.865	48.071	49.623	88.845
	AW	30.569	13.951	46.303	49.396	99.221

References

[1] R.O. Duda, P.E. Hart, D.G. Stork*, Pattern Classification,* John Wiley&Sons, 2001

[2] E. Yen, W. M. Chu*, Relaxing instance boundaries for the search of splitting points of numerical attributes in classification trees,* Imperial College, London, UK, Pages 1276-1289, March 2007

[3] F. Esposito, D. Malerba, G. Semeraro, J. Kay. "*A comparative analysis of methods for pruning decision trees,"* Volume 19, Issue 5, pp. 476 – 491, May 1997

[4] E. Alpaydın, *Introduction to Machine Learning,* Cambridge, Maustachusetts London*,* England, 2004

[5] C. Elkan, "*The Foundations of Cost-Sensitive Learning,"* Department of Computer Science and Engineering 0114 University of California, San Diego La Jolla, California 92093-0114, pp. 973-978, 2001

[6] B. Zadrozny, J. Langford, N. Abe, "*Cost-sensitive learning by cost-proportionate example weighting, Data Mining,"* 2003. ICDM 2003. Third IEEE International Conference on Volume, Issue, pp. 435 – 442, 19-22 Nov. 2003

[7] P. Turney, "*Types of Cost in Inductive Concept Learning*," Institute for Information Technology," National Research Council of Canada, M-50 Montreal Road, Ottawa, Ontario, Canada, K1A 0R6, 2000

[8] D. D. Margineantu ," *Methods for Cost Sensitive Learning, degree of Doctor of Philosophy in Computer Science* ," September 21, 2001

[9] C. L. Blake, C. J. Merz, *"UCI Repository of Machine Learning Databases"*, 2000.
http://www.ics.uci.edu/~mlearn/databases/

Curriculum Vitae