Evaluating the Performance of Educational Data Mining Algorithms on Features Extracted from Students Taking Programming Courses: A Case Study with Heriot-watt University

by

CHINONSO EZINNA EMMA-EBERE

H00255971

August 2017

Supervisor: Smitha Kumar
Second Reader: Chris Fensch

Computer Science
School of Mathematical and Computer Sciences

Dissertation submitted as part of the requirements for the award of the degree of MSc in Information Technology (Business)
Declaration

I, Chinonso Ezinna Emma-Ebere, confirm that this work submitted for assessment is my own and is expressed in my own words. Any uses made within it of the works of other authors in any form e.g., ideas, equations, figures, text, tables, programs) are properly acknowledged at any point of their use. A list of the references employed is included.

Signed: Chinonso Ezinna Emma-Ebere

Date: 14th August 2017
DEDICATION

To

The All Mighty God
For sticking with me and giving me strength when I needed it the most. This Journey would not have been possible without you. Thank you for giving me a great family and wonderful friends.

My Beloved Dad and Mum
For earning a very honest living and selflessly depriving themselves of the pleasures of life so that all of us, Obiajulu, Chioma, Chinwenwa, Uloma and myself can be happy. I love you more than you can ever imagine. I did all these for you.

My Wonderful Siblings
For supporting our parents to see me through this beautiful institution and providing for me when I had nothing, May God bless your efforts and your families. I promise to take care of you however I can. I love you all so much.
ACKNOWLEDGEMENT

My first acknowledgement goes to my best friend Chioma. You have always been a source of motivation to me. Thank you for letting me believe I can achieve the impossible if I just try. May God continue to keep and bless you.

Secondly, I would love to acknowledge my wonderful supervisor Mrs. Smitha Kumar. Sometimes I wonder how you managed to push me this much to do something I never taught i can do. Thank you for your guidance throughout.

Special Thanks also goes to Dr. Steve Gill and Dr. Hind Zantout who were always there to advise me on the best strategy to handle the constant pressure in this school.

I would also love to specially acknowledge Ogechukwu and Ojiugo, my siblings from another mother, who threatened to abandon me if I do not come out with a good result.

I cannot ignore the family I made here. A special salute goes to Dumebi and the rest of the house (Ademola, Faibee, Oluwasesan, Shalom, Jackie and Hilary). Thanks for giving me a home in a foreign land.

Finally, I want to acknowledge my friends (Abdul, Chima, Chineny, Kola, Favor, Mavis, Anna, Rishad, Ayesha, Matew, Micheal, Shyari, Ada Maj) you who made my stay here memorable. I will keep celebrating you guys forever.
ABSTRACT

Educational Data Mining (EDM) is a growing research field and has made a great impact within the educational institutions. Its ability to borrow techniques from Machine Learning (ML) and data mining has been utilized to great effect within the educational system. Its algorithms have been used to improve the effectiveness in the administrative, academic, financial and strategic decision-making within the educational institutions [1, 2, 3].

Programming can be a hard concept to grasp by some students and there is a lot of research on how to support students having this issue [4]. This research investigates the features (attributes) that impact the performance of students taking programming courses in Heriot-watt University. The data was obtained from the students via questionnaires. This research also compares various EDM classifier models (algorithms) that can accurately predict a student’s performance when applied to those features. Four EDM classifier models were built and their performances were compared, before and after feature reduction. The feature reduction process was done using four filtering techniques to reduce dimensionality within the feature subset. The best classifier model was chosen based on its performance when evaluated using F-measure, AUC (area under the curve) and prediction accuracy. All these are common evaluation metrics used in various EDM literature.

The results from our analysis showed that EDM techniques are suitable to identify what feature subset has the most influence on a student’s performance in school. Correlation-based filter algorithm identified 35 feature subsets where all four classifiers had the highest average performance than the other feature subsets. A student’s emotional stability in school and their experience in python programming was discovered to be the most influential features based on our experiments. Random forest classifier performed better within that set of features than the other classifiers. However, when compared to its initial values, only its F-measure value showed a statistically significant improvement.
# Table of Contents

Declaration ........................................................................................................................................ i
DEDICATION ...................................................................................................................................... ii
ACKNOWLEDGEMENT ....................................................................................................................... iii
ABSTRACT .......................................................................................................................................... iv
List OF Figures .................................................................................................................................. vi
List of Tables ...................................................................................................................................... viii

CHAPTER 1: INTRODUCTION ........................................................................................................ 1
  1.1 Introduction .................................................................................................................................. 1
  1.2 Aims and Objectives .................................................................................................................... 2
    1.2.1 Hypothesis Statement and Project Rationale ......................................................................... 2
    1.2.2 Project Aims .......................................................................................................................... 2
    1.2.3 Project Objectives ................................................................................................................ 2
  1.3 Dissertation Structure ................................................................................................................ 4

CHAPTER 2: LITERATURE REVIEW ............................................................................................. 5
  2.1 Educational Data Mining (EDM) ............................................................................................... 5
    2.1.1 Objectives of EDM ................................................................................................................. 6
    2.1.2 Importance of EDM ............................................................................................................... 6
    2.1.3 Components of EDM ............................................................................................................ 7
  2.2 Related Works on Features influencing Students performance .............................................. 7
  2.3 Data Mining Methodology ....................................................................................................... 8
  2.4 Business Understanding and Requirements gathering ............................................................ 10
  2.5 Data preparation ........................................................................................................................ 10
    2.5.1 Under-Sampling the Majority Class .................................................................................... 10
    2.5.2 Over-sampling the Minority Class ....................................................................................... 11
  2.6 Data Pre-processing: Dimensionality Reduction ................................................................. 13
    2.6.1 Dimensionality and Dimensionality Reduction .................................................................. 13
    2.6.2 Comparing the Feature selection techniques ...................................................................... 16
    2.6.3 Filter Algorithms ................................................................................................................ 16
  2.7 Data Modeling: Strategies, Techniques and Algorithms in EDM ........................................ 19
    2.7.1 EDM and Machine Learning .............................................................................................. 19
    2.7.2 EDM Learning Strategies .................................................................................................. 19
    2.7.3 EDM Techniques and Methods .......................................................................................... 20
    2.7.4 Predictive EDM Techniques ............................................................................................... 21
    2.7.5 Predictive EDM Algorithms (Classification Algorithms) .................................................. 21
  2.8 Training and Testing the Classifier Model .............................................................................. 25
  2.9 Classifier Model Evaluation ..................................................................................................... 26
    2.9.1 Confusion Matrix ................................................................................................................ 26
    2.9.2 Common EDM Evaluation Metrics derived from The Confusion Matrix ......................... 27
  2.10 Model Deployment .................................................................................................................... 28
  2.11 Waikato Environment for Knowledge Analysis - WEKA ...................................................... 29
List Of Figures

Figure 1: The dimensions that may affect the performance of students taking programming courses [2]

Figure 2: Steps to Knowledge discovery in Data Mining Source: [14]

Figure 3: Illustrating the Different disciplines where EDM takes its tolls and techniques from [18]

Figure 4: Educational Environments where EDM can be performed [18]

Figure 5: Shows the results from a survey on Data mining methodology preference [28]

Figure 6: Outline of the CRISP-DM and the interdependencies of the Knowledge discover process. [29]

Figure 7: Showing an Under-sampling process done on the Majority class using data from rain [35].

Figure 8: The Synthetic Minority Oversampling Technique [38]
Figure 9: Process of data cleansing or feature Subset selection [48]
Figure 10: Embedded or Hybrid Methods for Feature reduction. Source: [48]
Figure 11: Different strategies to learning and the categories of algorithms [54]
Figure 12: Shows the main categories of data mining techniques with some examples [55]
Figure 13: Shows the single class relationship between independent variables in the Naïve Bayes algorithm [59]
Figure 14: Shows the decision process of a decision tree algorithm like J48 [63]
Figure 15: Random Forest Based Classification Algorithm [67]
Figure 16: Illustrates a logistic distribution by categorizing the predicted class [70]
Figure 17: 5-fold cross validation testing [75]
Figure 18: Depicts the Area Under the ROC Curve (source: stack overflow)
Figure 19: WEKA Start Up GUI
Figure 20: Flow Chart Illustrating the entire methodology for the dissertation
Figure 21: Features from Students Data Set to be extracted from the questionnaire
Figure 22: WEKA Knowledge Flow Interface for Data Mining Tasks
Figure 23: Evaluation Metrics Offered by WEKA Datamining Software
Figure 24: Attribute Selected Classifier Using IG to Rank the Attributes
Figure 25: Imbalanced Data set with 44 Instances
Figure 26: F-measure and AUC performance Of the Classifiers on an imbalanced data set
Figure 27: Balanced Dataset with 60 Instances and No Feature Reduction Applied
Figure 28: F-measure and AUC performance Of the Classifiers on a balanced data set
Figure 29: Summary Chart for Naïve Bayes Classifier When All the Filter Algorithms were Applied
Figure 30: Summary Chart for Random Forest Classifier When All the Filter Algorithms were Applied
Figure 31: Summary Chart for LR Classifier When All the Filter Algorithms were Applied
Figure 32: Summary Chart for J48 Classifier When All the Filter Algorithms were Applied
Figure 33: Visualizing the IG-28 Tree from J48 Classifier
Figure 34: Visualizing the IG-35 Tree from J48 Classifier
Figure 35: Classifier Performance over different Feature Subsets
Figure 36: Sample Performance Evaluation for Random Forest On CB-35 Feature Subset
Figure 37: F-measure Values Before and After Feature Reduction
Figure 38: AUC Values Before and After Feature Reduction
Figure 39: AUC before Feature Reduction for All the Classifiers
Figure 40: AUC after Feature Reduction for All the Classifiers
Figure 41: Applying the Model from Question 5 on 10% of the same data it was built from
Figure 42: Test File with 6 instances as an unseen data

List of Tables

Table 1: Showing the New Data Generated After SMOTE was Applied
Table 2: Showing the New SMOTE Data that has Been Randomized
Table 3: Comparing the different feature selection algorithms [44] [42]
Table 4: Illustrates a confusion Metrics for evaluating a prediction model for people who had breakfast
Table 5: Peak values for F-measure and AUC for all the classifiers (First Part)
Table 6: Peak values for F-measure and AUC for all the classifiers (Second Part)
Table 7: Feature Subset and Feature Code Summary Table
Table 8: Final Performance Evaluation for Each Classifier on CB-35
Table 9: Final Feature Subset with ** & ~ indicating the most Relevant and Least Relevant Features
Table 10: p-values Obtained from the t test experiment for each classifier
Table 11: Showing the Classifier Evaluation Metrics Obtained from the New Training data
Table 12: Showing the Classifier Evaluation Metrics Obtained from the New Unseen Test data
CHAPTER 1: INTRODUCTION

1.1 Introduction

Programming courses have been a piece of cake for some students while others have found it very difficult to learn. The reason for this difficulty has been discussed long ago in conferences by [5, 6] and we still search for solutions [7]. A study by [8], based on data from 63 institutions, revealed 33% failure in introduction to programming courses. This figure might be small since there were only few institutions willing to give access to their data. However, investigating these attributes using data mining techniques can help design strategies that can benefit to any institution.

Researchers have been able to determine the performance of students by measuring the four dimensions in their learning styles [4]. Students who have a high active, sequential and visual learning style perform better in programming courses than others. EDM techniques were used to reach this conclusion, further highlighting its applicability. This can be seen in Figure 1 below:

![Figure 1: The dimensions that may affect the performance of students taking programming courses [4]](image)

The works by [9] focused mainly on the methods teachers adopt in teaching programming courses. They looked at it from the view of a learner. According to them, students taking programming courses should express competence both in logical understanding and basic programming concepts. The views of [7] also contributed to this by including declarative, memorization, problem solving and abstraction understanding as additional necessities.

