

APPLYING DATA MINING FOR PREDICTING COMPUTER SCIENCE
STUDENTS ACADEMIC PERFORMANCE

A Thesis
presented to
the Faculty of Natural and Applied Sciences
at Notre Dame University-Louaize

In Partial Fulfillment
of the Requirements for the Degree
Master of Science

by
JAD ALYOUSSEF

MAY 2019

© COPYRIGHT

By

JAD ALYOUSSEF

2019

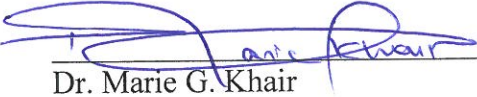
All Rights Reserved

Notre Dame University - Louaize
Faculty of Natural and Applied Science
Department of Computer Science

We hereby approve the thesis of

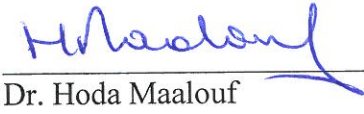
JAD ALYOUSSEF

Candidate for the degree of Masters in Computer Science



Dr. Marie G. Khair

Associate Professor, Supervisor,



Dr. Hoda Maalouf

Associate Professor, committee member



Dr. Nazir Hawi

Associate Professor, committee member

Acknowledgments

This thesis would not have been possible without the support I received from my supervisor Dr. Marie Khair, my family and friends, and especially from the faculty at Notre Dame University.

I would like to thank every person who took part in my thesis. It was quite motivating and strengthening to come across professionals who were so willing to take the time to help.

I am so grateful to my supervisor, Dr. Marie Khair, who believed in me and encouraged me to keep on working to achieve my goal despite, my heavy responsibilities at my present job. I am certain that I could not have completed my thesis without her continuous support, enlightenment and irreplaceable aid throughout the various stages of this research.

Above all, I would like to thank my parents, Jeries Al Youssef and Loudy Al Youssef, not only for furnishing the proper environment which granted me the possibility of working on my thesis, but also for their unconditional sacrifices which paved the way for me to accomplish my target.

Finally, I would like to express my gratitude to my friends for their constant presence and support throughout the stressful time of this process.

Thanks again to all of you! Without you, my thesis would be incomplete.

Table of Contents

| | |
|---|-----------|
| Acknowledgments | iv |
| Table of Contents | v |
| List of Figures..... | vi |
| List of Tables | vii |
| List of Abbreviations | ix |
| Abstract..... | x |
| Chapter 1: Introduction and Problem Definition | 1 |
| 1.1 Introduction to the General Problem..... | 1 |
| 1.2 Problem Definition..... | 1 |
| 1.3 Research Objectives | 2 |
| 1.4 Research Motivation | 2 |
| 1.5 Approach and Main Results | 4 |
| 1.6 Thesis Organization..... | 4 |
| Chapter 2: Data Mining | 5 |
| 2.1 Definition of Basic Concepts | 5 |
| 2.2 Neural Network and Multilayer Perceptron | 8 |
| 2.3 Decision Tree | 16 |
| 2.4 Relief Algorithm | 24 |
| 2.5 Data Mining Tools | 25 |
| Chapter 3: Educational Data Mining..... | 27 |
| 3.1 Introduction | 27 |
| 3.2 Research Areas..... | 27 |
| 3.3 Educational Data Mining | 28 |
| 3.4 Most effective Tasks | 33 |
| Chapter 4: Data Preparation and Preprocessing | 34 |
| 4.1 Study Design | 35 |
| 4.2 Data Collection..... | 36 |
| 4.3 Inclusion and Exclusion Criteria..... | 38 |
| 4.4 Data Pre-Processing | 39 |
| 4.5 Metrics..... | 42 |
| Chapter 5: Findings and Analysis | 44 |
| 5.1 Summary of the Main Results..... | 44 |
| Chapter 6: Conclusion and Future Work..... | 63 |
| 6.1 Conclusion..... | 63 |
| 6.2 Future Work | 64 |
| References | 65 |

List of Figures

| | |
|---|----|
| Figure 1 Data Mining Paradigms | 5 |
| Figure 2 Model of Simple Neuron | 11 |
| Figure 3 Artificial Neural Network Example | 13 |
| Figure 4 Decision Tree Example | 17 |
| Figure 5 Graph showing the proposed workflow, division of objectives and cases, and algorithms applied..... | 36 |
| Figure 6 Number of students in each major in the department of computer science..... | 37 |
| Figure 7 First and second semester grades to get final GPA for Business Computing Students . | 40 |
| Figure 8 First and second semester grades to get final GPA for Computer Science Students. | 41 |
| Figure 9 Courses which have the most effect on the final GPA for Business Computing Students. | 41 |
| Figure 10 Courses which have the most effect on the final GPA for Computer Science Students. | 42 |
| Figure 11 Decision Tree (01-BC-S3a) size of tree is 21 with 19 leaves..... | 46 |
| Figure 12 MLP (01-CS-S3b) 30 input perceptron, 2 output perceptron, and 3 hidden layers with 6 Perceptron each. | 48 |
| Figure 13 Decision Tree (01-BC-S3a) size of tree is 29 with 26 leaves..... | 51 |
| Figure 14 MLP (01-CS-S3b) 29 input perceptrons, 2 output perceptrons, and 3 hidden layers with 6 Perceptrons each. | 54 |
| Figure 15 Decision Tree (02-BC-S3a) size of tree is 10 with 7 leaves..... | 56 |
| Figure 16 MLP (02-BC-Sb) 6 input perceptron, 3 output perceptron, and 3 hidden layers with 6 Perceptron each. | 58 |
| Figure 17 Decision Tree (02-CS-S3a) size of tree is 7 with 5 leaves..... | 60 |
| Figure 18 MLP (02-CS-Sb) 6 input perceptron, 3 output perceptron, and 3 hidden layers with 6 Perceptron each. | 62 |

List of Tables

| | |
|--|----|
| Table 1 High Potential Attributes | 33 |
| Table 2 Objective 1 GPA before and after pre-processing | 40 |
| Table 3 Objective 2 GPA before and after pre-processing | 40 |
| Table 4 Courses evaluation according to Relief Attribute Evaluation (01-BC-S2)..... | 45 |
| Table 5 J48 Configuration (01-BC-S3a)..... | 45 |
| Table 6 J48 Results (01-BC-S3a) | 45 |
| Table 7 J48 Confusion Matrix (01-BC-S3a)..... | 47 |
| Table 8 MLP Configuration (01-BC-S3b)..... | 47 |
| Table 9 MLP Results (01-BC-S3b)..... | 47 |
| Table 10 MLP Confusion Matrix (01-BC-S3b)..... | 49 |
| Table 11 Courses evaluation according to Relief Attribute Evaluation (01-CS-S2)..... | 49 |
| Table 12 J48 Configuration (01-CS-S3a) | 50 |
| Table 13 J48 Results (01-CS-S3a)..... | 50 |
| Table 14 J48 Confusion Matrix (01-BC-S3a)..... | 52 |
| Table 15 MLP Configuration (01-CS-S3b) | 52 |
| Table 16 MLP Results (01-CS-S3b)..... | 53 |
| Table 17 MLP Confusion Matrix (01-CS-S3b) | 55 |
| Table 18 J48 Configuration (02-BC-Sa)..... | 55 |
| Table 19 J48 Results (02-BC-Sa) | 56 |
| Table 20 J48 Confusion Matrix (02-BC-Sa)..... | 57 |
| Table 21 MLP Configuration (02-BC-Sb)..... | 57 |
| Table 22 MLP Results (02-BC-Sb)..... | 58 |
| Table 23 MLP Confusion Matrix (02-BC-Sb)..... | 59 |
| Table 24 J48 Configuration (02-CS-Sa) | 59 |
| Table 25 J48 Results (02-CS-Sa)..... | 59 |
| Table 26 J48 Confusion Matrix (02-CS-Sa)..... | 60 |
| Table 27 MLP Configuration (02-CS-Sb) | 61 |
| Table 28 MLP Results (02-CS-Sb)..... | 61 |
| Table 29 MLP Confusion Matrix (02-CS-Sb) | 62 |

Table 30 List of results. 64

List of Abbreviations

| | |
|-------------|------------------------------|
| DM | Data Mining |
| EDM | Educational Data Mining |
| ITS | Intelligent Tutoring Systems |
| NN | Neural Network |
| ANN | Artificial Neural Network |
| SLP | Single Layer Perceptron |
| MLP | Multilayer Perceptron |
| DT | Decision Tree |
| TP | True Positive |
| FP | False Positive |
| FN | False Negative |
| ID3 | Iterative Dichotomiser 3 |
| RMSE | Root Mean Square Error |

Abstract

The aim of this thesis is to study the impact of a three-year-study of computer science at Notre Dame University on the students' academic performances. For this reason, the GPAs of the students after their first year of study were compared with their GPAs upon graduation. To perform this, a Decision Tree, as well as Neural Networks were used. These algorithms helped us to predict the final GPA based on the first two semesters' GPA. In addition, the most decisive courses on the student GPA were identified. For this reason, Relief algorithm was used to identify the three most important courses which affect the final GPA. This output helps instructors, managers, and even students to put more emphasis on these courses hoping for a better GPA. By using the F-measure, which is the weighted harmonic mean of the precision and the recall, the effectiveness of the algorithms is evaluated. The F-measure in this research for all the simulations performed ranged between 74.9% and 91.2%, which is statistically acceptable.

Keywords: Data Mining, Educational Data Mining, Neural Network, Decision Tree, Relief Attribute Evaluation

Chapter 1: Introduction and Problem Definition

1.1 Introduction to the General Problem

Universities thrive strongly to study their students' performances as well as the Students' educational patterns. The factors, which affect the performance of students, whether this performance is excellent or poor, are analyzed, surveyed, collected and understood based upon past and present information. However, the process is really complex and continuous. There are increasing research interests in using data mining methods that help discover knowledge from data originating from an educational background.

Educational Data Mining, EDM, is used to analyze and classify students' data which was provided by the University between the years 2000 and 2012. Moreover, the same data will be used to predict the performance of the current students in the coming semesters. The outcome obtained from this study will help students, as well as professors, to identify areas where improvement is possible and focuses on it for better results. Underperforming students might be early identified by the same data and as a result, their educators would welcome the idea of giving those underperformers the attention they deserve in order to enhance their performance at college.

1.2 Problem Definition

This thesis aims to study and analyze students' data by using neural network and decision tree algorithms to assist professors and educators to understand better students' performance and behavior by generating specific rules, classifications, and predictions.

1.3 Research Objectives

The purpose of this research entails two objectives. The first Objective is to determine which one of the selected courses have the greatest effect on the final Grade Point Average (GPA) per student. The second objective is to predict and verify the final GPA of the students using their total GPA in their first two semesters. Each of the previous steps has been divided into two cases: Computer Science concentration and Business Computing concentration. This is mainly performed due to the fact that the courses taken in the two majors are different.

1.4 Research Motivation

The main goal of EDM is to extract useful knowledge from educational data including student records, student usage data, and learning management systems. The extracted knowledge can improve the process of teaching and learning in the educational system. It can also lead to the development of new teaching processes. (Romero C. &, 2010)

EDM can be used to help students, teachers, administrators, and research stakeholders. For students, the aim of EDM is to make use of data to present appropriate tasks, learner activities, and resources aimed at optimizing student learning. For example, in real-time, adaptive and intelligent web-based educational systems use estimates of student ability and incorporate preferences and goals to present the most appropriate learning material.

When teachers are the focus, the objective is to obtain feedback on the content, delivery, and structure of learning. Such feedback may identify common misinterpretations and irregular patterns of learning and enable teaching staff to refine instructional methods. Examples here might include using feedback to determine optimal instructional sequences to support student learning.

When administrative staff is the focus, educational data might inform the optimization of learning management systems and servers, user interfaces, and contribute to an understanding of student attendance and retention. An example here may be to draw upon data in a university learning management system to create statistical models that predict student engagement and retention.

Furthermore, universities strive to get accredited from organizations such as the Accreditation Board for Engineering and Technology (ABET) and the New England Commission on Institutions of Higher Education (NECHE) which values and emphasizes on assessments and performance of students. Hence, the research and findings of the thesis will play as an additional advantage for the university to meet the accreditation requirements.

Moreover, the successful usage of educational data mining techniques will further be beneficial to the university allowing the management to be able to predict the grades scales of courses making it possible to prepare in advance for postgraduate programs. Also, using EDM techniques will pinpoint to professors' certain courses that students are not doing so well in. Impelling professors to search for reasons and allowing the chance to enhance and improve the course teaching methods if necessary.

This thesis aims to apply educational data mining techniques to study variables that affect the performance of students. This is done by first determining which of the courses have the greatest effect on the final GPA and second by predicting the student's final GPA relying on their performance in the first two semesters.

1.5 Approach and Main Results

Data mining provides various tasks that can be used to study the students' performance. In this thesis, the classification task is used to evaluate the student's performance by means of a decision tree method and neural network method. Data applied to predict the performance of the student at the end of the semester was collected from the University.

1.6 Thesis Organization

This thesis is divided into six chapters. The first chapter includes an introduction to the general problem, problem definition, research objectives, approach and main results, and thesis Organization.

The second chapter is a literature review on data mining techniques that present previous and recent studies in the field. As for the third chapter, it includes research areas in EDM as well as the latest research performed in this field, EDM Categories, and other effective Tasks. The fourth chapter explains the data used as well as the inclusion and exclusion criteria, the tools and the metrics used for result evaluation. The fifth chapter shows the findings analyzed after applying the different algorithms on the data. Chapter six is a summary of the main contributions of the thesis as well as possible extensions.

Chapter 2: Data Mining

2.1 Definition of Basic Concepts

Data Mining (DM) involves using a set of methods to obtain significant knowledge from data. Data mining has multiple usages, such as: clustering, classification, prediction, and association rule mining (Rahul B. Diwate, 2014). Moreover, the mostly used DM techniques applied are decision trees, neural networks, and Bayesian networks.

Classification techniques are extremely valuable techniques in data mining, which are to build classification models, which consequently, can be used to predict future data movements.