The observation from all these studies is that, people who struggle to learn the fundamental concepts of programming during their early school years, usually find programming difficult later. Some other issues raised were that many weak students pursue easier courses and others have poor logic reasoning. Furthermore, some students have poor problem-solving and analytical skills. Generally, they feel that computer programming on its own is a very difficult course to grasp. Some weak students who decide to stay in the course never have the confidence of finishing their programs alone. These students would prefer sympathetic marks from their lecturers so that they can pass [9, 10].
1.2 Aims and Objectives

1.2.1 Hypothesis Statement and Project Rationale

Data is gold and can be used to improve the quality of our institutions in different ways. EDM techniques can be used to utilize our data effectively and efficiently. It is important to show how significant these techniques are to our data mining tasks. By considering students taking programming courses in Heriot watt university, the null and alternative hypothesis states that:

- \( H_{null} \): There is no statistically significant change from the initial F-measure and AUC values for each of the classification algorithms after feature reduction on the chosen feature subset.
- \( H_A \): There is a statistically significant change from the initial F-measure and AUC value of at least one of the classification algorithms after feature reduction on the chosen feature subset.

This research focuses on identifying the most relevant features that are influential in the performance of students taking programming courses. The strengths and weaknesses of the EDM classifier algorithms when they are applied on the features will be compared. Furthermore, an evaluation and comparison would be done using an open source data mining software known as WEKA (Waikato Environment for Knowledge Analysis).

1.2.2 Project Aims

The aim of this dissertation is to determine the best set of optimal features, which affect the performance of students taking programming courses. It also aims to determine the data mining algorithm that gives the best performance when applied to those features. The performance of each algorithm would be evaluated and the results would be compared and analyzed.

1.2.3 Project Objectives

The project objectives are guided by the questions below and provide the basis of the type of data to be obtained:

1) What are the features affecting the performance of students taking programming courses in Heriot-Watt University?

The aim of this question is to identify the relevant features that affect this class of students. By using questionnaires, we would be able to identify some of the features based on some past literature on learnings difficulties of students.

2) How do the different EDM algorithms/techniques perform when applied to the features with an imbalanced class label?

We seek to observe the performance of the classifiers when they have fewer training scenarios. In this case, the class label will be imbalanced, with one class proportionally more than the other classes. By building the classifiers on this imbalanced data, we shall evaluate their performance based on their F-measure and AUC.
values before making our deductions. The classifiers algorithms to be used in this project are Naïve Bayes, Random Forest, Logistic Regression and J48.

3) **How does data-balancing affect the performance of the classifiers with respect to their F-measure and AUC Values, assuming all its features are retained?**

The aim of the question is to see if a data that is balanced is more suitable for EDM tasks. Having observed the results in question 2, we shall pre-process our data by applying a data balancing algorithm to give a more proportionate ratio of the majority and minority classes. We have chosen to use a supervised algorithm known as SMOTE (Synthetic Minority Over-Sampling Technique). As the name implies, it increases the minority class using some sophisticated techniques and at the same time, avoids overfitting the classifier.

4) **How does Dimensionality Reduction technique, Specifically Feature subset selection techniques, affect the performance of these features?**

We shall assume that the initial features are already identified from question 1. There may be some irrelevant features that may affect the performance of our classifier models listed in Question 2. The answer to this question would be gotten from going through a systematic process.

Firstly, we shall investigate the effect of dimensionality based on various literature to be discussed in chapter 2. Various approaches to reduce dimensionality will be considered, one of which is feature subset selection. Afterwards, we shall consider the balanced data set from Question 3 and pre-process the data once again to extract the relevant features. The data pre-processing this time around will involve applying feature selection techniques (using filters) to eliminate the inappropriate data. The feature selection techniques to be used include Correlation-based (CB), Relief-F(RF), Information Gain (IG) and Information Gain Ratio (IGR) feature evaluation techniques. Finally, we shall build our classification models using four (4) classification algorithms and draw our evaluations from there using F-measure and AUC values.

5) **Based on the results from the experiments, what are the most relevant features that determine the performance of a student taking programming courses? Furthermore, which of the four-classifier models is best suited for the feature subset?**

The answer to this question would be based on the results obtained from question 4. The classifier that performs best with the chosen subset of features with respect to F-measure, Accuracy and AUC value will be chosen.

6) **How do the results gotten from building all the classifiers on the final feature subset compare to the initial F-measure and AUC values gotten from the balanced dataset from Question 3? Is the difference in value statistically significant for each of the classifiers?**

This is another important part of report because we aim to see if there was a statistical improvement or not. The first part of the question will be answered by running a comparative review from the values obtained before and after feature reduction for each classifier. The second part would be to perform a t-test. This will be
done by using the initial value and the range of values resulting from the removal of each feature. A p-value of < 0.05 indicates a statistically significant difference.

7) How will each classifier model perform on unseen data applied on the most optimal feature set obtained from question 5?

Although the same data will be used for the second part of the experiment, the data models were trained and tested newly without using the old model. The balanced data set will be split into training and testing data. The classifier models will be built afresh on the new training data using the features identified in Question 5. Each resulting model will be applied on the test data and their results will be evaluated to determine their performance.

1.3 Dissertation Structure

- **Chapter 2**: This chapter provides a literature survey on EDM. It highlights some of the basic concepts, techniques and models that will be used in the dissertation. We will consider some previous work that point out the importance of EDM in any institution. The survey also aims to achieve the following:
  1) Evaluate some of the findings in the reviewed papers and Highlight some common features that affect students studying programming courses
  2) Look at various best practices or methodologies like the CRISP-DM for EDM activities and identify some common tasks, techniques and algorithms used in EDM
  3) Introduce Dimensionality Reduction (Feature Reduction) as a factor to improve the performance of a classifier model before concluding with an introduction to the Data mining tool called WEKA

- **Chapter 3**: This chapter describes the methodology used for the dissertation. It will cover the data gathering method and the step-by-step process involved in achieving our aim. It will also show how our model will be evaluated at each step.

- **Chapter 4**: This chapter demonstrates our experiments and evaluations. We shall also investigate the performance of our classifiers to a test dataset. It is based on these results that we will make our recommendations and conclusions

- **Chapter 5**: Highlights the **Professional, Legal, Ethical and Social Issues** that may affect the process and outcome of the project. The research will demonstrate an understanding of the implications, if any, for each element to the dissertation.

- **Chapter 6**: This chapter summarizes the entire work by highlighting important aspects of the project. Our results from chapter 4 would be summarized and our recommendations would be made. A suggestion shall be made on the future works that can be done to improve the quality of the work.
CHAPTER 2: LITERATURE REVIEW

2.1 Educational Data Mining (EDM)

Data mining is a technique used to identify unique patterns, usually within a large chunk of available data. Its objective is to discover useful information, which might be very difficult to identify due to the magnitude of data. This information can then be used to make important decisions to improve the business, process, or some other activity [11, 12, 1].

EDM is an arm of data mining. Schools that adopted its techniques have planned better at management level; reduced student dropout rate; and even predicted the likelihood of a student getting employed after school [13, 14]. EDM is a process whereby large sets of data are “mined” from a data source from which useful patterns are discovered to create knowledge. These “patterns” are information that have been hidden within these large sets of data. The data source could be from an educational Database, like Moodle, Blackboard and so on. Data could also be generated from people through interaction or via questionnaires [13, 1, 15] (see Figure 2).

![Figure 2: Steps to Knowledge discovery in Data Mining Source: [16]](image)

Techniques like classification, have been used to determine the criteria that students use when evaluating their instructor [17]. Some examples of the algorithms in a classification technique include Decision tree algorithms, Naïve Bayes and Logic Regression.

Figure 3 illustrates the different disciplines from which EDM acquires its techniques. In analyzing data and building classifiers for example, EDM adopts algorithms and techniques from machine-learning, data mining and statistical computations. Infact it is a combination of different computational fields, borrowing and utilizing some of the concepts and techniques within those fields [18, 2].
There is a huge amount of data that is being generated from Learning Managements systems (LMS) within the educational institutions. This sort of data usually represents how students learn. It is from this data that EDM becomes useful because they serve as inputs for analysis [2]. Some examples of the type of data generated by an educational system include:

- Student activity data such as downloads, log activities, exam results and mails
- Staff activity data like assignments uploads and test grading

2.1.1 Objectives of EDM

Due to the massive growth of data in the educational system, a lot of research has adopted EDM techniques over the years. The objectives of EDM as outlined by [18] fall under two main categories:

1. **Academic Objectives**: The aim of these objectives is to directly improve the person (staff or student), the departmental processes and possibly the educational domain. For example, we might intend to introduce and structure for a new course based on the requirements of the job market [2].

2. **Administrative and Organizations objectives**: This would usually require involving higher authorities for reasons like developing academic or affiliate relationships with other schools. For example, management can recommend or structure special courses for a specific category of students whom they feel need more attention [3].

2.1.2 Importance of EDM

Models are developed in EDM to serve as tools to be used on data extracted from an educational database. EDM is still a growing research field and over the years, a lot of the researchers have been able to identify the usefulness of EDM models. Some of the identified benefits include:

- The Methods developed in EDM can help discover unique patterns in the data within an educational database [20]. These methods would help understand the interactions of the students within an educational setting.
- Information gained from EDM can be used to objectively evaluate the performances of their teachers and the academic curricular [2].
- It can guide management’s decisions as to where to make financial investments more. This will in turn build the schools reputation and yield more return on investments.
2.1.3 Components of EDM

The researcher has previously stated that EDM was established from different fields. It is also important to look at what EDM is composed of. Based on different researches we can establish that EDM has 4 basic components

- **Educational Environment**: It is important to note the sort of environments where educational data can be mined from as seen in Figure 4. These can be a traditional class room, or a more dynamic environment such as web based or computational based environment. Dynamic environments could be e-learning systems and learning object repositories [19, 18].

![Figure 4: Educational Environments where EDM can be performed](image)

- **Educational Stakeholders**: The different stakeholders include students, administration, researchers and learners [19].

- **Data Mining Tools and Techniques**: This is the meat of the process were data mining techniques can be applied for prediction tasks (Classification, Clustering and Neural Networks etc.). It can also be used for verification tasks like testing hypothesis [18]. Some of the tools that are used in data mining include WEKA, R and ORANGE.

- **Educational Data**: Some researchers have extracted educational data from LMS such as Blackboard and Moodle. This data is in the form of event-logs. There are also a lot of literatures where questionnaires were used as the mode of data generation particularly from the Educational stakeholders. This approach will also be the case in this dissertation.

2.2 Related Works on Features influencing Students performance

Researchers have put together some interesting features that can influence the way a student performs in school and they have been quite successful [9, 10]. We looked at some of them briefly in chapter one but we will investigate this by going through some related work.
Studies by [13] showed some of the influencing factors that determine if a student would get employed after work using two classifier models. They could use features like examination score, communication skill, and placement preparation hours, breaks taken during the course, extracurricular activities, cultural activities and the number of industrial visits. This research adopts some of these attributes to its analysis.

Reference [21, 22] performed studies from various universities to investigate the student’s skills and understanding of programming. We can conclude that designing, reading and writing a programming code from scratch was a major issue. In other words, some students lacked the essential skills of a good programmer. However, the statistics from [21] was based on first year students only. Students alone are not to blame. The same issue has been faced by people teaching programming courses because their mode of teaching might not be beneficial to the students [23].

Another interesting work done by [24, 25] focused on the difficulties of learning and teaching programming courses. The questionnaire they used was divided into 3 parts and covered the general aspects of basic programming knowledge. They pointed out that students struggle because they do not understand what goes on in memory. One notable limitation though was the scope of their work. It was constrained to Object-oriented languages specifically C++. Another problem that was identified is that some students lack the skills and techniques for solving problems by building programs. Another survey by [26] from five computer science lecturers in a local university showed that using basic constructs and syntaxes when programming is difficult to grasp by the students. This is because most students lack application and understanding of computer programming. This dissertation has adopted most of these concepts to structure the questionnaire in Appendix 1.

Reference [7] identifies some other difficult topics that students face in learning programming courses. They divided their questionnaire into 5 categories which took into consideration, the personal profile and experience of the student. Some of these features was like the findings from the works done by [24, 26]. It was discovered that students struggled with understanding basic programming syntax. Students also found it hard to construct a program from scratch. From the inferences, there are similarities and differences between [24] and [9]. While the former only identified, students understanding of memory usage as a difficulty, the latter discovered more. However, the emphasis was mainly on the areas related to programming. In this dissertation, we will consider students personal attributes and see how the filter algorithms discussed in section 2.6.3 would react to them.

2.3 Data Mining Methodology

Data mining is a systematic process. The quality of any process is in its standardization. The Cross Industry Standard Process for Data mining (CRISP-DM) is a standardized process for data mining across all industries. It is an iterative process that has 6 major stages on its way to knowledge discovery described in Figure 2. It has been used by data miners like [27] and has been proven to be very effective.
There are other data mining methodologies that are used in data mining projects. Some of which include SEMMA, KDD Process and domain Specific methodology. However, a survey performed by Knowledge Discovery nuggets in 2014 showed that CRISP-DM is still the most preferred in Data mining projects. The diagram above shows the results from the survey for data mining methodologies from 2007 to 2014. It is clear to see that from 2007 till 2014, CRISP-DM has been leading the voting pools [28]. This will be the basis for choosing CRISP-DM methodology to explain Knowledge discovery process. An overview of the CRISP-DM is shown in the figure below.

---

**Figure 5:** Shows the results from a survey on Data mining methodology preference [28]

---

**Figure 6:** Outline of the CRISP-DM and the interdependencies of the Knowledge discover process. [29]
Sections 2.4 to 2.10 discusses the stages in the CRISP-DM methodology including the techniques and algorithms used in datamining. These stages will be grouped into Business Understanding and requirement gathering; Data preparation; Data pre-processing; Data Testing; Data Modelling; Evaluation and Deployment [29].

2.4 Business Understanding and Requirements gathering

This is the first phase for any data mining project, business or academic. The steps involved here include:

- **Business Understanding**: The miner must determine the desired output by setting project objectives, producing a project plan and determining the success criteria. The current situation must be assessed, giving more detail about the resources needed, project constraints and making critical assumptions [30]. In the case of this dissertation, we try to do create classifier models and evaluate the accuracy of the different models.

- **Data Understanding**: Having identified the aims and objectives of the data mining project, it is important to gather the data identified earlier. The data should be described and evaluated. Data can be nominal or numeric or any other value. Data can also be discretized to aid simplicity when building our model [29, 14]. The completeness and quality of the data must also be considered.

2.5 Data preparation

The preparation of data depends on the necessary algorithms for data pre-processing that was done initially by the miner [31]. This phase ensures that the activities to model the selected data from the original data set is done appropriately. There are two forms of data that can be integrated into the data mining software. The class labels for the data can either be **balanced or imbalanced**. A balanced class label refers to equal instances for various classes. For example, if the yes in the data is 50 and the no in the data is 45, then we can say the class is balanced. An imbalanced class label is the opposite. The disadvantage of having an imbalanced data is that the classifier is can be biased towards the class with more instances [32, 33, 34].

It is important to point out that both balanced and imbalanced data can be used in data mining or ML task. For example, we can expect an imbalanced dataset when building models for fraud detection. In fact, the data for fraud transaction is extremely unbalanced because only few transactions are considered as fraud. Despite this, it is still not good practice to train your model on and imbalanced data. Another example by an author is a data on dataset from breast cancer. Assume a model is trained on 20 malignant samples and 88 benign samples, this model can gain a high accuracy and predict benign for all the patients. However, there are few methods and algorithms that are commonly used to work with imbalanced data in data mining and ML. Some of these methods include under-sampling and over-sampling [35, 33, 32, 34].

2.5.1 Under-Sampling the Majority Class

Under sampling of data simply refers to the process of randomly choosing from the majority class, a subset of samples. These samples would equal the number coming in from the other class. If we consider our example
on the breast cancer from section 2.5, we will randomly take 20 from the benign samples. This will reduce the amount of data from the majority class and can run the risk of losing critical data that might be needed to train the classifier (see Figure 7).

![Diagram of undersampling process](image)

**Figure 7: Showing an Under-sampling process done on the Majority class using data from rain [35].**

**Figure 7** illustrates a typical under-sampling method performed on the dataset. The No heavy-rain class is much more than the Heavy-rain. The under-sampling technique balances both classes and data from the majority class has been reduced. WEKA, discussed later, offers a distribution spread algorithm for random under sampling. However, for the scope of this project, we shall consider an over-sampling algorithm also offered by WEKA.

### 2.5.2 Over-sampling the Minority Class

Over-sampling method is applied to the class with much fewer instances than the other class. Rather than reducing the instances in the class, over-sampling duplicates samples from the minority class. An example of this would be taken from Figure 7 where the minority class becomes as much as the majority class. The advantage of this is that critical information would be retained unlike in the case of under-sampling. However, the model stands the risk of being overfit because there is likely to be duplicates of the same sample [34].

**Overfitting** refers to the phenomenon where the model memorizes the training data. The best practice to work with these sorts of problems faced from over sampling and under-sampling is to perform **cross-validation** when building the classifier model [36]. There are lots of algorithms that can be used to performing over-sampling but the most common one, also offered by WEKA, is the Synthetic Minority Over-Sampling Techniques (SMOTE).

#### 2.5.2.1 SMOTE (Synthetic Minority Over Sampling Technique)

Over-sampling the minority class can be performed either with replacement or by developing “synthetic” (SMOTE) examples. Over-sampling with replacement means that when observing the minority class by removing one sample, that same sample must be replaced before making another observation. This disadvantage of this activity is that it leaves room for repeating the same observation when that same sample is replaced [33].

SMOTE was developed to handle the challenges of classifying imbalanced dataset. It abandons the approach of over-sampling with replacement by producing more synthetic examples for over-sampling the minority
class. The extra training that is generated from a feature-space and not the data-space. A minority class can be over-sampled by considered each sample of the minority class. A synthetic example along the line that joins all or some of the minority class’s K-nearest neighbor is then introduced. K-nearest neighbors are randomly selected and it depends on how much oversampling is required. “K” means the number of nearest neighbors. This generation of a synthetic example eliminates greatly the problem of overfitting [33, 37, 38]. The steps involved in SMOTE as stated below:

- **STEP 1**: Obtain the difference between the sample being considered and its nearest neighbor
- **STEP 2**: Take a random number between the range of 0 and 1 and use it to multiply the difference
- **STEP 3**: Take the value from STEP 2 and add it to the sample being considered. This will enable a random point to be selected along the segment of the line for two specific features.

The above steps will cause a more general decision region for the minority class than before. The figure below shows a more detailed algorithm for the SMOTE:

```
O is the original data set
P is the set of positive instances (minority class instances)
For each instance x in P
    Find the k-nearest neighbors (minority class instances) to x in P
    Obtain y by randomizing one from k instances
    difference = x - y
    gap = random number between 0 and 1
    n = x + difference * gap
    Add n to O
End for
```

**Figure 8: The Synthetic Minority Oversampling Technique [38]**

It is important to point out that SMOTE applies the same rate of sampling for all the instances of the minority class. [37] Noted that this is a limitation because the process produces a performance that is sub-optimal. They proposed another algorithm called genetic algorithm-based SMOTE (GASMOTE). This algorithm uses a different rate for sampling. This does not, however, dispute the fact that SMOTE performs well. In fact, [38] combined SMOTE and Complementary Neural Networks (CMTNN) for over-sampling and under sampling respectively. Various literature has also suggested and combined the use of SMOTE to work with imbalanced classes and reduce overfitting [39, 40]. It depends on the task being done, its peculiarity and its difficulty.

One limitation with SMOTE in WEKA is that, when it creates these samples, it arranges them directly at the bottom of the list. Assume all the instances produced by SMOTE had a NO class. All these NO classes will be placed at the bottom of the list like in the table below:

<table>
<thead>
<tr>
<th>Initial Value</th>
<th>Yes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Value</td>
<td>No</td>
</tr>
<tr>
<td>Initial Value</td>
<td>Yes</td>
</tr>
<tr>
<td>Initial Value</td>
<td>Yes</td>
</tr>
<tr>
<td>Initial Value</td>
<td>Yes</td>
</tr>
<tr>
<td>SMOTE VALUE</td>
<td>NO</td>
</tr>
<tr>
<td>SMOTE VALUE</td>
<td>NO</td>
</tr>
<tr>
<td>SMOTE VALUE</td>
<td>NO</td>
</tr>
<tr>
<td>SMOTE VALUE</td>
<td>NO</td>
</tr>
</tbody>
</table>

**Table 1: Showing the New Data Generated After SMOTE was Applied**
This can cause overfitting because, when training your classifier using Cross-validation (discussed in section 2.8), it picks a lot of NO instances as part of its fold. This does not give the classifier enough mixed scenarios of YES and NO to learn from. WEKA offers an Unsupervised Instance Randomize algorithm that shuffles the instances to give the data a better mix. A combination of this algorithm with SMOTE gives the final and much better data in the table below:

<table>
<thead>
<tr>
<th>SMOTE VALUE</th>
<th>NO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Value</td>
<td>No</td>
</tr>
<tr>
<td>Initial Value</td>
<td>Yes</td>
</tr>
<tr>
<td>Initial Value</td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SMOTE VALUE</th>
<th>NO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Value</td>
<td>Yes</td>
</tr>
<tr>
<td>Initial Value</td>
<td>Yes</td>
</tr>
<tr>
<td>Initial Value</td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SMOTE VALUE</th>
<th>NO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Value</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 2: Showing the New SMOTE Data that has Been Randomized

2.6 Data Pre-processing: Dimensionality Reduction

Data pre-processing is the process of eliminating features that are irrelevant and causing the classifier to have a bad performance. It is an iterative process because the data must be cleaned continually to reduce noise. A noisy data can lead to a poor model. Dimensionality reduction techniques like feature selection to eliminate noise are performed [20]. This is a very important stage and researchers like [41] have argued the need to perform this task before data can be fed into the system. It is important to point out that the choice of algorithms to use in this stage is very important. The algorithms might eliminate important features that might influence an outcome [42]. After the data is cleaned and balanced, the miner can then integrate the data and perform further pre-processing to reduce dimensionality in the data set.

2.6.1 Dimensionality and Dimensionality Reduction

One of the issue with data is that the number of features can be so numerous and lead to high dimensionality. Dimensionality is simply the number of attributes or features within your data set [43]. High dimensionality can affect the prediction accuracy of our models if not handled properly [44]. Dimensionality reduction (feature reduction) is the process of simplifying the dataset so that computational prediction becomes easier and more accurate [41, 43]. The most common way is using Feature Reduction techniques [45]. A Feature (attribute) in ML is simply the consideration of an individual property or aspect of a data set. This property is usually measurable [46, 44].

There are many feature reduction techniques used for reduction. The most common ones used by researchers like [45, 20] for classification models include feature extraction and feature subset selection. There are also many approaches to perform feature extraction but that would not be covered in the scope of this report. Feature subset selection techniques are commonly used by EDM miners and will be used in this report.
2.6.1.1 Feature Subset Selection Process in EDM

Feature selection is part of the pre-processing steps used in EDM to efficiently improve the performance of the mined data. It is the process of sub-setting your set of attributes i.e. removing the irrelevant attributes. The processes in feature selection do not however, alter the structure of your original data [45]. This means that you can use smaller amount of information to represent your data. The features that are dropped might just be noise to your data.

For Example, Consider a set of attributes $A_1 = \{\text{Age, Name, Number, Address}\}$. Assume the prediction accuracy of the set of data $A_1$ to be 67% accurate. By applying feature subset selection, $\text{New}\ A_1 = \{\text{Age, Name, Number}\}$, the prediction accuracy becomes 85% accurate. We have eliminated “address” from the initial set of attributes to improve the prediction accuracy from 67% to 85%. Some of the Data Mining researchers like [44] have termed this process as an NP-hard (Non-deterministic polynomial-time hard) problem. Wrapper methods, discussed later, uses search algorithms (Heuristic search) to locate feature subsets.