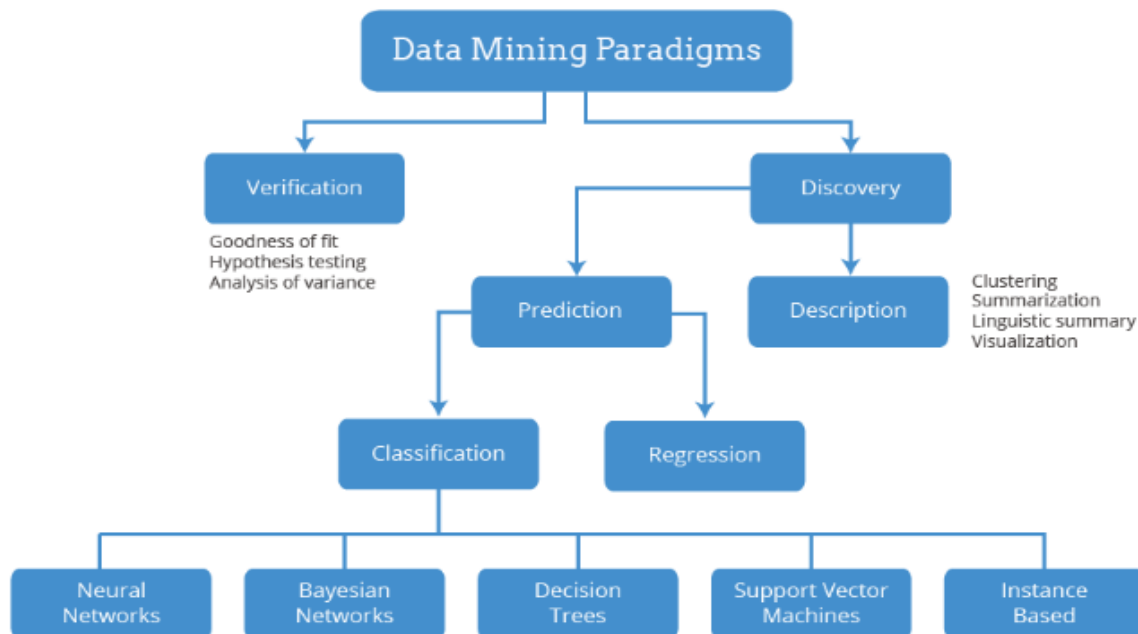


Figure 1 Data Mining Paradigms (Lior Rokach, Oded Maimon, 2014, pp. 8,9)

One of the most popular uses of data mining techniques is predictive modeling. To predict the results of new data in which the answers are unknown, one can use a predictive model using the old data available. There are two main types of predictive modeling: the classification model, and the regression model. There are several differences between classification and regression techniques. The most important difference is that classification is about predicting a label or a category while regression is about predicting numerical values.

2.1.1 Classification prediction modeling

Classification predictive modeling is the task of approximating a mapping function (f) from a set of input variables (X) to a set of discrete output variables (y), usually called labels or categories. The mapping function predicts the class or category for a given observation. The classification model is evaluated using detailed accuracy by a class which includes the percentage of correctly classified instances, true positive rate, false positive rate, precision, recall, and F-Measure for all predictions made.

2.1.2 Regression prediction modeling

Regression prediction modeling is the task of approximating a mapping function (f) from one or more input variables (x) to a continuous output variable (or variables) (y) which is a real value and often comes in the form of a quantity, such as amounts and sizes. The regression model is evaluated when using Root Mean Squared Error (RMSE) to measure the error rate of a regression model with the same units.

$$\text{RMSE} = \sqrt{\frac{\sum_t^n (\chi_{1,t} - \chi_{2,t})^2}{n}}$$

There could be an overlap between the above two techniques: the classification and the regression. The classification modeling may predict a continuous value in the form of a class label. The same applies for prediction modeling in which the algorithm is capable of predicting a discrete value in the form of quantity.

2.1.3 Clustering prediction mode

According to Berkhin, clustering is defined as “a division of data into groups of similar objects” (Berkhin, 2002). Instances within the same cluster are similar to each other but different from instances of other clusters. The clustering algorithm works better in case the data are regularized in a way that one attribute does not control the analysis.

Two of the most used types of clustering algorithms are the hierarchical and partitioning methods.

The hierarchical clustering is the method where the smallest clusters in the tree, bond together to create the next level of clusters. Each cluster holds one record. Hence, the root of the tree will eventually hold the cluster, which contains all records.

There are two types of hierarchical clustering:

a) The Agglomerative clustering algorithm is a tree which contains as many clusters as possible, with one record in each cluster. Then clusters that are the closest to each other, are joined together to create the next level of clusters. The process continues until there remains only one single cluster containing all records.

b) The divisive clustering algorithm begins with all the records in one cluster, and then the most appropriate cluster is split depending on how comparable the records are in each cluster. This process continues until some stopping criterion is encountered.

The Partitioning clustering consists of dividing the data set into different clusters, where each record belongs to only one cluster. The method begins by randomly choosing several clusters which are then optimized based on some validity measures such as K-means clustering, K-medoids clustering, and CLARA algorithm.

2.1.4 Association Rules

Association analysis is convenient in finding relationships hidden in large datasets. The importance of this analysis is to find associations for quantifying the relationship between two or more attributes (Larose, 2005). The strength of an association is measured by its support and confidence. The Support describes how many times an association is relevant to a certain data set. Confidence describes how often the association has been found to be true (Pang-Ning Tan, 2005).

2.2 Neural Network and Multilayer Perceptron

Dr. Robert Hecht-Nielsen, the inventor of one of the first neurocomputer, defines a neural network, or in other words, artificial neural network (ANN) as: "... a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs (Caudill, 1989)".

ANN is a field that investigates how simple models of biological brains can be used to solve difficult computational tasks, such as predictive modeling, which can be seen in machine learning.

The ANN can be used to approximate the relationship between input and output signals of a system (Rebizant W. et. AL.,2011). The goal is not to create realistic models of the brain, but instead to develop robust algorithms and data structures that can be used to model difficult problems.

The power of neural networks comes from their ability to learn the representation of the training data and to relate it to the output variable that needs to be predicted. In this sense, neural networks learn mapping functions. Mathematically, neural networks can learn any mapping function and have been proven to be a universal approximation algorithm (Brownlee, 2016).

The starting point of ANNs development was the invention of the perceptron, which was introduced by Rosenblatt in 1958. Multilayer Perceptron (MLP) algorithm is one of the most widely used and popular neural networks.

An artificial neural network is composed of perceptrons, connections and weights, and propagation functions.

The perceptron element is a single processing unit that receives weighted input and produces a rule-based threshold output. Linearly separable data can be classified by a Single Layer Perceptron (SLP) whereas non-linearly separable data can't (Atkinson, P., and Tatnall, A., 1997).

A perceptron is a single neuron model that is a predecessor to a larger neural network. Neurons are simple computational units that accept weighted inputs signals and produce an output signal using an activation function.

The perceptron is a mathematical model of a biological neuron; hence, in actual neurons, the dendrite receives electrical signals from the axons of other neurons; while in the perceptron, these electrical signals are represented as numerical values (Nielsen, 2015)

Thus, being the fundamental element of a neural network, multiple perceptrons are linked to each other by synapses and each synapse is defined by a synaptic weight. Perceptrons are positioned in layers that operate in parallel.

The first layer is the input layer which is responsible for receiving information from outside the network. In this layer, neurons do not execute any calculations. This first layer is followed by hidden layers. The activity of the input units, along with the weights at the connections of the input and hidden units, determine the activity of the hidden layers.

A neural network can include from zero to many hidden layers and their role is to enhance the network's performance. The existence of these layers at the network becomes more necessary as the number of input neurons grows.

The last layer is the output layer. This layer contains the network's predictions or classifications. The behavior of output units depends upon the activity of the hidden units and the weights between the hidden units and the output ones.

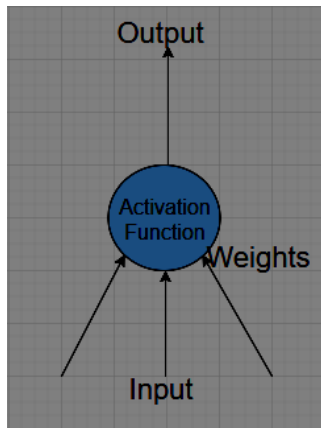


Figure 2 Model of Simple Neuron

The transfer function can be a log function, a sigmoid function, or a hyperbolic tangent, which scales all the information coming into the neuron. The sigmoid curve is often used as a transfer function because it introduces non-linearity into the network's calculations. The sigmoid function has the additional benefit of having an extremely simple derivative function, as required for back-propagation errors through a feed-forward neural network (Seggern, 2007).

As values are sent from one layer to the next, a weight is assigned to each interconnecting line and is multiplied by the values. Each neuron on the hidden layer sums all inputs, and the combined input is modified by the function. The output value of the transfer function is generally passed directly to all neurons in the next layer, again with a weight assigned to each value. Values of the interconnecting weights predetermine the neural network computation reaction to any arbitrary input pattern. As information is passed forward from the inputs towards the outputs, interconnecting weights are adjusted during the learning phase by a back-propagation algorithm, so that known outputs will best match predicted outputs.

It is a feed-forward, fully interconnected, supervised network. It builds models that classify patterns or make predictions according to patterns of inputs and outputs, which the network has learned.

Training of an NN is the procedure by which the values of the individual weights are determined in such a way that the relationship the network is modeling is accurately resolved. This is accomplished by varying the weights through all possible values, and by plotting the errors in the three-dimensional space.

The objective of training a neural network is to find the combination of weights, which will result in the smallest error. In practice, it is not possible to plot such a surface due to the multitude of weights and hence it is more relevant to find the minimum point of the error.

In neural networks, the process of obtaining the output and comparing it with the real value to get the error is identified as forwarding propagation. It is the first and simplest form of artificial neural network (Schmidhuber, 2015).

However, to minimize the error, the process of backward propagation is used by finding the derivative of the error with respect to each weight and then subtracting this value from the weight value.

Backpropagation is used in artificial neural networks to calculate a gradient that is needed in the calculation of the weights to be used in the network (Ian Goodfellow, 2016).

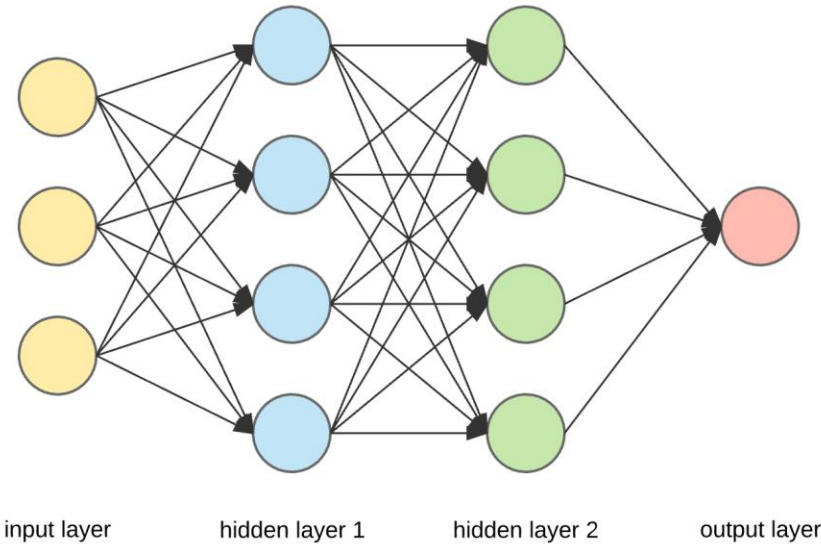


Figure 3 Artificial Neural Network Example

The most crucial step of training perceptron in the neural network is to be activated only for the precise type of inputs rather than all. Therefore, by propagating forward it would be possible to monitor how the neural network is behaving and hence locating the error. After locating the error, backpropagation is then used, and a form of gradient descent is used to update new values of weights. Then, again the process of forwarding propagation is repeated to see how well those weights are performing and then will backward propagate to update the weights. This process is repeated until the least value of error is achieved.

The computational effort needed for finding the correct combination of weights increases substantially when more parameters and more complicated topologies are considered.

According to Rumelhart et al., “initial weights of exactly zero cannot be used since symmetries in the environment are not sufficient to break symmetries in initial weights”.

Since their paper was published, it became a common practice to choose initial weights with a uniform distribution between plus and minus ρ , usually set to 0.5 or less. The reason behind avoiding the non-zeros is to break symmetry whereas the usage of small numbers is to avoid saturation (Rumelhart and McClelland, 1986).

Another recommended procedure is to choose the weights randomly and of different magnitude. In this way e.g., for the case of binary input vectors, one achieves an initial net input to each node which is somewhere in the interval $-1 < \text{net} < 1$ (John Hertz, Anders Krogh, Richard G. Palmer, 1991)

Another recommended procedure is to initialize the initial weights to small random values and then to assign learning rates to be inversely proportional to the average magnitude of vectors feeding into the node. The reason for starting from small weights is that large weights tend to prematurely saturate units in a network and render them insensitive to the learning process. (Don R. Hush, Bill G. Horne, 1993).

It is challenging to find the number of neurons and hidden layers within each layer of MLP. In general, there is no standard algorithm for calculating the proper number of hidden layers and neurons.

Even though, according to the universal approximation theorem (Cybenko, 1989) for any input-output mapping function in supervised learning, there exists an approximate correct MLP with a number of hidden neurons in the hidden layer.

However, the theorem does not indicate how to obtain the number of hidden layers. Therefore, a trial and error approach is generally used to find the number of hidden layers. The more neurons

there are in the hidden layer, the better the training error is, but the worse the generalization error is.

By using cross-validation methods, the optimal number of hidden layers could be found. If the model's performance ceases to improve sufficiently on the validation set, one could either stop learning (called early stopping), or modify some optimization parameters using different approaches such as learning rate, regularization, and momentum term.

In the Learning rate approach, the parameter measures the magnitude of the weight and updates to minimize the network's loss function. If the learning rate is set too low, training will progress very slowly while making tiny updates to the weights in the network. However, if the learning rate is set too high, it can cause unwanted divergent behavior in the loss function. Another commonly used technique is learning rate annealing, which recommends starting with a relatively high learning rate and then gradually lowering the learning rate during training.