The research put together by [47] highlighted the importance of feature selection methods which include:

- Reduction in dimensionality of feature space to increase the speed of the algorithm and limit the requirements for storing the data.
- Reducing of the features thereby saving the resources required for the next step in the collection of the data.
- Improving the prediction accuracy of the model that is created by Elimination of noisy, redundant or irrelevant data.
- Gaining understanding and knowledge about the data generation process.

The feature selection task is normally done at the data preprocessing stage in EDM. In WEKA, we can easily identify the irrelevant or noisy features and eliminate them. WEKA achieves this by using algorithms to score and validate the importance of a column in the model.

![Figure 9: Process of data cleansing or feature Subset selection [48]](image)

**Figure 9** above shows the simple feature selection procedure that is performed by the ML algorithm. There are Four basic steps involved in Feature Subset selection according and they include [20]:

Page 14 of 78
1. **Subset Generation**: This is a search process for producing candidate feature subsets for evaluation based on the search criteria which has been specified. Reference [48] identified the searching startegies being implemented in this step for creating or generating the subsets.

2. **Subset Evaluation**: This is an iterative way to evaluate and compare the goodness of the new feature subsets with the initial feature subset based on the specified evaluation criteria.

3. **Stopping Criterion**: This is the criteria that must be met for the evaluation process to be complete

4. **Result Validation**: This is when the ML algorithm assigns scores to different columns to identify their relevance

### 2.6.1.2 Feature Subset Selection Algorithms: Comparative Study

Having identified the general process of feature subset selection in the previous section, the algorithms fall under three main categories. They include are Wrappers, Filters, embedded techniques.

- **Filter Algorithms**: These algorithms would extract the features within a data set (Excluding the classifier) without implementing any Learning. It does this by using programs or processes (heuristics) that are based on the data itself. It can **Rank** the features so that the features with a higher rank are the most likely to give a high prediction and a better performing model [44].

- **Wrapper Algorithms**: Wrapper Algorithms implement learning to evaluate the relevance of the features. Wrapper algorithms rank subsets of features based on how accurate the predictor is [42]. The criterion is a feature subset which gives the best prediction [44]. The disadvantage of this algorithm as highlighted by researchers like [20] is that it is computationally expensive whenever big data is applied to it.

- **Embedded techniques**: Reference [45] highlighted that this technique combines the construction of the classifier and steps in feature selection. It uses filter techniques to reduce the “noise”, and wrappers techniques to evaluate the reduced subsets. It will not split the data set into training and testing, but it will still involve feature selection. The feature selection process depends on the criterion which gets generated during the process of learning [42].

Notice how the Features in **Figure 10** where reduced initially by the filter method, Exercising its computational effectiveness. Then being passed out to the wrapper method, utilizing its ability to evaluate features better with fewer data sets.

![Figure 10: Embedded or Hybrid Methods for Feature reduction. Source: [48]](image-url)
2.6.2 Comparing the Feature selection techniques

Table 3 below compares the effectiveness of these features reduction techniques in the data preprocessing stage of Data Mining explained earlier in this chapter.

<table>
<thead>
<tr>
<th>Comparing the three Feature Selection Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filters</td>
</tr>
<tr>
<td><strong>Fast:</strong> This is because they do not require any learning</td>
</tr>
<tr>
<td><strong>Low Model Complexity:</strong> this might result in a not so good fit to the training Example. This is because Filters tend not to capture related information between features and thus may be biased</td>
</tr>
<tr>
<td><strong>Smaller Training data:</strong> Filters work best when the training data size is small because they will have a more accurate prediction</td>
</tr>
<tr>
<td><strong>Variable/Feature Ranking Criteria:</strong> This is the criteria used for variable selection and is applied before classification. This makes it easy for the model to make computations</td>
</tr>
</tbody>
</table>

Table 3: Comparing the different feature selection algorithms [44] [42]

As earlier established, filter algorithms are computationally efficient. They are commonly used in data mining tasks and work well with less features. We shall limit the scope of the dissertation to applying four (4) filter algorithms for our data mining activities. The remaining subsections discusses the filter algorithm we will use in this report.

2.6.3 Filter Algorithms

**Feature ranking** is a method used to score individual features in ML applications [49]. It is a search method available in WEKA data mining software and has a variety of algorithms that can perform this task. Individual features are ranked based on the concepts of some of the filter algorithms the research will implement. Ranking works only with filter algorithms. It is important to point out that there are other search criteria like **best first** and **greedy search** methods. These other methods work with wrappers and embedded techniques, which are not being considered in this report.
2.6.3.1 Information Gain (IG) Feature Evaluation algorithm
This algorithm is a filter algorithm and is part of the supervised type of learning. It is a very strong and reliable filtering technique and it is computationally easy. IG filter algorithm also has a simple interpretation to its structure [50]. It measures how worthy a feature is in relation to its class label [51]. This is calculated using the equation below:

\[ IG(X, Y) = H(x) - H(X|Y) \]

Where \( IG \) is the information gain; \( X \) is the feature and \( Y \) is the class label. \( H \) is the entropy, which is a measure of the uncertainty with respect to a random variable. \( H(X) \) is the entropy of \( X \) and \( H(X|Y) \) is the entropy of \( X \) after observing \( Y \) [50]

\[ H(X) = - \sum_{i} P(x_i) \log_2(P(x_i)) \]
\[ H(X|Y) = - \sum_{j} P(y_j) \sum_{i} P(x_i|y_j) \log_2(P(x_i|y_j)) \]

The highest value in IG is 1 and any feature having a high IG is relevant. In IG, each feature is evaluated on its own and the features are ranked according to their value. IG does not eliminate features like some wrapper algorithms, it just ranks them [49]. The miner is left with the choice of how many features must be removed. Therefore, benchmarking is very important for this purpose [20].

One feature of the IG algorithm which is also a limitation noted by [42, 52] is that it will usually biased towards a decision tree splitting process that has many attributes. It is also not effective with new data. **Information gain ratio** was introduced to balance these limitations.

2.6.3.2 Information Gain Ratio (IGR) or Gain Ratio (GR) Feature Evaluation algorithm
This algorithm as earlier introduced solves the limitation of Information gain algorithm. It is the IG over the essential information of the features. This essential information is usually referred to the intrinsic value \( H(X) \). This is achieved by normalizing the IG with a split information value and would produce a value in the range of 1 to 0 [45, 50]. Suppose we are trying to predict a variable of \( Y \) using \( X \). if the IGR returns a value of 1.0, the \( X \) accurately predicts \( Y \). This calculation of IGR can be done using the equation below [50]:

\[ IGR = \frac{IG}{H(X)} \]

Using an example of a group of students, the assumption is to build a decision tree that uses numerous attributes, one of which is the matriculation number. The aim of the prediction task in the example is to decide which student is best suited for extra-tutorial attention. Since matriculation number is unique to every student, the decision tree using IG will naturally rank the matriculation number attribute highly. This is because it will have a high IG since it is unique to every student. However, the intent is not to decide based on a student’s matriculation number. IGR will solve this issue by considering a large amount of distinct values so that the decision tree is not just biased towards a unique value. So, it takes the size and number of branches into consideration when selecting attributes [46, 47]. The research adopts both algorithms in the analysis along with a few others which will be discussed in the next sections.
2.6.3.3 Correlation-Based Feature (CFS) Evaluation algorithm

This is a filter algorithm that ranks feature subsets based on the correlation heuristic evaluation function that was proposed by [51]. The evaluation function is biased towards attributes, highly correlated with the class label. There will also be a no correlation between the features. This is the basis by which it ranks features. In other words, a feature is best when it correlates more with the class to be predicted [45]. This method will ignore irrelevant features and screen redundant features. This is due to their lower correlation with the class for the former and their high correlation between each other for the later [51, 53].

Reference [45] pointed out the difference between this algorithm and other filter algorithms. Firstly, CFS adopts symmetric uncertainty to calculate the class and correlation of the feature. It also uses a heuristic merit for subsets of the features rather than assessing the features independently. The heuristic evaluation function is expressed using the equation below [47]:

\[ M_s = \frac{kh_{cf}}{\sqrt{k + k(k - 1)h_{ff}}} \]

\( M_s \) is the heuristic merit of the subset S of the features that contain K number of features; \( h_{cf} \) is the mean of the feature class correlation, \( h_{ff} \) is the average of the feature to feature correlation [51].

Despite its advantages, [48] highlighted some of the drawbacks. Firstly, it works best on smaller number of instances. This means applying filter selection to a huge number of features might not give the best results. Secondly, its inability to handle numeric class problems might be a restriction to the kind of ranking results it may give.

2.6.3.4 Relief-F (RF) Feature Evaluation algorithm

The relief algorithm is a univariate model that uses statistical searching methods instead of heuristics searching like the CFS. Also, the time required in the number of features and training instances must be linear irrespective of the class to be predicted. The limitation of the relief algorithm is its inability to work with noisy and incomplete data [53].

The Relief-F algorithm was introduced to tackle the limitations of the relief algorithm highlighted in the paragraph above. It is also a supervised multivariate attribute selection algorithm and is simple and effective in the biological area [53, 50]. It chooses the features that can be distinguished the most amongst the numerous classes. The highest weights are given to the features that can discriminate an instance from the neighbors of the different classes. The instances are chosen iteratively during this process [45]. One major benefit of this algorithm is its scalability to a set of data that has constant increase in dimensionality [48]. Its algorithm is based on the equation below:

\[ SC_R(f_i) = \frac{1}{p} \sum_{t=1}^{p} \left\{ -\frac{1}{m_{xt}} \sum_{x \in H(x_t)} d(f_{t,i} - f_{j,i}) + \sum_{y \neq y_{x_t}} \frac{1}{1 - P(y_{x_t})} \sum_{x \in M(x_t, y)} d(f_{t,i} - f_{j,i}) \right\} \]
Where $y_{x_t}$ is the class label of an instance $x_t$; $P(y)$ is the probability of the instance coming from a class label $y$; $NM(x,y)$ and $NH(x)$ is a set of the nearest points to $x$ either with a different or the same class as $x$. $y \neq y_{x_t}$ is a predefined constant [50].

The advantage of all these algorithms is that they are all good at assessing the relevance of features by ranking them [20]. There are still other filter algorithms which were not discussed here like Chi-square, symmetric uncertainty, fast correlation base and sequential forward selection. Some of these algorithms are available in WEKA but the scope of this dissertation would be limited to the above four. The next section focuses on the classifier models (algorithms) that will be built, based on the features that were retained by the filters.

### 2.7 Data Modeling: Strategies, Techniques and Algorithms in EDM

This is where we carefully revisit the requirements we stated in the business understanding stage to identify which model and testing option works best for our technique [29]. For example, if regression techniques are to be applied, there is a need to choose the best regression algorithm. It is important to point out that most data mining tasks are predictive in nature and thus, use classification algorithms. However, before we dive into the algorithms, the following subsections describes the different tasks EDM offers. It also describes EDM’s place in data science.

#### 2.7.1 EDM and Machine Learning

There is a strong relationship between these two concepts as shown in Figure 3. Machine Learning (ML) has been around since the 1970’s and has since grown to be very popular within numerous industries. It is a discipline that enable computers learn and understand how to identify patterns or trends by studying a set of training examples [54]. This concept is very important because EDM, adopts ML algorithms to work on mined data. Data mining aims to solve problems by analyzing the existing data within a database and discovering hidden patterns in data. ML techniques are used in data mining process to build models that can find and describe the structural patterns of data [52]. To summarize, while data mining is used for knowledge discovery, ML would develop some algorithms to give a computer system the capability to learn on its own. There is no need for any explicit programming.

#### 2.7.2 EDM Learning Strategies

There are a few strategies for learning in ML. Some authors like [54] highlight just two strategies which include supervised and unsupervised learning. Further research in few books like [46] pointed out another type of learning which is reinforcement learning. The strategies for learning based on the researches are discussed and illustrated in Figure 11 below [46, 54]:

- **Supervised Learning:** This is when the training set contains data and the correct class or output for the data set. For example, solving a problem in class with your students and telling them the answer to the problem. Based on the knowledge of how to go about this problem, they should be able to solve other
problems where the answer is not known yet. Classification and Regression algorithms fall under this type of learning.

- **Unsupervised Learning:** To understand this strategy, let us take the same students given in the example in supervised learning above. Imagine if the teacher introduced a topic like statistics and instead of solving problems with them, the teacher gives them an assignment. The teacher then asks them to find the answers themselves. So, it involves presenting the system with a data set with no solution. The system is expected to get the answers by itself. Clustering and Dimension Reduction algorithms are good examples.

![Figure 11: Different strategies to learning and the categories of algorithms][54]

Re-enforcement Learning is an additional strategy however, most EDM literature adopts supervised and unsupervised learning. We will dwell more on these two strategies for the data mining tasks.

### 2.7.3 EDM Techniques and Methods

The techniques commonly used in EDM are divided into two and categories, descriptive or predictive, and are illustrated in **Figure 12**.

![Figure 12: Shows the main categories of data mining techniques with some examples][55]

- **Descriptive EDM:** Descriptive techniques are unsupervised learning models which focus more on the intrinsic data structure, data relationships and interconnections [31]. Some examples of the descriptive EDM techniques include clustering and relationship mining (Association rules).
• **Predictive EDM:** Predictive techniques are more for supervised learning explained earlier usually to predict a target value.

Having highlighted this, the scope of this project is limited to the predictive tasks as this is the main objective of our classifiers. This shall be discussed in the next section.

### 2.7.4 Predictive EDM Techniques

Prediction on its own is a very broad topic. Financial institutions can use this technique to predict fraud. A miner can predict if a student is a fast or slow learner, or if a student is most likely to fail or pass his exams [56, 20]. A prediction technique creates a model that will accurately deduce or predict from an entire data set, one specific feature or part of that data. The sets of data that predict the outputs are known as the predictor variables (features). The data that needs to be predicted is known as the predicted variable or dependent variable (class variable). The prediction process is influenced by the other features or parts of the same data set. The models built for prediction are built using classification, regression and time-series techniques. The two main techniques are classification techniques and the regression techniques however, most EDM tasks prefer to build classification algorithms. In a classification task, the aim is simply to use historical data to predict a target attribute of an unlabeled example [31, 57, 3]. The following subsections would briefly describe the two main techniques.

#### 2.7.4.1 Classification Techniques

This is an instance of supervised learning whereby the inputs are a dataset were each data has a class mapped to it. The computer would then be able to learn how to classify any new data that comes in [46, 54]. For example, imagine a data set of students’ performances in different courses. If there is an indication of which of the students are on a first class, second class and so on, the computer would be able to classify students based on their grade. Some of the classification algorithms include, Naïve Bayes, Adaptive Networks and support Vector Machine (SVM) [58].

#### 2.7.4.2 Regression Techniques

This is another instance of supervised learning where there is a desire to have a constant response value unlike classification. In order words, the intended output would contain either a single or more continuous variable. The example given by [46] sufficiently explains a type of regression problem. Consider a chemical manufacturing process that contains a concentration of pressure, reactants and temperature inputs. We can predict the yield [36]. One of the oldest type of regression algorithms is the linear regression but it will not be discussed in this report.

### 2.7.5 Predictive EDM Algorithms (Classification Algorithms)

The task in this dissertation is a predictive task and the algorithms we will use would be classification algorithms. WEKA offers a range of classifier algorithms however we shall consider just four of them.
2.7.5.1 Naïve Bayes classifier algorithm

Naïve Bayes is a very simple probabilistic algorithm and works well with the assumption that for a given class, all the attributes are independent [47]. It is a naïve classifier in the sense that all the features are treated independently of each other [42]. Learning of an optimal structure of a Bayesian network is an NP-hard problem. Naïve Bayes classifiers can avoid complexity of the learning structure which is not easily traceable in some other Bayes networks. This is because each feature has one parent which is the class [59]. This is illustrated in Figure 13 below where C is the class label and \( X_n \) is the predictor.

![Figure 13: Shows the single class relationship between independent variables in the Naïve Bayes algorithm [59]](image)

The above diagram can also be represented in the equation below, where \( X_1 \) to \( X_n \) are independent of each other with the given parent attribute or class C:

\[
P(C|X) = \frac{P(C) \prod_{i=1}^{n} P(X_i|C)}{P(X)}
\]

One advantage of Naïve Bayes is that even within a set of explicit assumptions, it assures optimal induction [47]. However, these assumptions of a single class can be violated given some real-world problems. Another drawback noted by [60] is that it often assigns inaccurate probability estimations. Therefore, it only works best in situations that require a categorical output. Despite these drawbacks, it is effective, easy to implement, has a simple structure and is fast. In addition, it works very well with data that has high dimensionality. Naïve Bayes algorithm works with both multiclass and binary classification problems as well as numeric prediction or regression problems. When using it with regression problems, Naïve base would give accurate results where the independence assumptions hold [60, 52, 46].

2.7.5.2 J48 (C4.5) Pruned and Unpruned Decision Tree Algorithm

Decision tree algorithms seek to understand the way attribute-vectors behave towards \( n \) number of instance [61]. C4.5 algorithm is performed using the open source J48 and uses the concept of entropy to build the decision tree [62, 52]. It makes decisions at each node of the tree by choosing the most effective attribute. The attribute that separates its sample subset effectively, and places them into one class or the other is considered. Amongst these classes, the one with the highest information gain is the one that is chosen. The steps performed by J48 algorithms can be seen below [61]:

1. During decision, if the instance or attribute belongs to the same class of the leaf of the tree, then the leaf will be labeled with the same class and then returned
2. Every attribute is tested and the information gained is calculated based on this test.
3. The information Gain that is the highest is selected for branching
The entropy, measure of data disorder, and information gain is given by the formula below while Figure 14 below illustrates a simple decision tree built using the J48 algorithm:

\[
\text{Entropy (\bar{y})} = - \sum_{j=1}^{n} \frac{|y_i|}{|\bar{y}|} \log \left( \frac{|y_i|}{|\bar{y}|} \right)
\]

\[
\text{Entropy (j|\bar{y})} = \frac{|y_i|}{|\bar{y}|} \log \left( \frac{|y_i|}{|\bar{y}|} \right)
\]

\[
\text{Gain} = \text{Gain (\bar{y},j)} \text{ Entropy } \bar{y} - \text{Entropy (j|\bar{y})}
\]

Figure 14: Shows the decision process of a decision tree algorithm like J48 [63]

2.7.5.3 Random Forest

Random decision forest is a nonparametric algorithm that is derived from regression and classification techniques. It is comprised of a combination of multiple trees all generated using samples from a bootstrap. This then uses a third of the total samples that were left out, for validation. It uses a subset of the predictors or decision trees selected at random to make decisions at each node. The outcome is the average of the results gotten from all the decision trees within the forest. This is for classification problems. It can also output the class that is the average or mean prediction of the other individual trees for regression tasks [52, 64]. Random forest is related to adaptive nearest neighbor algorithms in that they both consider the neighborhood of the new point they are trying to predict [65].

The steps involved in constructing a random forest from multiple decision trees outlined by [66] include:

1. Apply \( K \) number of iteration of bagging (from the instances or records) to create \( K \) number of trees. **Bagging or bootstrap aggregation** [64] is a standard for a training set \( D \) having a size of \( n \), where \( n \) is the number of instances in the original data set of \( D \). bootstrapping is also called **resampling**, which is refers to an estimate gotten from data subsets by randomly selecting data points with or without replacements also called bootstrapping

2. Apply attribute bagging (Random Subspace Creation) to the attribute for each of the \( K \) sample training set to create **multiple decision trees**. The decision tree is learnt in such a way that the variable from a new node is the best variable, having the least **Relative absolute error (see section 2.9.1)** amongst the random subspace.
3. Repeating the first and second step would give us K number of trees in the random Forest

4. For classification tasks within the forest, for each decision trees, simply make a prediction of the class for each instance. The class that was predicted more often by each tree becomes the class

![Random Forest Simplified](image)

**Figure 15: Random Forest Based Classification Algorithm [67]**

Figure 15 illustrates the decision-making process within a random forest by multiple decision trees. We can see the importance of the random forest decision tree algorithm explained by [68]. It is used to reduce overfitting. The accuracy it produces is reasonable and it uses more features in data analysis. However, one limitation it that it is not a good algorithm, in a general case, with new data. Secondly, if the features are categorical and have multiple class labels, then random forest will be biased towards the instance that has more labels [65]. The researcher will observe this when evaluating the models.

### 2.7.5.4 Logistic Regression

Linear regression models are good for building classifiers with numeric classes. This classification can be done by either using linear or non-linear regression techniques. However, linear regression problems have certain limitations even if they predict an outcome value [69]. Firstly, it is a binary classifier for non-class and class instances. It assigns these numbers to the instances depending on the relationship with the class [52]. The second limitation is that it assumes that errors in a least-square regression are distributed normally using the same standard deviation. According to [52, 46], this assumption is violated in a classification problem because the instances only take values of 0 and 1.

The above two limitations is fortunately avoidable with **logistic regression**. Logistic regression is similar to linear regression except for its low optimality in fitting data directly to a line. It is also works well with multiclass labels. The example by [70] demonstrates this by trying to predict if a student passed a course using study hours as the predictor instances. The resulting graph could not be fit linearly using logistic regression. Logistic regression evaluates an individual attribute’s unique contribution to the predicted class. So, there will not be any constraint between the values of 0 and 1. Therefore, there is no need to worry about any positive (+) and negative (−) infinity values. The logit scale used in logistic regression transforms the linear regression using the natural logarithm of the “odd” of being in one of the predicted class [71]. **Figure 16** was the resulting graph that solved the issue.
Figure 16: Illustrates a logistic distribution by categorizing the predicted class [70]

It is an iterative identification of the best combination of linear variables that have the highest likelihood of predicting the class. The greater the odds, the better the outcome. Other regression algorithms include Decision trees, Bayes network and Fuzzy classifications. The general knowledge about regression and the above example is enough for the scope of this dissertation.

2.8 Training and Testing the Classifier Model

Having identified the algorithms or classifiers to be modeled (see section 2.7.5) we must set up a plan on how to train our classifiers. The data will be split into training and testing set to train and test our classifier models. WEKA offers three testing options but the most commonly used is the cross-validation and is relevant to this work [72].

The goal of a data mining task is to find information in large chunks of data. In a real-world scenario, the data available is not always large. There are other conventional means apart from cross-validation such as, splitting the data into training and testing sets. However, most data miners do not have the luxury of data to do this. Cross validation is a testing option in WEKA where the true accuracy of a model is estimated. In other words, it accesses how the model will generalize with data that is unseen. It works by splitting the sample into a training and testing set. By doing this, there is no significant loss of the modeling capability. Cross-validation works well with small amounts of data and reduce the risk of model overfitting. There are different types of cross validation but the most common one used in most data mining literature is the K-fold cross-validation [57, 73].

In a K-fold cross-validation testing option, the original data will be partitioned in to an equal size of subsamples. K refers to the subsample size. One of the subsamples is used for validation while the other k-1 will be used for training. This process of cross-validation is iterated k-times and each k subsample will be used to validate the data just once. It is the average of the k results that is used to produce one single estimation. One major advantage is that all the k observations are used for both training and testing [74, 75]. Let us consider an example using the figure below as an example.
Figure 17: 5-fold cross validation testing [75]

Figure 17 shows a test set of $k=5$-fold cross validation. The data is divided into 5 equal subsets and 5 equal field (parts). There would be four (4) training sets that will be used to train the model and the remaining one (1) will be used for testing. At the end, the average of all the test results will be taken and will be the prediction statistics for the model.

2.9 Classifier Model Evaluation

This is the stage where you access the performance of your models according to some evaluation metrics. In other words, it is a method to determine if your model is representing the truth based on your instances [76]. The most common evaluation metrics in datamining will be discussed in the next subsections. The model evaluation will be used in this report to evaluate the filters algorithms that will be used to reduce the features.