A second approach is a regularization defined as “any modification we make to the learning algorithm that is intended to reduce the generalization error, but not its training error” (Ian Goodfellow, Yoshua Bengio, 2016). Regularization is a key component in preventing overfitting. There are many regularization strategies. Some put extra constraints on the machine learning model, such as adding restrictions on the parameter values while some add extra terms in the objective function that can be thought of as corresponding to a soft constraint on the parameter values (Gupta, 2017).

The third method is referred to as the momentum method (Polyak, 1964), which is a technique for accelerating gradient descent that accumulates a velocity vector in directions of persistent reduction in the objective across iterations.

2.3 Decision Tree

A decision tree is a map of possible outcomes of both sequential and hierarchal choice. A decision tree makes it possible to weigh prospect actions against one another based on different variables such as costs, benefits and probabilities. A decision tree can be applied to either classification or to regression tasks.

A decision tree consists of three main components:

1. Root node also called Decision node. It symbolizes a decision or a choice to be made. This node is demonstrated by a square node with two or more arcs.
2. Internal node also called Chance node. It symbolizes the probabilities of certain results. This node is demonstrated by a circle node with two or more arcs.
3. Leaf node also called End node or sometimes called “Terminal”. It symbolizes the final outcome associated with different paths of the decision tree.

The decision tree begins with a single node, the root, which branches into possible outcomes. Each of these outcomes leads to the creation of new nodes, hence, the development of new branches forming the shape of a tree. At each level of the tree, the data are split according to an attribute. After enough splits, only one remaining class is represented in the node. The branch represents the outcome whereas a node represents a class label. The root from node to branch represents

classification rules. Leaf nodes can either hold class labels (classification), continuous values (regression), non-linear models (regression), or even models produced by other machine learning algorithms (Rodrigo C. Barros, 2015).

Decision tree integrates both nominal and numeric attributes. Each node is labeled with the attribute it tests, and its branches are labeled with its corresponding values (Lior Rokach, Oded Maimon 2005).

Other important characterizations regarding decision trees are the concepts of depth and breadth. The average number of layers from the root node to the leaf nodes is referred to as the average depth of the tree. The average number of internal nodes in each level of the tree is referred to as the average breadth of the tree. The tree complexity is indicated by both breadth and depth; that is, the higher their values are, the more complex the corresponding decision tree is.

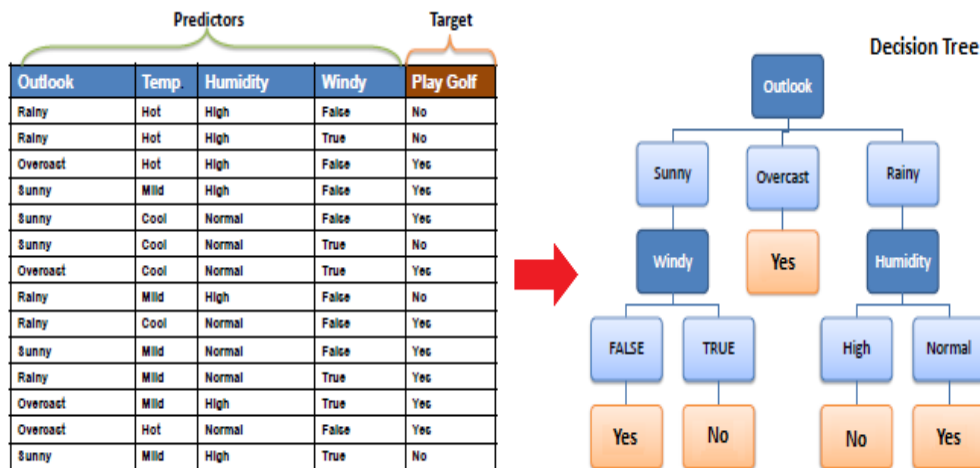


Figure 4 Decision Tree Example

There are many decision trees that can be built from the same data provided. It is considered a challenge to choose the optimal decision tree from the available data.

For example, Hyafil and Rivest have demonstrated that constructing a minimal binary tree with respect to the expected number of tests required for classifying an unseen object is an NP-complete problem (Laurent Hyafil, 1976). Hancock et al. have verified too that finding a minimal decision tree consistent with the training set is NP-hard. Also, the case of finding the minimal equivalent decision tree for a given decision tree and building the optimal decision tree from decision tables is set to be NP-hard (Hancock, 1996).

The above-mentioned theories indicate that growing optimal decision trees is only viable in precise problems. Hence, it was crucial to develop heuristics for solving the problem of growing decision trees. In the last three decades, several approaches were developed which can provide reasonably accurate, sub-optimal, decision trees in a reduced amount of time. Among these approaches, there is a clear preference to implement algorithms that rely on a greedy, top-down, recursive partitioning strategy for the growth of the tree (top-down induction).

2.3.1 Stopping Criteria

All decision trees require a stopping criterion to prevent growing a tree which is undesirable, difficult, and hard to interpret. Some (but not limited to) of the stopping criteria that trigger stopping the growing phase of the tree are:

- All instances in the training set belong to a single value of “Yes”
- The Number of cases in the node is less than the minimum number of cases for a node.
- Depth of the node is reached.

- Predictor values for all records are identical - in which no rule could be generated to split them.
- The best splitting criteria is not greater than a certain threshold.

2.3.2 Pruning Trees

One of the common problems that arise in decision tree algorithms is that the tree is either too large risking overfitting the training data, or too small causing under-fitting of the data. To solve this predicament, a pruning methodology can be used. It entails growing the tree to allow overfitting of the training data, then cutting the original tree into smaller ones by eliminating small branches that do not donate any additional valuable information. Pruning can happen in either a top-down manner, where it starts at the root to pass through nodes and slim subtrees, or the bottom up manner which starts at the leaf nodes.

2.3.2.1 Heuristic Pruning

Employing tight stopping criteria tends to create small and under-fitted decision trees. On the other hand, using loose stopping criteria tends to generate large decision trees that are over-fitted to the training set. To solve this dilemma the author in (Leo Breiman, 1984) developed a pruning methodology based on a loose stopping criterion and allowing the decision tree to over-fit the training set. Then the over-fitted tree is cut back into a smaller tree by removing sub-branches that are not contributing to the generalization accuracy. It has been shown in different studies that pruning methods can improve the generalization performance of a decision tree, especially in noisy domains.

2.3.2.2 Cost complexity pruning

Cost complexity pruning, also known as weakest link pruning, works in two stages. The first stage requires replacing a subtree with a leaf node to create a new tree. This process is repeated until gaining a sequence of subtrees ending with a tree with a single leaf node. In the second phase, the optimal tree is then selected. If the dataset is large enough, then the dataset is broken down into a training set and a pruning set. If the dataset is not large enough, then cross-validation methodology is proposed.

2.3.2.3 Reduced error pruning

Another procedure which is known as reduced error pruning entails checking all internal nodes from a bottom to a top manner. This is done for the purpose of checking the possibility of replacing these internal nodes with the most frequent class without reducing the accuracy of the tree. If the accuracy is not reduced, hence the node is pruned. This is continued until no further pruning is possible. This results with the smallest accurate sub-tree with respect to a given pruning set.

There are many other pruning techniques proposed other than the ones discussed above, and there exist many studies that compare the performance of each technique. These studies conclude that some methods such as Cost-complexity Pruning, and Reduced Error Pruning tend to over-pruning; “creating small but less accurate decision trees”. While other techniques such as errors conclude, there is no one pruning method that overtakes other pruning methods.

2.3.3 Decision Tree induction algorithm

2.3.3.1 ID3 Algorithm

ID3 (Iterative Dichotomiser 3) is a decision tree algorithm first introduced in 1980 by a machine researcher named J. Ross Quinlan. This algorithm implements a top-down, greedy search using certain sets to test every single attribute at every tree node. This process continues until the tree perfectly classifies the training examples, or until all attributes have been used.

This is done by presenting a question of: “Which attribute should be tested at the root of the tree?”. To solve this question, each possible attribute is evaluated using a statistical test to find the most useful possible attribute to classify a given set. The statistical test is defined as information gain function that best identifies the appropriate splitting. Information gain is simply defined as “the expected reduction in entropy caused by partitioning the examples according to this attribute”, (Shannon, 1948). To define information, gain accurately, entropy, which is a measure used in information theory, should be first defined. “Entropy characterizes the impurity of an arbitrary collection of examples” (Mitchell, 1997).

Using entropy, it is possible now to define a measure of the efficiency and success of a given attribute in classifying the training records. Using the measure of information gain, a process of selecting a new attribute and separating the training instances is implemented repeatedly for each non-terminal successor node, while using the training instances related to that specific node. Hence, combined attributes higher in the tree are omitted to give the possibility for any given attribute to appear at least once along any path through the tree. This process only stops when one of the following conditions arises: Either every attribute has once been included along the same

path through the tree, or all the training instances associated with the leaf node share a common attribute value.

ID3 algorithm handles only categorical values and is harder to implement on continuous data. As continuous data generates more splitting options of the attribute, hence it will be time-consuming searching for the best value.

Also, the ID3 algorithm can lead to overfitting of training data. Although this algorithm generates small decision trees, it does not necessarily generate the smallest possible tree which causes overfitting.

Most importantly, the ID3 algorithm does not guarantee the best solution possible, as it can be trapped in local optima. One of the possible enhancements that can be made on the algorithm is the use of backtracking while searching for the best decision.

2.3.3.2 C4.5 Algorithm

Following the ID3 algorithm, Quinlan later presented C4.5, the successor and more improved version of the ID3 algorithm. The C4.5 algorithm accepts both distinct and continuous features, handles partial data and also solves overfitting issues by using a technique known as pruning.

To decrease the prediction error rate, a pruning technique is applied in machine learning that works on decreasing the size of a decision tree by removing parts of the tree that are not considered powerful to classify instances.

The pruning's second purpose is reducing the complexity of the last classifier along with better accuracy in prediction, which is completed by the decrease of over-fitting and exclusion sections built on noisy or erroneous data.

2.3.3.3 J48 Algorithm

Decision Tree J48 is the java implementation of the C4.5 algorithm in data mining (Baker, 2010). J48 algorithm is an extension of the ID3 algorithm. It uses a divide and conquer approach to growing decision trees that was led by Hunt and his co-workers (Neeraj Bhargava, 2013). J48 algorithm includes additional features such as accounting for missing values, decision trees pruning, continuous attribute value ranges, derivation of rules and many more.

To maximize the gain, this process uses Entropy to divide the gain by the total entropy.

Pruning, whose importance is valuable to the results due to the existence of outliers, is the next step. Some instances in the data set are not very clear and vary from the other instances on its zone. The classification is therefore executed on the instances of the training set and hence, the tree is designed. The purpose of applying pruning is to reduce classification errors which are being formed in the training set by specialization.

In other algorithms the classification is performed recursively till every single leaf is pure, that is the classification of the data should be as perfect as possible. The objective is progressively generalization of a decision tree until it gains equilibrium of flexibility and accuracy.

2.3.3.4 CART Algorithm

CART stands for Classification and Regression Trees. This Algorithm was popularized by Breiman et al. (Leo Breiman, 1984). CART can handle both numeric and categorical variables

and it can easily handle outliers. It is characterized by the fact that it grows a large tree and then prunes the tree by choosing a split between all possible splits at each node using Twoing criteria resulting in obtaining the purest node through implementing cost-complexity pruning.

CART is also capable of producing regression trees, which make it possible to predict a real number rather than a class. However, some disadvantages of CART decision trees are the limitations of splitting only one variable, and the trees formed can be unsteady.

2.4 Relief Algorithm

In machine learning, algorithms are affected by noisy data. Hence, in order to avoid redundant complexity, the noise should be reduced as much as possible for the purpose of improving the efficiency of the algorithm. One of the possible ways to accomplish this is by feature selection. Feature selection is also known as the variable selection in the process for removing all redundant and irrelevant data from the original databank to end up with only the relevant features.

Relief is an algorithm developed by Kira and Rendel (Kenji Kira, 1992) which is capable of efficiently assessing the quality attribute. Relief algorithm is equipped with resolving classification complications with vastly dependent attributes, such as parity problems. This is done by testing a sample instance repeatedly while considering the worth of the specified attribute for the closest instance of a similar and completely different category.

Nevertheless, Relief algorithm still suffers from several weaknesses, some of which are: sensitivity to noise, incapability of dealing with incomplete data, regression difficulties, and multiclass. However, researchers have introduced an extended version of Relief, named ReliefF, which has

improved the algorithm in multiple areas. Relief is not limited only to two class problems but is capable of dealing with noisy and incomplete data, can also be executed for multiclass problems, and its regression variant relief can deal with regression problems (Igor Kononenko).

Although Relief algorithms have been mostly used during the preprocessing step as a feature subset selection method, Relief algorithms are also general feature estimators and have been applied successfully in various situations such as: choose splits in the building phase of decision tree learning (Kononenko et al., 1997), choose splits and guide the constructive induction in learning of the regression trees (Robnik Sikonja and Kononenko, 1997), an attribute weighting method (Wettschereck et al., 1997), and also in inductive logic programming (Pompe and Kononenko, 1995).

2.5 Data Mining Tools

There are plenty of tools available for data mining tasks. Some are RapidMiner, Weka, R-Programming, and Orange. For this study, the WEKA program has been used.

2.5.1 WEKA

WEKA (Eibe Frank, 2005) which stands for “Waikato Environment for Knowledge Analysis”, developed at Waikato University, is an assembly of machine learning algorithms for performing data mining tasks. Weka is an open-source software issued under the GNU General Public License. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes. Weka was used in this research because it is one of the best tools in the market and because it is an open-source software.

According to an article by Shravan, “Weka has proved to be an ideal choice for educational and research purposes, as well as for rapid prototyping” (SHRAVAN, 2017).

Chapter 3: Educational Data Mining

3.1 Introduction

Romero defined Educational Data Mining EDM as “an emerging interdisciplinary research area that deals with the development of methods to explore data originating in an educational context” (Romero C. , 2010). And as Ventura mentioned, EDM uses computational approaches to analyze educational data to study educational questions. (Ventura, 2010)

So, (EDM) is consisted of applying data mining techniques on the educational data to resolve some educational research issues. It is mainly concerned with developing methods to explore the unique characteristics of data in the educational sector to understand the behavior of students in the educational sector and be able to predict on their future behavior (Tiffany Barnes, 2009). For this reason, the EDM needs to convert raw data acquired in the educational system into useful knowledge which could be used by the stakeholders in the educational sectors during their decisions.