2.9.1 Confusion Matrix:

A confusion matrix is used to see how your classification model has performed after training (see section 2.8). It is usually expressed as an $X$ by $X$ matrix, where $X$ is the number of classes that is being predicted [77]. So many evaluation criteria such as the AUC, F-measure and prediction accuracy are drawn out from the confusion matrix. We shall use a binary class of yes and no to explain the components of a confusion matrix for simplicity. Consider the confusion matrix in Table 4 which has a binary (2 classes) representation of yes and no. Assume a model was created to predict if an individual has had breakfast or not. The result in a confusion metrics is shown below:

<table>
<thead>
<tr>
<th>No of instances= 100</th>
<th>Predicted: No</th>
<th>Predicted: Yes</th>
<th>Actual No Total= 30</th>
<th>Actual Yes Total = 70</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual: No</td>
<td>TN= 20</td>
<td>FP= 10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Actual: Yes</td>
<td>FN= 15</td>
<td>TP= 55</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Predicted No total = 35</td>
<td>Predicted Yes Total = 65</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4: Illustrates a confusion Metrics for evaluating a prediction model for people who had breakfast

- **True Positive (TP):** This is a case where the predicted class is *Yes* and the actual class is *Yes*. Referring to our table above, we predicted *yes* and they had eaten.

- **True Negative (TN):** This is when the predicted class is *NO* and the actual class is *NO* [14]. In other words, the model predicted that they had not eaten and they truly had not eaten breakfast.
• **False Positive (FP):** In this case, the predicted value or class is Yes and the actual value is No. This means from our example, we predicted yes but they had not actually eaten.

• **False Negative (FN):** This is when the predicted class is No and the actual class is Yes. We predicted that they had not eaten, but they have eaten.

The following can be deduced from our understanding of Table 4

➢ 30 people did not have breakfast, where as 70 people had breakfast.
➢ The model predicted that 35 people did not have breakfast while 65 people had breakfast.

### 2.9.2 Common EDM Evaluation Metrics derived from The Confusion Matrix

The example and terminologies in section 2.9.1 can be used to define some common EDM evaluation metrics necessary to understand the behavior of our model [52, 77, 14].

a. **Prediction Accuracy:** This rate tells us how accurate the prediction of our model is, based on what it predicted “Yes” and “No” and was truly “Yes” and “No”. This is expressed as:

\[
\text{Accuracy} = \frac{TP + TN}{\text{Total No Of Instances}}
\]

b. **Relative Absolute Error (RAE):** The Relative Square Error (RSE) is used to compare models whose errors are measured in different units. The RAE is like the RSE however, the absolute values of the predicted and actual values are considered [78]. WEKA expresses this error as a percentage. This can be mathematically illustrated in the equation below:

\[
RAE = \frac{\sum_{i=1}^{n}|p_i - a_i|}{\sum_{i=1}^{n}|a - a_i|}, \text{ where } p_i = \text{predicted value per instance}, a_i = \text{actual value per instance and } \bar{a} = \text{average of all the actual values}
\]

c. **Sensitivity, Recall or True Positive Rate:** This is the rate at which our model predicts “Yes” when it is truly “Yes”. In other words, how sensitive Is the model to the “Yes” prediction.

\[
\text{TP Rate} = \frac{TP}{\text{Total No Of Actual Yes}}
\]

d. **False Positive rate:** This is the at which our model predicts “Yes” when it is truly “No”.

\[
\text{FP Rate} = \frac{FP}{\text{Total No Of Actual No}}
\]

e. **Specificity:** This is how often the model predict “no” when it is truly “no”. the opposite of sensitivity

\[
\text{Specificity} = \frac{TN}{\text{Total No Of Actual No}}
\]

f. **Precision:** how often is the model correct whenever it predicts yes

\[
\text{Precision} = \frac{TP}{\text{Total No Of Actual yes}}
\]

g. **F-measure:** This is the weighted average between the recall and precision. In other words, it measures the accuracy of a test where 1 is the best value and 0 is the worst value [52].

\[
\text{F-measure} = 2 \times \frac{1}{\frac{1}{\text{recall}} + \frac{1}{\text{precision}}}
\]
h. **Area Under the Curve (AUC):** The AUC is gotten from the plot of the Receiver Operator Characteristics Curve (ROC). The ROC is a graph that that shows the performance of the classifier model that was built over an established threshold. **TP-Rate** is going to be the (y-axis) and **FP Rate** is the (x-axis). This **AUC** is the quantity or area under the curve of the ROC graph. We use this measure to quantify how our classifier is performing. It should ideally be 1.0 [79]. Reference [52] added that it is the best measurement to use to select the classifier. Any classifier having an AUC of about 0.8 is a good classifier. This can be illustrated in an extract from weka in the figure below:

![Figure 18: Depicts the Area Under the ROC Curve (source: stack overflow)](image)

Model evaluation is very important both in ML and data mining activities and the choice of the evaluation criteria to be used must be carefully considered. In other words, it is not just good practice to depend on just the prediction accuracy of your model. Depending on your tasks, other criteria must be considered. The most commonly used metrics in EDM literature from are the AUC and F-measure. For example, [14] used Confusion Metrics and Geometric mean to evaluate the performance of 5 classifier models. However, their proposed algorithm (ICR M2) performed much better than the other EDM algorithms based on that measurement. Another example is from the work done by [13], where they evaluated decision tree algorithms for building models to perform recruitment analysis using precision, recall and F-measure. Random tree was preferred to J48 based on its score from F-measure. It is clear to see that more than one criteria can be considered. It is possible you want more recall than precision for your task.

This report will combined the use of **prediction accuracy**, **AUC** and **F-measure** to evaluate our filter algorithms and classifier models. We shall occasionally consider a lower **RAE** to break a tie and make our decision between two closely matched models.

### 2.10 Model Deployment

This is basically the last stage of the EDM process. Having successfully satisfied all requirements in the evaluation phase, the model can be deployed on new data set. This is an iterative process and must continually be re-assessed to maintain a good performance [29]. The process must be revisited and models must be constantly evaluated to access their performance.
2.11 Waikato Environment for Knowledge Analysis - WEKA

In previous sections, we have been able to outline the processes and techniques used in EDM. It is important to talk about the platform where this work would be achieved. There are a lot of software that can perform data mining tasks. Some examples include R, RapidMiner, Python, WEKA and Hadoop. Tools like R and WEKA are opensource, while RapidMiner explained by [52] is for commercial purposes.

WEKA is a data mining software that contains ML algorithms for handling EDM activities. It was produced by the Department of computer science at the university of Waikato and was named after a bird found only in the New Zealand [72]. Its input file extension is an arff file. WEKA is written in java, and thus can allow java file extension within its system to manipulate or optimize its ML capabilities [52]. It is also an open source tool and can be downloaded directly from the website. It can run on windows, Linux and most operating systems, making it easier to implement. The algorithms present in WEKA can be used directly in a java program [51]. Some of the tasks it can perform include data pre-processing, building classifier models, clustering, association rules, Feature selection and data visualization (see Appendix 2).

Figure 19: WEKA Start Up GUI

Figure 19 shows the various interphases supported by WEKA like Explorer, for beginners and workbench for much more in-depth analysis. This research will use WEKA to select the features, build the classifiers and perform the evaluations and data analysis on the data set. All the algorithms and techniques discussed in this literature are supported by WEKA [51]. This Software is more than just any open source tool, it has been used to do in-depth data mining analysis and build classifiers that are presently in use. Researchers like [13] have all used WEKA in their various tasks. It is also being used by students to understand and learn the EDM process necessary to achieve a task [72].

2.12 Chapter Summary

Over the course of this chapter, we have introduced the concept of EDM and how it is related to ML. In fact, we have shown how dependent the processes of EDM is on ML algorithms. The problem situation was explained in detail and the importance of EDM in the Educational sector was highlighted. The features that are perceived to affect students taking programming courses was discussed. Based on the past literature, a questionnaire has been constructed to identify and evaluate those features (See Appendix 1).
The classifier models to be built include Logistic Regression (LR), Naïve Bayes (NB), Random Forest (RF) and J48. **LR** adjusts well to noise and overfitting can be avoided. **J48 and NB** are very reliable in decision making because they use statistical measures to make decisions. **RF** less prone to overfitting and is more popular because it is easy to implement and works well with multiclass labels [65, 27, 54]

The concept of dimensionality was explained and how high dimensionality can produce a poor prediction from the classifier was equally highlighted. The techniques to reduce dimensionality was highlighted and four techniques will be used in this research. They include Information gain, Information gain ratio, correlation-based and relief-f algorithms where all selected. This was due to their statistical accuracy and common use in EDM literature. Also, some of the algorithms work well with multi class labels and can handle a reasonable amount of data attributes. [42, 53]

Finally, introduced WEKA which is the software that would be used for this dissertation. It is a very common and easy to use to and has been used in numerous EDM literature
CHAPTER 3: METHODOLOGY

3.1 Work Flow Analysis

The methodology that would be used in this report is the CRISP-DM discussed in section 1.2.3. It will help us in answering the research questions that guide our objectives. The flow chart below depicts the project stages from the point of gathering the data till the discussion of the result. The remaining sections would describe key aspects based on this diagram.

![Flow Chart Illustrating the entire methodology for the dissertation](image)

3.2 Requirements

The requirements for the dissertation include data gathering using questionnaires, using a data mining tool (WEKA) to choose the algorithms, build the models on the data subset and the evaluate the results. The following subsections describe how we intend to answer the research questions asked in chapter 1.

3.2.1 Data Gathering and Extraction using Questionnaire

The student data set was collected using a questionnaire. The data has been discretized to improve the prediction accuracy of our classifiers [14]. Another benefit of a discretized data is that it makes our model simpler while reducing noise [80]. The questionnaire is divided into eight (8) parts, each collecting its own unique group of features. All these features answer question 1 of our research. The figure below illustrates the structure of the student data set. A more detailed look at the questionnaire is available in Appendix 1.
Figure 21 below consists of four (4) columns and fifty-three (53) features in total. One of these features (CL53) will be the class or predicted feature. The first column describes the name of the feature we have extracted. The second column indicates the feature code and would be used to represent the individual features in our WEKA arff file. The remaining column contains the discretized and non-discretized equivalents to the answers in the questionnaire. The following points explains the features in each category.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Feature Code</th>
<th>Score</th>
<th>Feature</th>
<th>Feature Code</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Profile and Experience</strong></td>
<td></td>
<td></td>
<td><strong>University Resources</strong></td>
<td>ES28</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>YearOfStudy</strong></td>
<td>PE1</td>
<td>5 4 3 2 1</td>
<td><strong>Class Size</strong></td>
<td>ES29</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Age</strong></td>
<td>PE2</td>
<td>5 4 3 2 1</td>
<td><strong>Lecture Notes</strong></td>
<td>ES30</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Sex</strong></td>
<td>PE3</td>
<td>5 4</td>
<td><strong>Lecturers Mode of Teaching</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Student Type</strong></td>
<td>PE4</td>
<td>5 4</td>
<td><strong>Lecture Slides</strong></td>
<td>ES31</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Health</strong></td>
<td>PE5</td>
<td>5 4 3 2 1</td>
<td><strong>Demos</strong></td>
<td>ES32</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Finance</strong></td>
<td>PE6</td>
<td>5 4 3 2 1</td>
<td><strong>UL</strong></td>
<td>ES33</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>work</strong></td>
<td>PE7</td>
<td>5 4</td>
<td><strong>Class Quiz</strong></td>
<td>ES34</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Work2</strong></td>
<td>PE8</td>
<td>5 4</td>
<td><strong>Class Exercise</strong></td>
<td>ES35</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Pre-College Core</strong></td>
<td>PE9</td>
<td>5 4</td>
<td><strong>Lecture Intensity</strong></td>
<td>ES36</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Emotional Balance</strong></td>
<td>PE10</td>
<td>5 4 3 2 1</td>
<td><strong>Revision</strong></td>
<td>ES37</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Previous Academic Background</strong></td>
<td>PAB11</td>
<td>5 4 3 2 1</td>
<td><strong>Programing Concepts</strong></td>
<td>PAB12</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Background knowledge</strong></td>
<td>PAB13</td>
<td>5 4 3 2 1</td>
<td><strong>Constructors</strong></td>
<td>PC50</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Social Status</strong></td>
<td>PAB14</td>
<td>5 4 3 2 1</td>
<td><strong>Variable Declaration</strong></td>
<td>PC60</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Previous Experiences</strong></td>
<td>EX14</td>
<td>5 4 3 2 1</td>
<td><strong>Conditional Operators</strong></td>
<td>PC41</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>C</strong></td>
<td>EX15</td>
<td>5 4 3 2 1</td>
<td><strong>Loop Operations</strong></td>
<td>PC42</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Java</strong></td>
<td>EX16</td>
<td>5 4 3 2 1</td>
<td><strong>Pointers References</strong></td>
<td>PC43</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>JavaScript</strong></td>
<td>EX17</td>
<td>5 4 3 2 1</td>
<td><strong>Collections</strong></td>
<td>PC44</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Python</strong></td>
<td>EX18</td>
<td>5 4 3 2 1</td>
<td><strong>IO Operations</strong></td>
<td>PC45</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>PresentProg</strong></td>
<td>EX19</td>
<td>5 4 3 2 1</td>
<td><strong>Error Handling</strong></td>
<td>PC46</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Students Learning and Studying Mode</strong></td>
<td>L520</td>
<td>5 4 3 2 1</td>
<td><strong>Libraries</strong></td>
<td>PC47</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Material Study</strong></td>
<td>L521</td>
<td>5 4 3 2 1</td>
<td><strong>String Handling</strong></td>
<td>PC48</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Video Tutorial</strong></td>
<td>L522</td>
<td>5 4 3 2 1</td>
<td><strong>Memory Storage</strong></td>
<td>PC49</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Group Meetings</strong></td>
<td>L523</td>
<td>5 4 3 2 1</td>
<td><strong>Extracurricular Activities</strong></td>
<td>L524</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Peer Tutorials</strong></td>
<td>L525</td>
<td>5 4 3 2 1</td>
<td><strong>External Activities</strong></td>
<td>EC50</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Online Tutorials</strong></td>
<td>L526</td>
<td>5 4 3 2 1</td>
<td><strong>Personal Practice</strong></td>
<td>EC51</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Attendance</strong></td>
<td>L527</td>
<td>5 4 3 2 1</td>
<td><strong>Design Confidence</strong></td>
<td>EC52</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Instructor Guidance After Class</strong></td>
<td>L528</td>
<td>5 4 3 2 1</td>
<td><strong>Performance</strong></td>
<td>CL33</td>
<td>5 4 3 2 1</td>
</tr>
<tr>
<td><strong>Lab Attendance</strong></td>
<td>L529</td>
<td>5 4 3 2 1</td>
<td><strong>poor Average</strong></td>
<td>Good</td>
<td></td>
</tr>
</tbody>
</table>

---

**Figure 21: Features from Students Data Set to be extracted from the questionnaire**

- **Profile and Experience**: These features identify the impact that personal attributes and experiences have on a student’s performance. For example, the present year of study, age, sex and health.

- **Previous Academic Background**: These features access the impact previous academic background has on a student. For example, a science student might do better than an arts student.

- **Previous Programming Experiences Before Coming To school**: These features access the impact that past programming experience has on the student presently. These were carefully selected from literature discussed and cited in section 2.2.

- **Students Learning and Studying Mode**: This category of features assesses the learning strategy of the student. For example, the type of material, online or hardcopy, a student uses to study. Furthermore, the analysis will reveal if students who do not attend lectures perform much better. The student-teacher communication strategy will be considered as a feature that can influence the performance of a student.
• **Lecturers’ Mode of Teaching:** Reference [17] used data mining techniques to predict the performance of lecturers. Features like the length of the lecture, on-the-spot quiz and so on, is investigated as a likely influence.

• **Programming Concepts:** Most of the features in this category are based on the works from [7, 4, 81]. However, the research goes deeper to consider more programming specific questions. Major programming concepts taught to new programmers like memory utilization by the application will be analyzed.

• **Extracurricular Activities:** This category considers the passion and personal effort the student makes in learning programming. For example, attending programming conferences and personal practice time.

• **Predicted Class:** This category hosts the multiclass labels which would be predicted based on the values from other categories. The students will rate their present performance in school. The students answer to the question will be divided into three groups namely: poor, average and good.

### 3.2.2 Data Extraction and Conversion

WEKA supports an interface known as the knowledge flow where a data mining process can be constructed from data integration to model evaluation. This process is illustrated in **Figure 22** and was created in WEKA knowledge flow and was used in this report.

![Figure 22: WEKA Knowledge Flow Interface for Data Mining Tasks](image)

WEKA is an open source software tool and uses an *arff* file extension as its input. A sample of the *arff* file with discretized features and the multiclass label can be seen in **Appendix 3**. The data extracted initially from section 3.2.1 will be replaced with the feature code described in **Figure 21** and integrated into WEKA through the *arff* loader in **Figure 22**. Note that there are other data gathering tools that can be used to extract data from our questionnaire. However, due to the small amount of our data, an excel sheet was suitable for this task. Data Preprocessing and Integration. **Appendix 4** depicts the interface where the data was integrated into WEKA. It is clear to see that WEKA can do some basic analysis of the data such as viewing the current weight of features. It is also from this interface that dimensionality reduction can be done by eliminating features.
3.2.3 Computational Analysis and Model Evaluation

An objective of this report is to see which of the models performs best on an optimal set of features. A walk through to this conclusion would consequently answer the remaining of our research questions. The following subsections would give us a more detailed understanding of how this would be achieved using WEKA.

3.2.3.1 Building the Classifiers on Imbalanced and Balanced Class Label Without Feature Reduction

The classifier models were built using the Naïve Bayes, Random Forest, J48 and Logistic regression algorithms provided by WEKA. WEKA’s 10-fold cross-validation test (see section 2.8) options will be used to train the model to prevent model overfitting. A sample of this process can be illustrated in Appendix 5. The observations from the F-measure and AUC value for each algorithm will be recorded and analyzed. As a reminder, the initial data provided has an imbalanced class label. Therefore, this section would be performed with first the imbalanced data and then a balanced data. This will answer Questions 2 and 3 of our objectives. The data would then be balanced by applying SMOTE (see section 2.5.2.1) offered by WEKA.

- **Success Criteria:** An improved F-measure and AUC value is expected after the data is balanced when compared to its initial value. This shows that noise can make a classifier biased towards the majority class. Preferably, and F-measure score of above 0.5 indicates good precision and recall values from the confusion metrics. Alternatively, and AUC score of above 0.6 indicates better true positive rate (TP rate) than false negative rate (FN rate). The figure below is a sample evaluation screen in WEKA.

![Figure 23: Evaluation Metrics Offered by WEKA Datamining Software](image)

3.2.3.2 Building the Classifiers by Eliminating Features using Filter Techniques

The feature selection process is an iterative process for this research as illustrated in Figure 20. We have adopted this technique from the research performed by [20]. We shall apply Information Gain (IG), Correlation-Based (CB), Relief-F (RF) and Information Gain Ratio (IGR) filter-feature selection techniques to
the balanced data set. Based on our results, we shall successfully answer questions 4 and 5 from our research objectives.

The ranking search method offered by WEKA will be used to rank the individual features. See sample extract from Appendix 6. We have decided to build each classifies by eliminating 26 features for this project. The intention is to see the performance of the classifiers while retaining as much features as possible. WEKA offers an attribute selected classifier where we can make these configurations as can be seen in the figure below.

![Attribute Selected Classifier Using IG to Rank the Attributes](image)

**Figure 24: Attribute Selected Classifier Using IG to Rank the Attributes**

This classifier is a Meta classifier that evaluates the worth of an attribute first, before building any classifier on that attribute. This is perfect for this report because we can decide how many features to keep for each iteration. We can see from the sample screenshot above, that we have chosen to keep 26 attributes. We can also observe that those 26 attributes are based on their rank from IG. Then J48 would be applied on them. We shall plot a chart for each of our classifier to show the effects of the F-measure and AUC value for each feature being removed.

- **Success Criteria 1**: A good answer to this question is based on the highest performing values from our measures for all the classifier when the filter techniques are implemented. **RAE** will be used to break a tie between measures as we shall see. A lower RAE indicates a better classifier. So, feature subset selection should improve the values recorded from the balanced dataset in question 3, answering question 4.

- **Success Criteria 2**: The results from this experiment will also highlight the highest performing subset of features. All the classifiers will be applied on the various feature subsets and the group of classifiers that have the highest average performance would be chosen. Furthermore, the classifier with the highest prediction accuracy within that chosen subset would be considered the best. This will answer question 5 comfortably.
3.2.3.3 Analyzing the impact of Feature Reduction on the Balanced Dataset
At this point, we have successfully seen the performance of the features before and after balancing the data. This experiment will compare the performance between the balanced data and feature reduction from the balanced data. The results from this part will give us a suitable answer for question 6. A t-test statistical evaluation will also be performed on the before and after values of all the classifiers. This will show if the change in performance is significant or not. This step will either reject or accept the null hypothesis stated in our project aims and objectives. A sample of the before and after values from iteratively removing our features from one of the classifiers is shown in the Appendix 7.

- **Success Criteria:** A higher value of F-measure and AUC suggest that there has been an improvement while a lower value suggests otherwise. A $p$-value of $< 0.05$ from our t-test suggest that there has been a significant change.

3.2.3.4 Analyzing the impact of feature reduction on new unseen data
The methods we have adopted up on-till now have used 100% of the balanced data set. We have used cross-validation to train and test our data. Since the data available is small, we do not have the enough to split into training and test data. Consequently, we will build the new models from the same classifiers using the features in question 5. The training data will be 90% of the total instances. This retains more instances than attributes in our data set and prevent bias towards the class label. The performance of the new models will be evaluated using F-measure, AUC and prediction accuracy. The resulting models will be used to test the remaining 10% of the unseen data from our total instances, reserved as test data.

- **Success Criteria:** A prediction accuracy, F-measure and AUC value of above 0.6 indicates the suitability of the feature subset and classifier model to new data set. The best classifier will be determined from the highest overall average performance from the evaluation metrics.

3.2.3.5 Model Evaluation
**Formative Evaluation:** WEKA already provides us with some very powerful statistical analysis and metrics for the evaluation. We performed a formative evaluation at each step of building our classifier by comparing the F-measure and AUC values to the previous value. This activity enabled us to have an objective analysis when comparing their performances. We ensured this by excluding one feature at a time using the filter algorithms. We have demonstrated this from section 3.2.3.1 to 3.2.3.2. The wrapper algorithm would have intuitively taken out the features it feels are irrelevant. This might make some of our judgments subjective.

**Summative Evaluation:** The final results from our analysis would be compared to our initial values before feature reduction. It was on this basis that we concluded effect of feature reduction as well as the final features from our experiment. This was demonstrated in section 3.2.3.3. The results of the evaluation also showed an appropriate model for the chosen feature set. Finally, final features will be evaluated on new data as we shall see in our analysis.
CHAPTER 4: ANALYSIS OF THE DATASET

4.1 Overview of The Analysis:
We shall begin our analysis with the questions identified in our project objectives. We shall then use the experiments and evaluations, explained in the methodology in chapter 3, to answer those the questions. The tool that will be used to achieve this is WEKA for data mining.

One of the aims of this section is to identify the most relevant features from our feature set that best predicts the class label. A good way to commence this task is by considering the historical data (data from the questionnaire) to determine what features influenced their performance in the past. Another goal of this section is to identify the classifier that performs best, based on those set of features. We shall compare the performance of all the algorithms gotten from our results choose the best.

4.2 Evaluation and Discussion of the results:

4.2.1 Question 2 Analysis:
• How do the different EDM algorithms/ techniques perform when applied to the features with an imbalanced class label?
We have built four models (classifiers) without applying any feature reduction techniques on the initial data set. The models were built using ten-fold cross-validation training techniques to get an accuracy close enough to the truth. Cross-validation is a common technique in data mining and ML tasks that help reduce the effect of overfitting when building a classifier [60].