3.2 Research Areas

Educational Data Mining (EDM) has been developed and used as a research area for many diverse researchers from all over the world in various context, some of which are stated below:

1. Offline education focuses on gaining knowledge and skills from studying through human interaction between the professor and students.
2. E-learning and learning management system (LMS).

E-learning provides online instruction, and LMS provides communication, collaboration, administration, and reporting tools. Web mining (WM) techniques have been applied to student's data stored by these systems in log files and databases in order to predict students' performance.

3. Intelligent tutoring system (ITS) and adaptive educational hypermedia system (AEHS) is intended to adapt teaching to the needs of each student rather than a general approach of "one fits all" DM has been applied to data picked up by these systems, such as log files and user models.

3.3 Educational Data Mining

There are some important issues that differentiate the application of DM, specifically to education, from how it is applied in other domains. Below is an explanation of some of the most important issues.

1. Objective: DM's objective differs from one application area to another. In EDM, there are two main objectives. There is the applied research objective which for example focuses on enhancing the learning process for students and guiding students' learning path, while the other objective is pure research objective, which focuses on comprehending a deeper understanding of the educational system. These goals are not easy to quantify and use their own specified set of measurement techniques.
2. Data: In educational environments, there exists various types of data available for mining. These data are specifically related to the educational area and hence contain basic information, relationships with other data, and multiple levels of hierarchies. Some examples are the

domain model, used in ITS (Intelligent tutoring system) and AEHS (adaptive educational hypermedia system), which represents the relationships among the concepts of a specific subject in a graph or hierarchy format and the Q-matrix that shows relationships between items/questions of a test/quiz system and the concepts evaluated by the test. Also, it is essential to take educational aspects of the learner and the system into account.

3. Techniques: Educational data contain special characteristics that request the issue of mining to be treated in an alternative way. Even though most of the traditional DM techniques can be applied directly, others cannot, and have to be adapted to the specific educational problem at hand.

EDM involves different groups of users or participants. Different groups look at educational information from different angles, according to their own mission, vision, and objectives for using DM (Hanna, 2004). To enumerate some of the Groups, we can say, Teachers, Students, Instructors, Educators, Learners, Pupils, Tutors, Course Developers, Educational Researchers, Organizations and Universities.

Baker in his paper (Ryan Baker, 2009) suggests four key areas of application for EDM: Improving student models, improving domain models, studying the pedagogical support provided by learning software, and scientific research into learning and learners. In addition, he suggests five approaches/methods be applied: prediction, clustering, relationship mining, a distillation of data for human judgment and discovery with models.

Categories Analysis and Visualization of Data.

The objective of the analysis and visualization of data is to pinpoint valuable information and enhance decision making. For example, in the educational environment, it assists the educators to examine the students' course activities and uses obtained information to gain a closer view on students' learning curve. Statistics and visualization information are two essential techniques that have been used specifically for this task.

Providing Feedback for Supporting Instructors.

One of the objectives of EDM is providing feedback to authors/teachers/administrators in order to improve their decision-making process. Receiving feedback helps teachers pinpoint weaknesses and strengths which would enable them to take appropriate proactive and/or remedial action such as to improve students' learning, organize instructional resources more efficiently and much more. It is important to differentiate between this task and data analyzing and visualizing tasks, which only provide basic information directly from data (reports, statistics, etc.).

Association-rule mining has been the most common technique used for this task. Association-rule mining is capable of forming strong rules by finding interesting relationships between different variables available in a large database.

Recommendations for Students.

Another EDM objective is to be able to make recommendations directly to the students with respect to their personalized activities, links to visits, the next task or problem to be done, etc., and to be able to adapt learning contents, interfaces, and sequences to each particular student. Several

DM techniques have been used for this task, but the most common are association-rule mining, clustering, and sequential pattern mining. Sequence/sequential pattern mining aims to discover the relationships between occurrences of sequential events to find if there exists any specific order in the occurrences.

Predicting Student's Performance.

The objective of prediction is to estimate the value of a variable that describes the student. In education, the values normally predicted are performance, knowledge, score, or mark. This value can be numerical/continuous value (regression task) or categorical/discrete value (classification task). Regression analysis finds the relationship between a dependent variable and one or more independent variables. Prediction of a student's performance is one of the oldest and most popular applications of DM in education, and different techniques and models have been applied (neural networks, Bayesian networks, rule-based systems, regression, and correlation analysis).

Student Modeling.

The objective of student modeling is to develop cognitive models of human users/students, including modeling of their skills and declarative knowledge. EDM has been applied to automatically consider user characteristics (motivation, satisfaction, learning styles, affective status, etc.) and learning behavior in order to automate the construction of student models. Different EDM techniques and algorithms have been used for this task (mainly, Bayesian networks).

Several DM algorithms (naive Bayes, Bayes net, support vector machines, logistic regression, and decision trees) have been compared to detect student mental models in intelligent tutoring systems (ITS) (Rus, 2009).

Detecting Undesirable Student Behaviors.

One objective is to detect those students who portrait unusual behavior such as low motivation, cheating, dropping out, and academic failure. Several EDM techniques, mainly classification, and clustering have been used to indicate these types of students in order to provide them with the required help. Several of the classification algorithms that have been used to detect problematic student's behavior are the decision tree neural networks, naive Bayes, instance-based learning, logistic regression, and support vector machines for predicting/preventing student drop out.

Grouping Students.

The objective of clustering is to divide students into groups based on different traits such as personal characteristics and specific features. Then, a personalized learning system is built by the developer based on the obtained groups to encourage effective group learning and to provide adaptive contents. The EDM techniques used in this task are classification (supervised learning) and clustering (unsupervised learning). Classification is the process of predicting the class of given data points. Classes are sometimes called as targets/labels or categories. Clustering is the assignment of a set of observations into subsets called clusters where the same cluster contain similar observations.

3.4 Most effective Tasks

Predictive modeling is usually used for predicting student performance. There are several tasks used, which are classification, regression and categorization. The most popular task to predict student performance is classification. The most used attributes in EDM are listed below in table 1 with its probability in predicting students' performance.

| Variable | Description | Probability |
|----------|--|-------------|
| GSS | Students grade in Senior Secondary education | 0.8642 |
| LLoc | Living Location | 0.7862 |
| Med | Medium of Teaching | 0.7225 |
| MQual | Mother's Qualification | 0.6788 |
| SOH | Students other havit | 0.6653 |
| FAIn | Family annual income status | 0.5672 |
| FStat | Students family status | 0.5225 |

Table 1 High Potential Attributes (Ventura, 2010)

Chapter 4: Data Preparation and Preprocessing

This chapter gives an outline of the data used in our study. The chapter also describes the research design that was chosen for the purpose of this study and the reasons for this choice. The instrument that was used for data collection is also described, and the procedures that were followed to carry out this study are included, along with the methods used to analyze the data.

The aim of this study is to predict the final GPA of the students by using the cumulative grades of the first semester. Accordingly, the two BS majors, Business Computing (BC) and Computer Science (CS), have been chosen because of the easy accessibility of the information provided by the University.

The mission of both majors as stated by the University is to “educate students in the principles and practices of the business computer and computer science. These programs prepare students for careers and graduate studies in Business or IT related fields” (<http://www.ndu.edu.lb>).

So, the objectives of the computer science educational program can be as follows: enable students to work effectively as leaders or members of teams involved in the design and development of computer and software systems; have successful professional careers in computer science and related fields; apply scientific and engineering methodologies for analysis and resolution of problems; pursue advanced study and conduct research in computer science and related fields.

While, the objectives of the business computing educational program are: to prepare graduates with the knowledge and skills necessary to be effective professional contributors or leaders in the design, administration and management of information technology systems and databases; to

prepare graduates for professional careers in roles including, but not limited to: project managers, systems analysts, applications developers, webmasters, database administration and quality assurance; to provide graduates with the communication and interpersonal skills to become effective team-oriented problem solvers as well as effective communicators with non-technical stakeholders.

Both the business computing program and the computer science program include a total of 43 courses, 37 of which are divided between major and core requirements.

4.1 Study Design

The purpose of this research entails two objectives. The first objective is to determine which one of the selected courses have the greatest effect on the final GPA per student. The second objective is to predict and verify the final GPA of the students using their total GPA in their first two semesters. Each of the previous steps has been divided into two cases: Computer Science concentration and Business Computing concentration.

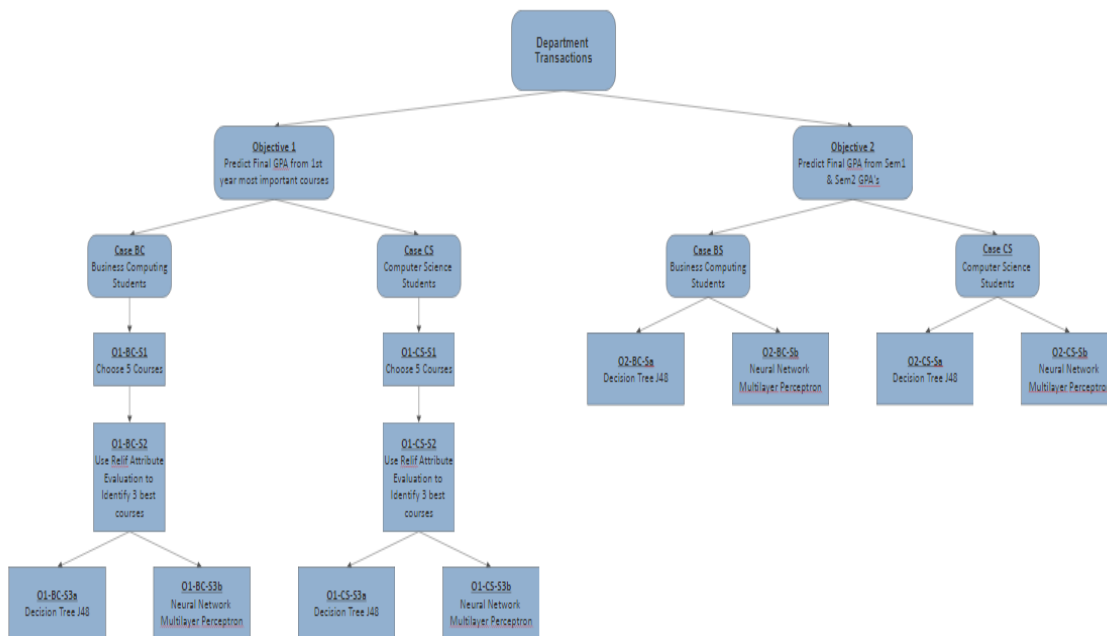


Figure 5 Graph showing the proposed workflow, division of objectives and cases, and algorithms applied.

Figure 5 explains the workflow of this thesis, starting from the division of the objectives and move forward to the courses selected and the algorithm applied. To be able to refer to each process and step, a naming convention was used as shown in the figure above.

4.2 Data Collection

The Data was collected for all students enrolled in the Computer Science Department between the years 2000 and 2010. Data were received in one excel workbook with multiple sheets, where student IDs were masked for privacy. The data gathered were originally in four sheets with 50 different attributes. The data set included attributes related to students' pre-admission information and course grades taken by these students.

The excel workbook held data for 769 students who were enrolled in six different majors in the Computer Science Department. They were divided as such: 331 Business Computing Students,

235 Computer Science Students, 67 Computer Graphics and Animation Students, 101 Computer Information System Students, 3 Information Technology Students, and 32 Geographical Information System Students as shown in figure 6 below.

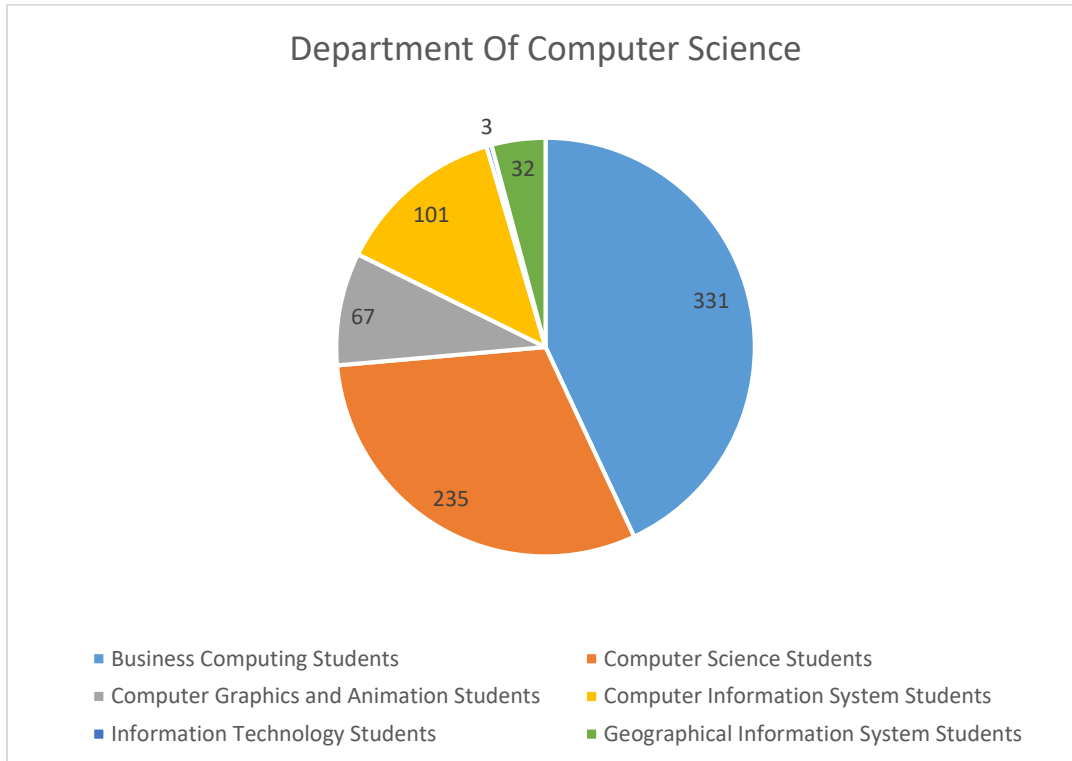


Figure 6 Number of students in each major in the department of computer science

Each sheet of the workbook is created as a database table. This could help in handling the data more efficiently and effectively to meet the requirements of the study.

For the first objective, the selected attributes are ID, Course Number, Grade, Cumulative GPA and Major, for both Business Computing Concentration (A) and Computer Science Concentration (B).

As for the second objective, the other 5 attributes were selected. The selected attributes are ID, Year, Semester, Cumulative GPA, Term GPA, for both Business Computing Concentration (A) and Computer Science Concentration (B).

4.3 Inclusion and Exclusion Criteria

The 2 majors selected, Business Computing and Computer Science, are chosen because they hold the largest number of students within our collected data. This is also due to the fact that data mining algorithms focus on learning from datasets with the maximum number of instances (Qiong Gu, 2008).

In the evaluation of objective 1 (courses with the greatest effect on final GPA), 5 courses were selected for Computer Science Students: CSC 201, CSC212, CSC213, MAT211, MAT215. As for Business Computing Students, 5 courses were selected: CSC 216, CSC217, CSC226, ACO201, and ACO202.

This selection is highlighted by the fact that these courses are among the most relevant to their subsequent majors and are taken during the first year of college.

In the evaluation of objective 2, (Prediction of Final GPA using first and second semester GPA), the GPA scores of semesters 1 and 2, in particular, are selected due to the necessity of early prediction, throughout the first year of college

In regard to the exclusion criteria, it served as data cleansing purposes. Data Cleansing is an important step in discovering the data, eliminating irrelevant items, filling missing values and removing inaccurate records.

The exclusion criteria include:

Change in Major: During the learning process, some students change their majors for different reasons. Hence, the grades are considered irrelevant by the system.

Drop Outs: Students who drop out of university or the major cannot be taken into consideration because their cumulative GPA does not correctly reflect their academic achievement.

Outliers: Students starting with a GPA lower than 1.7, were eliminated to achieve more accurate results.

After the elimination process, 256 Business Computing student records and 150 computer science student records are left for objective 1.

Regarding objective 2, 329 Business computing student records and 233 computer science student records are left. The difference in the total population number for objective 1 and 2 is due to the different selected attributes for either group.

4.4 Data Pre-Processing

Data pre-processing are an important step in the data mining process. Data pre-processing include cleaning, normalization, and transformation. The formula in the table below was used for objective 1 to convert the final GPA to High/Low value.

| Before Pre-Processing | After Pre-Processing |
|-----------------------|----------------------|
| 2-2.69 | Low |
| 2.7-4 | High |

Table 2 Objective 1 GPA before and after pre-processing.

The formula in table 3 was used for objective 2 to convert the final GPA from number to letter grade. This formula is relative to the grading system used in the university.

| Before Pre-Processing | After Pre-Processing |
|-----------------------|----------------------|
| 1.7 – 2.69 | C |
| 2.7-3.69 | B |
| 3.7-4 | A |

Table 3 Objective 2 GPA before and after pre-processing.

The next step would be to build database queries that select the required data for each objective and present it in a way that can be used in the WEKA data mining techniques. The figures below show 4 different queries; one for each section of the objective.

```

SQL Output Statistics
CREATE TABLE FirstSecndSemGradeToGPAuponGrad AS WITH ORDERED AS
(SELECT Num_To_Letr_Grade((SELECT DISTINCT Cumulative_Gpa
                           FROM Th_Course_Grades_Final_Gpa t
                           WHERE t.id = tt.id)) Cumulative_Gpa,
       Id,
       Year,
       Semester,
       Num_To_Letr_Grade(Term_Gpa) Term_Gpa,
       ROW_NUMBER() over(PARTITION BY id order by Year, DECODE(semester, 'Fall', 1, 'Spring', 2, 'Summer', 3)) AS rn
FROM th_per_semester_gpa tt
WHERE tt.term_gpa >= 1.7
      AND major = 'BS IN BUSINESS COMPUTING')
SELECT id, year, semester, term_gpa, cumulative_gpa
FROM ordered
WHERE rn <= 2;

SELECT LISTAGG(term_gpa, ':') within group(order by id, year, semester) Sem1,
       Cumulative_GPA as Final_GPA
FROM FirstSecSemGradeToGPAuponGrad
WHERE cumulative_gpa not in ('F', 'D')
GROUP BY id, Cumulative_GPA;

```

Figure 7 First and second semester grades to get final GPA for Business Computing Students

```

SQL | Output | Statistics
CREATE TABLE FirstSecndSemGradeToGPAuponGrad AS WITH ORDERED AS
(SELECT Num_To_Letr_Grade((SELECT DISTINCT Cumulative_Gpa
                           FROM Th_Course_Grades_Final_Gpa t
                           WHERE t.id = tt.id)) Cumulative_Gpa,
      Id,
      Year,
      Semester,
      Num_To_Letr_Grade(Term_Gpa) Term_Gpa,
      ROW_NUMBER() over(PARTITION BY id order by Year, DECODE(semester, 'Fall', 1, 'Spring', 2, 'Summer', 3)) AS rn
FROM th_per_semester_gpa tt
WHERE tt.term_gpa >= 1.7
      AND major = 'BS IN COMPUTER SCIENCE')
SELECT id, year, semester, term_gpa, cumulative_gpa
FROM ordered
WHERE rn <= 2;

SELECT LISTAGG(term_gpa, ';') within group(order by id, year, semester) Sem1,
      Cumulative_GPA as Final_GPA
FROM FirstSecSemGradeToGPAuponGrad
WHERE cumulative_gpa not in ('F', 'D')
GROUP BY id, Cumulative_GPA;

```

Figure 8 First and second semester grades to get final GPA for Computer Science Students.

```

SQL | Output | Statistics
CREATE TABLE courseToFinalGrade AS
SELECT id,
      DECODE(course_nb, 'CSC 201', 'A', 'CSC 216', 'B', 'CSC 217', 'C', 'ACO 201', 'D', 'ACO 202', 'E', 'CSC 226', 'F') Course,
      Grade,
      cumulative_gpa,
      Num_To_High_Low (cumulative_gpa) cumulative_gpa_letter
FROM (SELECT *
FROM TH_COURSE_GRADES_FINAL_GPA t
WHERE course_nb in ('CSC 201', 'CSC 216', 'CSC 217', 'ACO 201', 'ACO 202', 'CSC 226')
AND major in ('BS IN BUSINESS COMPUTING')
AND exists (SELECT 1
            FROM TH_COURSE_GRADES_FINAL_GPA th
            WHERE course_nb = ('CSC 216')
            AND th.ID = t.ID)
AND exists (SELECT 1
            FROM TH_COURSE_GRADES_FINAL_GPA thz
            WHERE course_nb = ('CSC 217')
            AND thz.ID = t.ID)
AND exists (SELECT 1
            FROM TH_COURSE_GRADES_FINAL_GPA thy
            WHERE course_nb = ('CSC 201')
            AND thy.ID = t.ID)
AND exists (SELECT 1
            FROM TH_COURSE_GRADES_FINAL_GPA th
            WHERE course_nb = ('ACO 202')
            AND th.ID = t.ID)
AND exists (SELECT 1
            FROM TH_COURSE_GRADES_FINAL_GPA thz
            WHERE course_nb = ('ACO 201')
            AND thz.ID = t.ID)
AND exists (SELECT 1
            FROM TH_COURSE_GRADES_FINAL_GPA thz
            WHERE course_nb = ('CSC 226')
            AND thz.ID = t.ID)
) order by major, id, course_nb;

SELECT id, LISTAGG (grade, ';') within group (order by id, cumulative_gpa_letter, course), cumulative_gpa_letter
FROM courseToFinalGrade
WHERE cumulative_gpa >2
GROUP BY id, cumulative_gpa_letter
ORDER BY cumulative_gpa_letter;

```

Figure 9 Courses which have the most effect on the final GPA for Business Computing Students.

```

SQL Output Statistics
CREATE TABLE CourseToFinalGrade AS
SELECT id,
DECODE(course_nb,'CSC 201','First','CSC 212','Second','CSC 213','Third') Course,
Grade,
cumulative_gpa,
Num_To_High_Low (cumulative_gpa) cumulative_gpa_letter
FROM (
SELECT *
FROM TH_COURSE_GRADES_FINAL_GPA t
WHERE course_nb in ('CSC 201','CSC 212','CSC 213')
AND major in ('BS IN COMPUTER SCIENCE')
AND exists (SELECT 1
FROM TH_COURSE_GRADES_FINAL_GPA th
WHERE course_nb = ('CSC 212')
AND th.ID = t.ID)
AND exists (SELECT 1
FROM TH_COURSE_GRADES_FINAL_GPA thz
WHERE course_nb = ('CSC 213')
AND thz.ID = t.ID)
AND exists (SELECT 1
FROM TH_COURSE_GRADES_FINAL_GPA thy
WHERE course_nb = ('CSC 201')
AND thy.ID = t.ID))
ORDER BY major, id, course_nb;
SELECT id,
LISTAGG (grade,':') WITHIN GROUP (ORDER BY id, cumulative_gpa_letter, course),
cumulative_gpa_letter
FROM courseToFinalGrade
WHERE cumulative_gpa >2
GROUP BY id, cumulative_gpa_letter
ORDER BY cumulative_gpa_letter;

```

Figure 10 Courses which have the most effect on the final GPA for Computer Science Students.

4.4.1 Tools Used

Oracle Database is used to: store the data, perform data cleansing, select attributes, discretize attributes, and generate files compatible with the format used in data mining tools. WEKA 3.6 is used to apply EDM techniques and algorithms.

4.5 Metrics

To measure the effectiveness of the EDM techniques used in this research, the F-Measure was used (Han, 2011). The F-measure is generally used in domains such as information retrieval and machine learning (Olson, 2008). The F-measure (eq1) of a system is described as the weighted harmonic mean of its precision (eq2) and recall (eq3).

Equation 1

$$FMeasure = 2x \frac{Precision \times Recall}{Precision + Recall}$$

Equation 2

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

Equation 3

$$Recall = \frac{TP}{FN + TP}$$

True Positive (TP) is the number of positive instances correctly classified as positive.

False Positive (FP) is the number of negative instances incorrectly classified as positive.

False Negative (FN) is the number of positive instances incorrectly classified as negative.

Chapter 5: Findings and Analysis

5.1 Summary of the Main Results

This research shows the two objectives, each with two cases. The results of each case will be presented separately, and then a comparison will be held to show the difference in the performance of the algorithms.

Objective1: Predict Final GPA from 1st year's most important courses. Objective 1 was divided into two cases because every major has a different course set and hence different courses to predict from. Business Computing Students is referred to as Case BC and Computer Science students as Case CS.

The courses selected for this process must be either major courses or core courses, taken by the students in their early semesters. According to the knowledge and experience acquired in this field, 5 courses were selected for each major, to begin with.

O1-BC-S1: For business computing students, the 5 courses chosen were ACO 202, ACO 201, CSC 226, CSC 217, CSC 216 and CSC 201.

O1-BC-S2: ReliefAttributeEval is the implementation of Relief Algorithm in WEKA. It takes a list of attributes as an input and gives the importance of each attribute in that list. In this step ReliefAttributeEval was used to choose the best 3 out of 5 courses. According to ReliefFAttributeEval the courses are ranked as follows:

| Course Number | Influence Factor |
|---------------|------------------|
| ACO202 | 0.2168 |
| ACO201 | 0.1789 |
| CSC226 | 0.1137 |
| CSC217 | 0.0785 |
| CSC216 | 0.0578 |

Table 4 Courses evaluation according to Relief Attribute Evaluation (01-BC-S2)

O1-BC-S3a: Decision Tree J48 was used to predict the final GPA for each student from the grades acquired in courses ACO202, ACO201, and CSC226. The configuration table below was used to get the best results from the algorithm.

| Configuration | Value |
|-------------------|-----------|
| Confidence Factor | 0.35 |
| Test Mode | 66% Split |

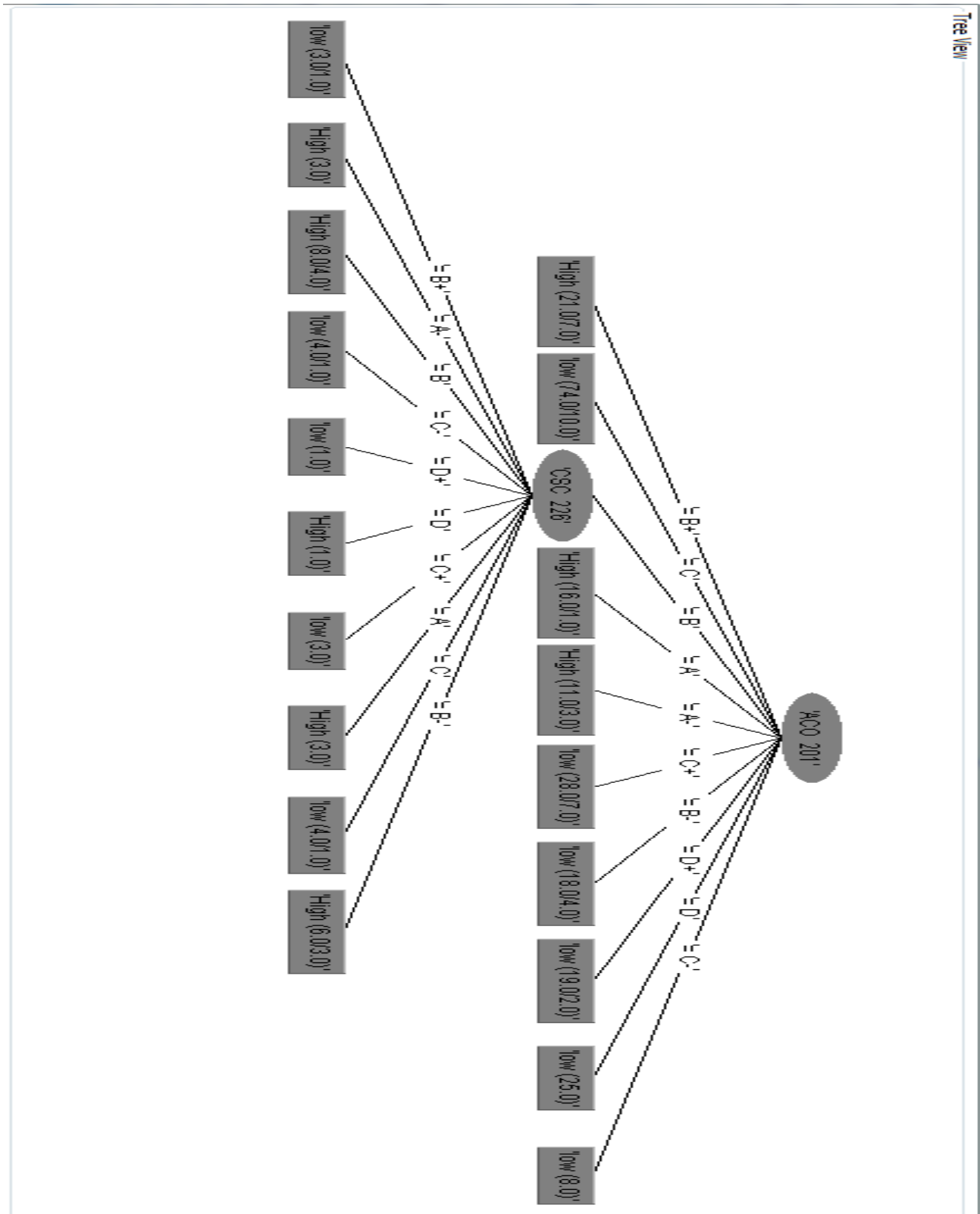
Table 5 J48 Configuration (01-BC-S3a)

This test mode splits the 256 instances into 169 training instances and 87 testing instances. The results below were acquired.

| Result | Value |
|--|---------|
| Correctly Classified Instances | 71 |
| Percentage of Correctly Classified Instances | 81.6092 |
| Relative Absolute Error | 73.3552 |
| F-Measure (High) | 0.724 |
| F-Measure (Low) | 0.862 |

Table 6 J48 Results (01-BC-S3a)

Figure 11 Decision Tree (01-BC-S3a) size of tree is 21 with 19 leaves



The following confusion matrix was built.

| | Predicted False | Predicted True |
|--------------|-----------------|----------------|
| Final GPA | High | Low |
| Actual False | 21 | 6 |
| Actual True | 10 | 50 |

Table 7 J48 Confusion Matrix (01-BC-S3a)

O1-BC-S3b: Multilayer Perceptron (MLP) was used to predict the final GPA for each student from the grades acquired in courses ACO202, ACO201 and CSC226. The configuration below was used to get the best results from the algorithm.

| Configuration | Value |
|-------------------------|----------|
| Number of Hidden Layers | 6*6*6 |
| Learning Rate | 0.3 |
| Momentum | 0.2 |
| Test Mode | 10 folds |

Table 8 MLP Configuration (01-BC-S3b)

The results below were acquired after running the MLP (Multilayer Perceptron).

| Result | Value |
|--|---------|
| Total Instances | 256 |
| Correctly Classified Instances | 203 |
| Percentage of Correctly Classified Instances | 79.2969 |
| Relative Absolute Error | 55.8506 |
| F-Measure (High) | 0.662 |
| F-Measure (Low) | 0.851 |

Table 9 MLP Results (01-BC-S3b)

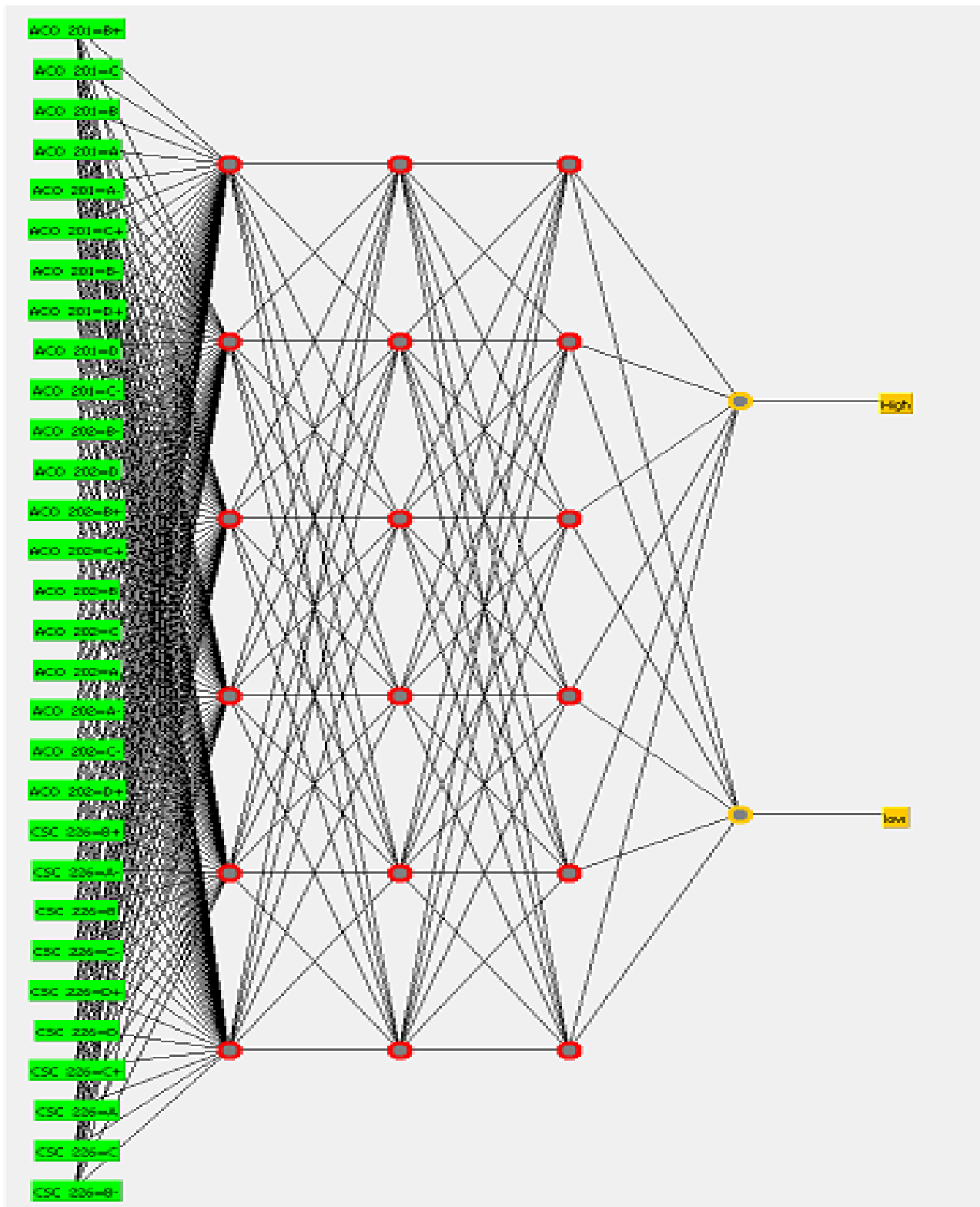


Figure 12 MLP (01-CS-S3b) 30 input perceptron, 2 output perceptron, and 3 hidden layers with 6 Perceptron each.

The confusion matrix hereunder was built.

| | Predicted False | Predicted True |
|--------------|-----------------|----------------|
| Final GPA | High | Low |
| Actual False | 52 | 25 |
| Actual True | 28 | 151 |

Table 10 MLP Confusion Matrix (01-BC-S3b)

Thus, this experiment shows that for business computing major we can predict the final GPA of each student from the grades of the 3 courses selected, and they are: ACO 202, ACO 201 and CSC 226.

01-CS-S1: For computer science students, the 5 courses chosen were CSC 213, CSC 212, CSC 201, MAT 215 and MAT 211.

01-CS-S2: ReliefFAttributeEval was used to choose the best 3 out of 5 courses. According to ReliefFAttributeEval, the courses were ranked as follows:

| Course Number | Influence Factor |
|---------------|------------------|
| CSC213 | 0.20989 |
| CSC212 | 0.16187 |
| CSC201 | 0.08567 |
| MAT215 | 0.06911 |
| MAT211 | 0.00728 |

Table 11 Courses evaluation according to Relief Attribute Evaluation (01-CS-S2)

O1-CS-S3a: Decision Tree J48 was used to predict the final GPA for each student from the grades acquired in courses CSC213, CSC212, and CSC201. The configuration below was used to get the best results from the algorithm.

| Configuration | Value |
|-------------------|-----------|
| Confidence Factor | 0.45 |
| Test Mode | 70% Split |

Table 12 J48 Configuration (O1-CS-S3a)

This test mode splits the 150 instances into 105 training instances and 45 testing instances. The following results were acquired.

| Result | Value |
|--|---------|
| Correctly Classified Instances | 38 |
| Percentage of Correctly Classified Instances | 84.4444 |
| Relative Absolute Error | 50.479 |
| F-Measure(High) | 0.821 |
| F-Measure(Low) | 0.863 |

Table 13 J48 Results (O1-CS-S3a)

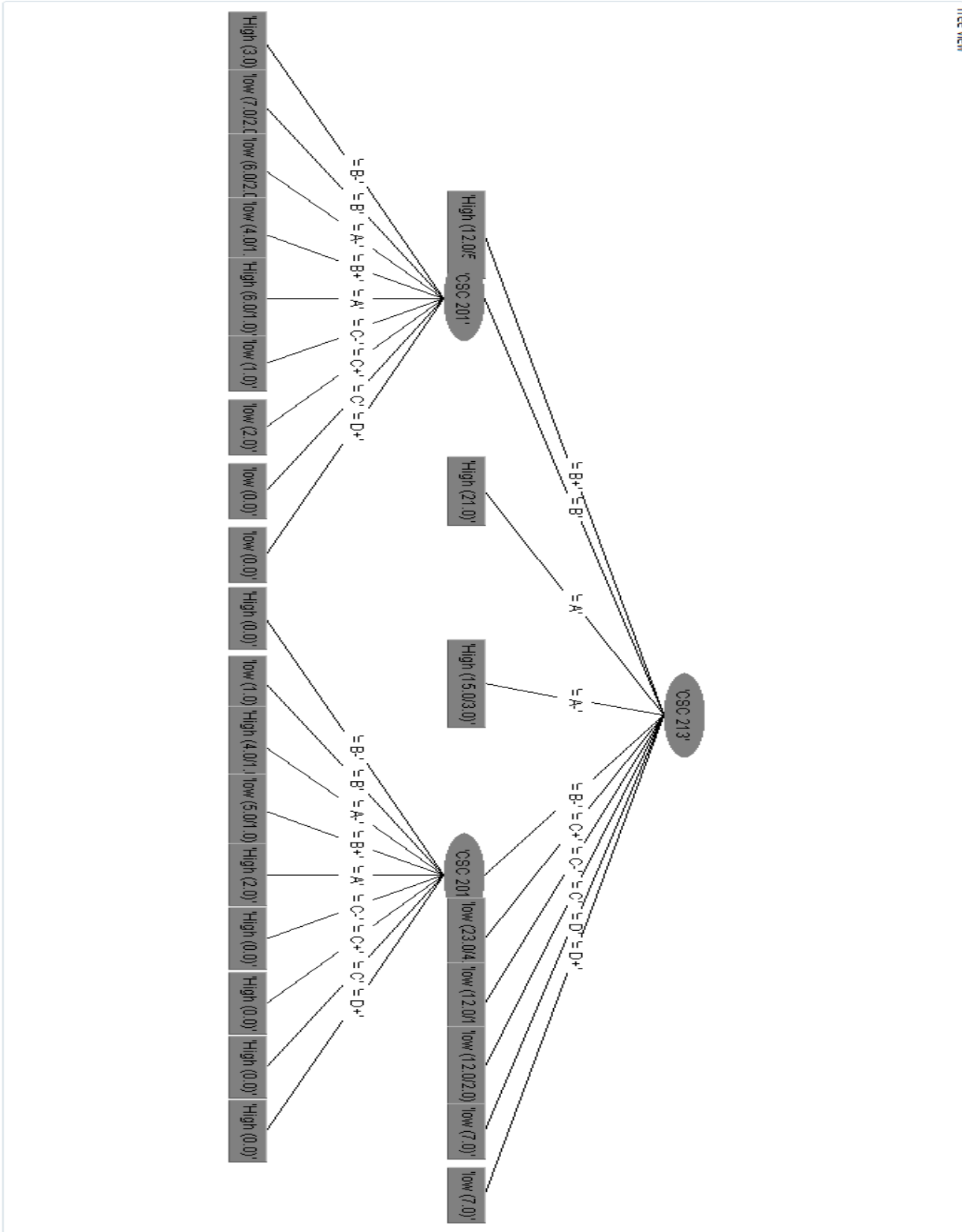


Figure 13 Decision Tree (01-CS-S3a) size of tree is 29 with 26 leaves

The confusion matrix below was built.

| | Predicted False | Predicted True |
|--------------|-----------------|----------------|
| Final GPA | High | Low |
| Actual False | 16 | 3 |
| Actual True | 4 | 22 |

Table 14 J48 Confusion Matrix (01-CS-S3a)

01-CS-S3b: Multilayer Perceptron was used to predict the final GPA for each student from the grades acquired in courses CSC213, CSC212 and CSC201. The configuration below was used to get the best results from the algorithm.

| Configuration | Value |
|-------------------------|-----------|
| Number of Hidden Layers | 6*6*6 |
| Learning Rate | 0.3 |
| Momentum | 0.2 |
| Test Mode | 70% Split |

Table 15 MLP Configuration (01-CS-S3b)

This test mode splits the 150 instances into 105 training instances and 45 testing instances. The following results were acquired.

| Result | Value |
|--|---------|
| Correctly Classified Instances | 41 |
| Percentage of Correctly Classified Instances | 91.1111 |
| Relative Absolute Error | 35.5427 |
| F-Measure (High) | 0.9 |
| F-Measure (Low) | 0.92 |

Table 16 MLP Results (01-CS-S3b)

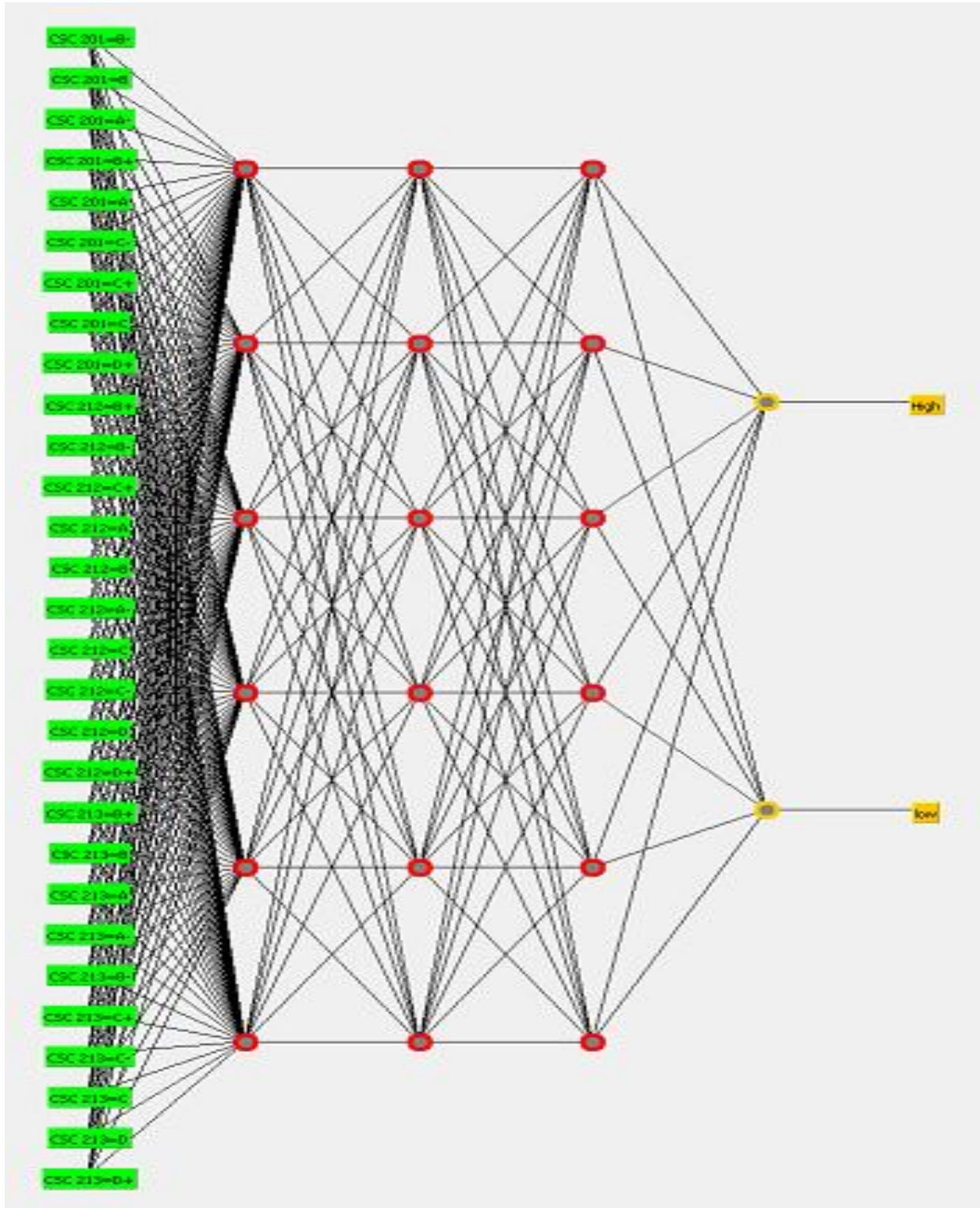


Figure 14 MLP (01-CS-S3b) 29 input perceptrons, 2 output perceptrons, and 3 hidden layers with 6 Perceptrons each.

The confusion matrix below was built.

| | Predicted False | Predicted True |
|--------------|-----------------|----------------|
| Final GPA | High | Low |
| Actual False | 18 | 1 |
| Actual True | 3 | 23 |

Table 17 MLP Confusion Matrix (01-CS-S3b)

Thus, this experiment shows that for Computer Science major we can predict the final GPA of each student from the grades of the 3 courses selected, and they are: CSC 213, CSC 212 and CSC 201.

Objective2: Predict Final GPA from Semester1 and Semester 2. Business Computing Students Case BC and Computer Science students Case CS.

O2-BC-Sa: Decision Tree J48 is used to predict the final GPA for Business Computing students from the grades acquired in Semester 1 and Semester 2. The configuration below is used to get the best results from the algorithm.

| Configuration | Value |
|-------------------|----------|
| Confidence Factor | 0.45 |
| Test Mode | 10 Folds |

Table 18 J48 Configuration (02-BC-Sa)

The results hereunder were noticed.

| Result | Value |
|--|---------|
| Total Instances | 327 |
| Correctly Classified Instances | 251 |
| Percentage of Correctly Classified Instances | 76.7584 |
| Relative Absolute Error | 72.3688 |
| F-Measure(A) | 0 |
| F-Measure(B) | 0.542 |
| F-Measure(C) | 0.851 |

Table 19 J48 Results (02-BC-Sa)

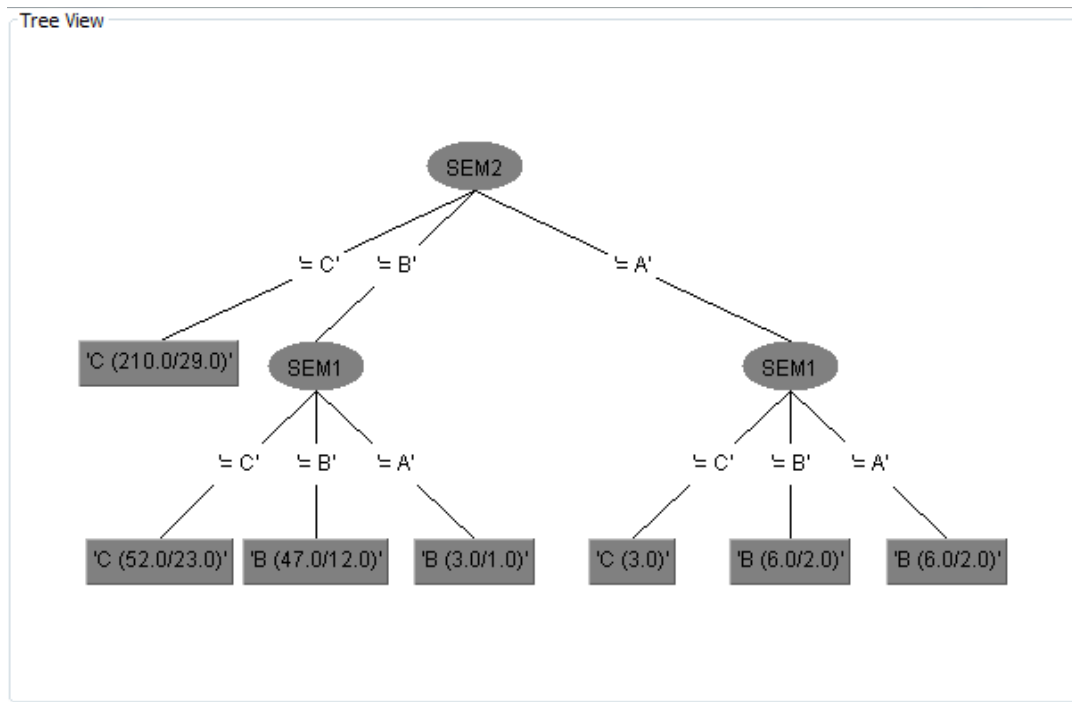


Figure 15 Decision Tree (02-BC-S3a) size of tree is 10 with 7 leaves

The confusion matrix below was built.

| | Predicted C | Predicted B | Predicted A |
|----------|-------------|-------------|-------------|
| Actual C | 206 | 20 | 0 |
| Actual B | 52 | 45 | 0 |
| Actual A | 0 | 4 | 0 |

Table 20 J48 Confusion Matrix (02-BC-Sa)

O2-BC-Sb: Multilayer Perceptron was used to predict the final GPA for business Computing students from the grades acquired in Semester 1 and Semester 2. The following configuration was used to get the best results from the algorithm.

| Configuration | Value |
|-------------------------|----------|
| Number of Hidden Layers | 6*6*6 |
| Learning Rate | 0.3 |
| Momentum | 0.2 |
| Test mode | 10 folds |

Table 21 MLP Configuration (02-BC-Sb)

The results below were acquired.

| Result | Value |
|--|---------|
| Total Instances | 327 |
| Correctly Classified Instances | 257 |
| Percentage of Correctly Classified Instances | 78.5933 |
| Relative Absolute Error | 69.9095 |
| F-Measure(A) | 0 |
| F-Measure(B) | 0.583 |
| F-Measure(C) | 0.863 |

Table 22 MLP Results (02-BC-Sb)

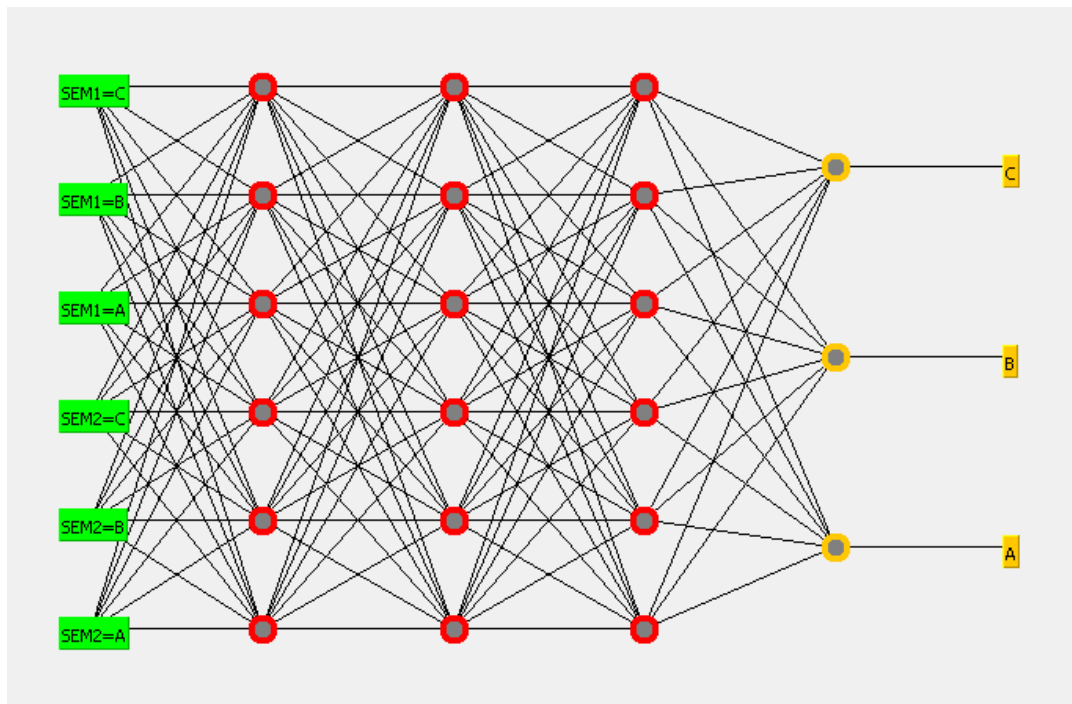


Figure 16 MLP (02-BC-Sb) 6 input perceptron, 3 output perceptron, and 3 hidden layers with 6 Perceptron each.

The following confusion matrix was built.

| | Predicted C | Predicted B | Predicted A |
|----------|-------------|-------------|-------------|
| Actual C | 208 | 18 | 0 |
| Actual B | 48 | 49 | 0 |
| Actual A | 0 | 4 | 0 |

Table 23 MLP Confusion Matrix (02-BC-Sb)

O2-CS-Sa: Decision Tree J48 was used to predict the final GPA for Computer Science students from the grades acquired in Semester 1 and Semester 2. The next configuration was used to get the best results from the algorithm.

| Configuration | Value |
|-------------------|-----------|
| Confidence Factor | 0.45 |
| Test Mode | 66% Split |

Table 24 J48 Configuration (02-CS-Sa)

This test mode splits the 233 instances into 153 training instances and 80 testing instances. The results below were acquired.

| Result | Value |
|--|---------|
| Correctly Classified Instances | 60 |
| Percentage of Correctly Classified Instances | 75.9494 |
| Relative Absolute Error | 72.185 |
| F-Measure(A) | 0 |
| F-Measure(B) | 0.642 |
| F-Measure(C) | 0.869 |

Table 25 J48 Results (02-CS-Sa)

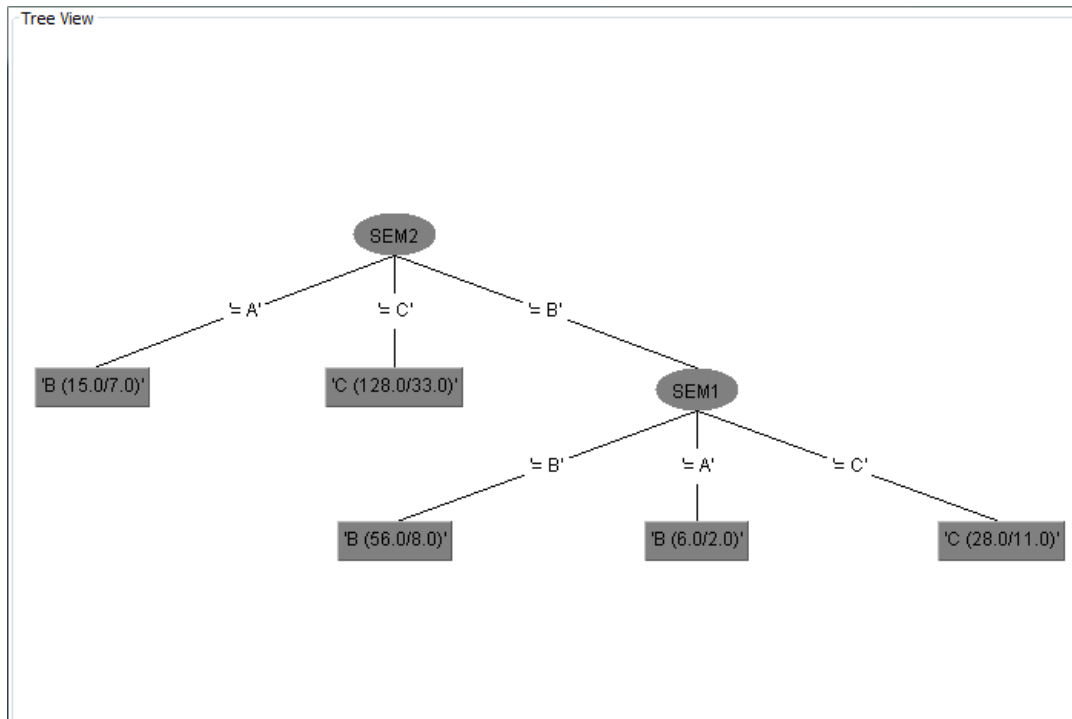


Figure 17 Decision Tree (02-CS-S3a) size of tree is 7 with 5 leaves

The confusion matrix below was built.

| | Predicted B | Predicted A | Predicted C |
|----------|-------------|-------------|-------------|
| Actual B | 17 | 5 | 10 |
| Actual A | 1 | 0 | 0 |
| Actual C | 3 | 0 | 43 |

Table 26 J48 Confusion Matrix (02-CS-Sa)

O2-CS-Sb: Multilayer Perceptron was used to predict the final GPA for Computer Science students from the grades acquired in Semester 1 and Semester 2. The Below configuration below was used to get the best results from the algorithm.

| Configuration | Value |
|-------------------------|-----------|
| Number of Hidden Layers | 6*6*6 |
| Learning Rate | 0.3 |
| Momentum | 0.2 |
| Test mode | 70% Split |

Table 27 MLP Configuration (02-CS-Sb)

This test mode splits the 233 instances into 163 training instances and 70 testing instances. The results below were acquired.

| Result | Value |
|--|---------|
| Correctly Classified Instances | 58 |
| Percentage of Correctly Classified Instances | 82.8571 |
| Relative Absolute Error | 65.0858 |
| F-Measure(A) | 0 |
| F-Measure(B) | 0.769 |
| F-Measure(C) | 0.874 |

Table 28 MLP Results (02-CS-Sb)

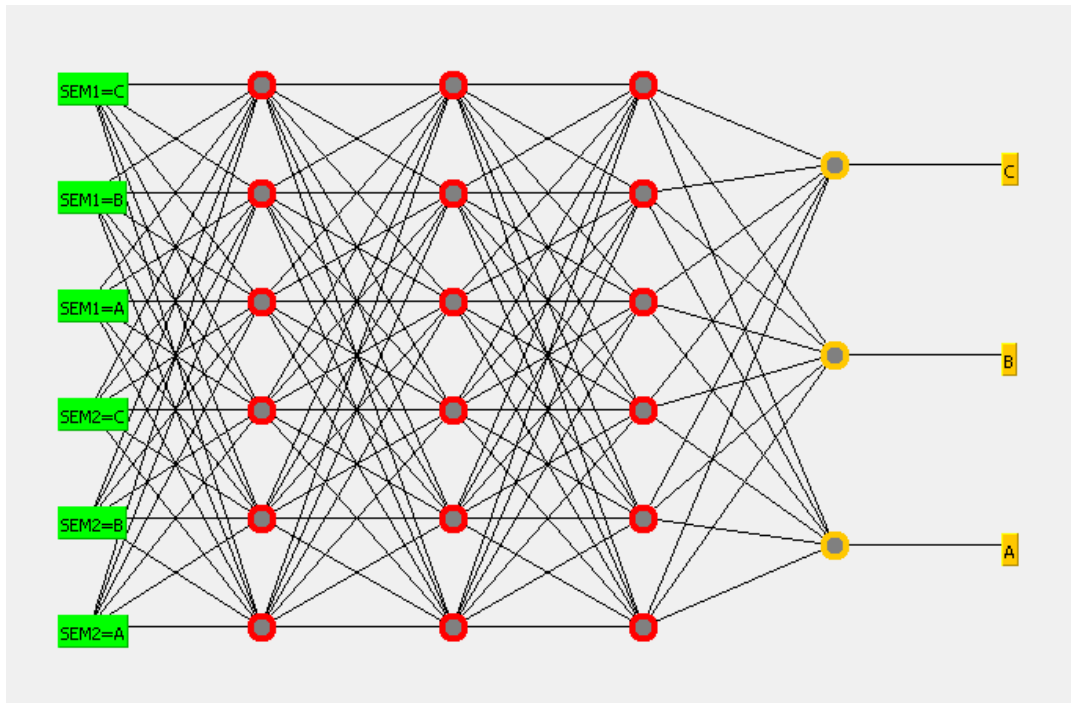


Figure 18 MLP (02-CS-Sb) 6 input perceptron, 3 output perceptron, and 3 hidden layers with 6 Perceptron each.

The confusion matrix below was built.

| | Predicted B | Predicted A | Predicted C |
|----------|-------------|-------------|-------------|
| Actual B | 20 | 0 | 8 |
| Actual A | 1 | 0 | 0 |
| Actual C | 3 | 0 | 38 |

Table 29 MLP Confusion Matrix (02-CS-Sb)

This experiment shows that for computer science major we can predict the final GPA of each student from the GPA of the grades of the first and second semesters.

Chapter 6: Conclusion and Future Work

This research covers various fields in computer science and education; mainly Educational data mining, Machine learning and Artificial Intelligence for Education. EDM brings together an interdisciplinary community of computer scientists, learning scientists, and researchers from other fields. EDM applies techniques coming from statistics, machine learning, and data mining to analyze data collected during teaching and learning, tests learning theories, and informs decision-making in educational practice. The field of EDM has grown substantially in recent years with two related annual conferences, two books, several surveys in books and journals, an increasing number of specific tools, and so forth. Currently, there is a large amount of work to be done in the EDM community in order for it to be considered as a mature area. EDM has to be much more widely used and applied, not only by researchers but also by teachers and institutions (Romero C. &, 2010).

6.1 Conclusion

To support the aims of higher education institutions, this thesis proposed a model that would predict student performance in future courses based on the grades of the students in the first two semesters and the courses that affect the student's final GPA the most. The workflow for our system was presented in figure 5 in chapter 4 study case section. A case study was also presented, using student records obtained from the Department of Computer Science in NDU, for the programs of business computing and computer science. The prediction system is based on the decision tree classification methods, J48 and the neural network method, Multilayer Perceptron.

First, a preprocessing step was applied to the data and then the model was built using training data for each of the objectives. By applying the algorithms for each objective, we found that MLP achieved a better accuracy performance the J48. Below is a table of the results:

| Objective 1 | | | | Objective 2 | | | |
|-------------|-----------|-----------|-----------|-------------|----------|----------|----------|
| O1-BC-S3a | O1-BC-S3b | O1-CS-S3a | O1-CS-S3b | O2-BC-Sa | O2-BC-Sb | O2-CS-Sa | O2-CS-Sb |
| 81.6092 | 79.2969 | 84.4444 | 91.1111 | 76.7584 | 78.5933 | 75.9494 | 82.8571 |

Table 30 List of results.

6.2 Future Work

We have seen how fast EDM is growing as reflected in the increasing number of contributions published every year in international conferences and journals and the number of specific tools specially developed for applying DM algorithms in educational data/environments. Therefore, it could be said that EDM is now approaching its adolescence, i.e., it is no longer in its early days, but is not yet a mature area.

In an aim to increase the accuracy of the tests done in this research, more algorithms can be applied like K means and Support vector machine, then these results can be compared with the results from this research to have a better understanding of the students' performance. Also having more recent data will result in much more accurate results related to the current courses offered by the university.

References

- Atkinson, P., and Tatnall, A. (1997). Introduction Neural Networks in Remote Sensing. *International Journal of Remote Sensing*, 699-709.
- Baker, R. S. (2010). Data mining for education. *International encyclopedia of education*, (pp. 112-118.).
- Berkhin, P. (2002). *Survey of Clustering Data Mining Techniques*. Retrieved from Accrue Software, Inc.
- Caudill, M. (1989). *Neural Network Primer, Part I*.
- Don R. Hush, Bill G. Horne. (1993). Progress in supervised neural networks. *IEEE Signal Processing Magazine*, 8-39.
- Eibe Frank, M. H. (2005). Data mining and knowledge discovery handbook. In *WEKA-A machine workbench for data mining* (pp. 1305-1314). Springer, Boston, MA. doi:https://doi.org/10.1007/978-0-387-09823-4_66
- Gizem, A. &. (2009). *Coomunications & Networks, Network Books*.
- Gupta, T. (2017, August 20). *Deep Learning: Regularization Notes*. Retrieved from Towards Data Science: <https://towardsdatascience.com/deep-learning-regularization-notes-29df9cb90779>
- Han, J. K. (2011). *Data mining: Concepts and techniques* (3 ed.). San Francisco, California, USA: Morgan Kaufmann Publishers Inc. .
- Hancock, T. (1996, May). Lower Bounds on Learning Decision Lists and Trees. *Information and Computation*, 126(2), 114-122.
- Hanna, M. (2004). Data mining in the e-learning domain. *Campus-Wide Inf. Syst.*, 29-34.
- I.V, S. (2017, March 25). Top 10 open source data mining tools.
- Ian Goodfellow, Yoshua Bengio. (2016). *Deep Learning*.
- Igor Kononenko, M. R.-S. (n.d.). ReliefF for estimation and discretization of attributes in classification, regression and ILP problems.
- John F. Kolen, Jordan B. Pollack. (1990). Backpropagation is Sensitive to Initial Conditions. *Complex Systems*, 4(3), 269-280.
- John Hertz, Anders Krogh, Richard G. Palmer. (1991). *Introduction to the theory of neural computation*. Addison-Wesley Longman Publishing Co.,.

- John McGonagle, George Shaikouski, Andrew Hsu,. (n.d.). *Backpropagation*. Retrieved from Brilliant.org: <https://brilliant.org/wiki/backpropagation/>
- Kenji Kira, L. R. (1992). The feature selection problem: traditional methods. *AAAI-92*.
- Laurent Hyafil, R. L. (1976). *Constructing Optimal Binary Decision Trees is NP-complete*.
- Leo Breiman, J. F. (1984). *Classification and Regression Trees* . Wadsworth,New York: Chapman and Hall.
- Lior Rokach, Oded Maimon. (2014). Decision Trees. In *Data Mining and Knowledge Discovery Handbook* (pp. 166-187).
- Neeraj Bhargava, G. S. (2013). Decision tree analysis on J48 algorithm for data mining. *Proceedings of International Journal of Advanced Research in Computer Science and Software Engineering*,.
- Nesterov, Y. (2013). Introductory lectures on convex optimization: A basic course. *Springer Science & Business Media*, 87.
- Nesterov, Y. (2013). Introductory lectures on convex optimization: A basic course. *Springer Science & Business Media*, 87.
- Nielsen, M. A. (2015). *Neural Network and Deep Learning*. Determination Press.
- Olson, D. D. (2008). *Advanced data mining techniques*. (1 ed.). Springer Publishing Company, Inc.
- Pang-Ning Tan, M. S. (2005). Association analysis: basic concepts and algorithms. *Introduction to Data Mining*, 360-362.
- Polyak, B. (1964). Some Methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathemantical Physics*, 1-17,19.
- Qiong Gu, Z. C. (2008). Data mining on imbalanced data sets. *In advanced computer theory and engineering*. IEEE . doi:10.1109/ICACTE.2008.26
- Rahul B. Diwate, A. S. (2014). Data Mining Techniques in Association Rule. *International Journal of Computer Science and Information Technologies*, 227-229.
- Rebizant W. et. AL. (2011). Fundamentals of System analysis and Synthesis. In *Digital Signal Processing in Power system Protection and Control Signals and Communication Technology* (pp. 29-52).
- Rodrigo C. Barros, A. C. (2015). *Automatic Design of Decision-Tree Induction Algorithms*. Springer.

- Romero, C. &. (2010). Educational data mining: a review of the state of the art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40(6), pp. 601-618.
- Romero, C. (2010, November). Educational Data Mining: A Review of the State of the Art. *EE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICSS—PART C: APPLICATIONS AND REVIEWS*, 40, pp. 601-618.
- Rumelhart and McClelland. (1986). *The appeal of parallel distributed processing*.
- Rus, V. &. (2009). Automatic Detection of Student Mental Models During Prior Knowledge Activation in MetaTutor. *Proceedings of the 2nd International Conference on Educational Data Mining.*, (pp. 161-170).
- Ryan Baker, K. Y. (2009). The state of educational data mining in 2009: A review and future visions. *Journal of Education DataMining*, 1(1), 3-17.
- Schmidhuber, J. (2015). Deep learning in neural networks. *Neural Network*, 61, 85-117.
- Seggern, D. H. (2007). *CRC Standard Curves and Surfaces with Mathematics*. (Second ed.). CHAPMAN & HALL.
- Shannon, C. E. (1948). A Mathematical Theory of Communication. *The Bell System Technical Journal*, 27, 379–423, 623–656.
- Shannon, C. E. (1948). A Mathematical Theory of Communication. *The Bell System Technical Journal*, 27, 379–423, 623–656,.
- Tiffany Barnes, M. D. (2009). Educational data mining 2009. In *Proceedings 2nd International Conference on Educational Data Mining*. Cordoba, Spain.
- Utgoff, P. (1989). Incremental induction of decision tree. In *Machine Learning* (pp. 161-186). doi:10.1023/A:1022699900025
- Ventura, C. R. (2010). Educational Data Mining: A Review of the State of the Art. *IEEE transactions on Systems, Man, and Cybernetics—part c: Applications and Reviews*. Retrieved from IEEE transactions on Systems, Man, and Cybernetics—part c: Applications and Reviews.
- Ventura, C. R. (2010, November). Educational Data Mining: A Review of the State of the Art. *IEEE transactions on Systems, Man, and Cybernetics—part c: Applications and Reviews*, 20(6).
- Zell, A. (1994). *Simulation Neuronaler Netze[Simulation of Neural Network*. Germany.