![Image of Weka Explorer with imbalanced data set](image)

**Figure 25: Imbalanced Data set with 44 Instances**
The figure above shows the imbalanced class label discussed in section 2.5 without applying any preprocessing or data balancing techniques. The initial data has 44 instances and 53 attributes that includes the
class label (CL53). The chart within the diagram shows that four (4) out of those instances were labeled as having a poor performance. This results in an uneven weight distribution across the other class labels. Figure 26 below shows the initial performance of the classifiers. This is represented by the F-measure and AUC evaluation criteria. It is important to re-emphasize that the AUC is simply a bargain between the sensitivity and specificity of the classifier. The F-measure, on the other hand, is better suited for evaluating the performance of the classifier since it is the symphonic mean between the recall and precision [20].

We noticed that the models built displayed an effectiveness which varies between 0.40 and 0.54 for F-measure, and from 0.50 to 0.56 for AUC. These results indicate that EDM classifiers can predict at least 40% and at most 56% of the performance of students taking programming courses from the imbalanced data.

Secondly, J48 has the highest prediction accuracy of 54.5% even if it had a lower AUC score than Logistic Regression and Naïve Bayes. However, Its F-measure and AUC value was noted at 0.536 and 0.534 respectively. Note that a classifier like J48 can be biased toward the majority class. This may not be an accurate prediction because of the imbalanced class label that leads to a difficult data distribution. However, these values are not so good even if they performed better than all other classifiers. Nevertheless, these are the highest values amongst the other classifiers and can attain these value without removing any features from the imbalanced dataset.

We can conclude and answer the question by saying that the classifiers performed badly when applied to an imbalanced data that has not been pre-processed. The low F-measures indicates low precision and recall values when classifying. The low AUC values reveal a lower True Positive rate than False positive rate.

4.2.2 Question 3 Analysis:

- **How does data-balancing affect the performance of the classifiers with respect to their F-measure and AUC Values, assuming all its features are retained?**

To answer this question, we must attempt to balance the minority class we saw earlier in Figure 25. This would reduce the effect of overfitting in our model. An imbalanced data would result in the classifiers being more interested in the noise from the data than the actual relationship between the data [46].
WEKA provides a supervised algorithm known as SMOTE (see section 2.5.2.1) for balancing a minority class. We have increased the minority class by 100%, twice. The second largest class was increased by 25%. We then applied an unsupervised-randomized technique to the new dataset. The diagram below shows the results of the experiments, particularly noting the increase in the chart from Figure 25 above.

![Figure 27: Balanced Dataset with 60 Instances and No Feature Reduction Applied](image)

The total instances of the new dataset have gone from 44 to 60. The minority classes (poor and average) have improved weights the data is more balanced. Figure 28 shows the improvement each classifier had with a more balanced data set without feature reduction. The classifiers have more training scenarios to learn from than before. Their F-measures vary from 0.57 to 0.69, while the AUC values vary from 0.64 to 0.87. This indicates an improvement in the preciseness of our classifier. Notably, Random Forest outperformed the other classifiers with an F-measure value of 0.691 and an AUC value of 0.866. In fact, it classified 41 out of the 60 instances correctly, one more than Naïve Bayes could manage.

![Figure 28: F-measure and AUC performance Of the Classifiers on a balanced data set](image)
We can conclude that EDM algorithms can make predictions on either a balanced or an imbalanced data. Furthermore, data balancing had a positive effect on the performance of the classifiers. This is largely because the classifiers now have more scenarios to learn from. Neglecting this step can result in some classifiers being biased and lead to an over-fitted model.

4.2.3 Question 4 Analysis:

- **How does Dimensionality Reduction technique, Specifically Feature subset selection, affect the performance of these features?**

To answer this question, we first performed the data preprocessing described in section 3.2.3.2 on the balanced dataset. By applying Information gain (IG), Information Gain Ratio (IGR), Relief F (RF) and Correlation-based (CB) filter algorithms, we iteratively removed twenty-six (26) features. All these filter techniques can rank features based on their relevance. We then built our classifier at each step and recorded the AUC and F-measure values. The results from the experiment are shown in the graphs for each classifier. These graphs summarize the maximum and minimum levels each classifier can attain when a certain number of features are removed. The Y-axis represents the number of features that were removed. The X-axis represents the weighted average of the AUC and the F-measure for each of the filter algorithms. All these representations are shown for each classifier. Finally, our evaluations and conclusions where made when comparing the classifiers.

a. **Naïve Bayes Analysis**: Naïve Bayes classifier attained the highest F-measure value of 0.704 when 9 features were removed using CB filter techniques (CB-43). It also attained the same value with IGR techniques when 8 features were removed and 44 were retained (IGR-44). However, it had a lower relative absolute error (RAE) of 48.63% with IGR-44. This was lower than the 49.2% obtained by CB-43. In this case, we shall consider IGR-44 for F-measure. On the other hand, it attained the highest AUC Value of 0.86 when 5 features were removed using RF Filtering technique. Therefore, we considered the RF-47 (Relief-F with 47 features retained) for further analysis. The figure below summarizes the experiment.

![Naive Bayes Classifier](image)

**Figure 29: Summary Chart for Naïve Bayes Classifier When All the Filter Algorithms were Applied**
b. **Random Forest Analysis:** Random Forest classifier attained its highest values for F-measure at 0.738 after excluding 17 features and retaining 35 using CB filter algorithms (CB-35). Further observation revealed a marginally lower RAE of 60.74% with CB-35 than for other filter algorithms. The AUC for RF was the highest at 0.883 using IG filter after excluding 13 features and retaining 39 (IG-39). This can be observed this in the figure below.

![Random Forest Analysis Chart](image)

**Figure 30: Summary Chart for Random Forest Classifier When All the Filter Algorithms were Applied**

c. **Logistic Regression Analysis:** Logistic Regression could attain a high F-measure value at 0.714 when using IG filter technique. It reaches this value by excluding 19 features and retaining 33 features (IG-33). It is interesting to point out that IG had an unstable performance when retaining more features. Its performance started dropping on reaching its peak value. The AUC, on the other hand, was the highest at 0.812, retaining the same number of features (IG-33). Further consideration of the RAE shows a lower value of 43.8%, much lower than the other classifiers. In general, logistic regression had a more desirable response to IG filter than the other filters. The results can be seen in the figure below.

![Logistic Regression Analysis Chart](image)

**Figure 31: Summary Chart for LR Classifier When All the Filter Algorithms were Applied**
d. **J48 Analysis:** The F-measure was highest at 0.611 when IG filter was used to remove 24 features and retain 28 (IG-28) while building J48 classifier. An interesting thing to note about this classifier is its response to the reduction of features in a general sense. Although its performance dropped slightly after a few features were removed, the F-measure value for IG still experienced some steady improvement. This was more visible when three-quarter of the features were removed. This improvement outperformed the other filter algorithms as none of them had a score greater than 0.60.

![J48 Classifier](image)

**Figure 32: Summary Chart for J48 Classifier When All the Filter Algorithms were Applied**

The AUC for IG also had a similar performance although, it had a drop when a quarter of the 26 features where removed. By excluding 17 features and retaining 35 (IG-35), it achieved a value of 0.71 and had some stability along its axis as more features were removed. We also observed that its RAE was lower than the other filters at 63.7%. **Figure 33** below shows the final tree constructed from IG-28 and IG-35.

![Figure 33: Visualizing the IG-28 Tree from J48 Classifier](image)
The above IG-28 tree visualization shows that EX18 (Experience in Python programming) is the most influential attribute/feature. The numbers in parenthesis simply explains the number of observations that obeyed the classification rule (left). The number on the right of the parenthesis is the number of exceptions. The IG-35 tree below also shows that EX-18 is influential in its classification rules.

![IG-28 Tree Visualization]

**Figure 34: Visualizing the IG-35 Tree from J48 Classifier**

We can observe that the performance of most of the classifiers improved when features are being removed. This validates the discussion we had in chapter two. Omitting any more features beyond this point may cause the classifier to be over-fitted. Therefore, benchmarking before feature reduction is important. We can see this clearly from the F-measure of the Random forest classifier when applying the Correlation-based (CB-33) filter in above Figure 30 above.

### 4.2.3.1 Further Analysis from the Results from Question 4

It is important to highlight at this point that the best performing curves from our earlier analysis are interesting to observe but are not of absolute importance. The most important thing is the maximum values which we identified during our discussions. These values for all the filters and classifiers have been summarized in the Table 5 and Table 6 below:

<table>
<thead>
<tr>
<th></th>
<th><strong>Naïve Bayes</strong></th>
<th><strong>Random Forest</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>No of Features</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>F-measure</strong></td>
<td>IG</td>
<td>IGR</td>
</tr>
<tr>
<td>Value</td>
<td>49</td>
<td>44</td>
</tr>
<tr>
<td>AUC</td>
<td>0.689</td>
<td>0.704</td>
</tr>
<tr>
<td><strong>ROC Area</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>No of Features</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUC</td>
<td>49</td>
<td>44</td>
</tr>
<tr>
<td>Value</td>
<td>0.851</td>
<td>0.856</td>
</tr>
</tbody>
</table>

**Table 5: Peak values for F-measure and AUC for all the classifiers (First Part)**
Table 5 and Table 6 above show the peak AUC and F-measure for all the classifier algorithms along with their respective cardinalities. These values have been extracted from all the graphs from our experiments. The f- measures from both tables have the highest values at IGR-44, CB-35, IG-33, and IG-28 filter techniques.

The AUC values show their highest values at RF-47, IG-39, IG-33 and IG-35 filtering techniques. A summary table of the feature code is presented below to perform a more in-depth predictive analysis.

<table>
<thead>
<tr>
<th>Subset</th>
<th>Feature Code in Order of Relevance from Highest to Least</th>
</tr>
</thead>
</table>

Table 7: Feature Subset and Feature Code Summary Table

It is clear to see from Table 7 that the effects of dimensionality reduction varied from classifier to classifier with the various filter algorithms. We can conclude and answer question 4 by comparing the initial values in Figure 28 with the peak values in the Table 5 and Table 6. Therefore, reducing the features improved the F-measure and AUC performance of all the classifiers. This answer considers the most optimal subset (peak values) suited for each classifier. For example, we can see from Table 8 that not all the classifiers were improved they were applied to a specific feature subset (CB-35).
4.2.4 Question 5 Analysis:

- Based on the results from the experiments, what are the most relevant features that determine the performance of a student taking programming courses? Furthermore, which of the four-classifier model is best suited for the feature subset?

By building the classifier from each subset established in Table 7, we identified the most optimal subsets that give the best prediction. The figure below is a plot of the prediction accuracy of all the classifiers across the various feature subsets.

![Classifier Performance over different Feature Subsets](image)

**Figure 35: Classifier Performance over different Feature Subsets**

Some of the classifiers gave as low as 55% prediction accuracy for this task. The highest prediction was reached at 73.3% by Random forest with 35 features chosen from Correlation-based filter algorithm. This means that Random Forest (RF) reached optimal dimensionality after removing 17 features. Logistic Regression came the closest to this with 71% at 33 feature subsets from information gain filter algorithm. The other classifiers also had good predictions generally. Another observation is that J48 is the only classifier to have a prediction accuracy below 60% from all the feature subsets. In fact, its best performance came with IG-28, which showed improvement with lesser features.

Prediction accuracy is not the only factor we must consider. The general performance of the other four classifiers is equally important. To choose the best subset, the average performance of each feature subset with respect to its prediction accuracy was performed. This was described in section 3.2.3.2. CB-35 had a higher average performance than all the other subsets. This is chosen as the best feature subset. Figure 36 below shows the final performance and a confusion matrix for the Random Forest classifier extracted from WEKA.
Having identified the feature subset (CB-35) as the most optimal subset, the highest prediction came from Random Forest. It also had a better F-measure and AUC score than the other classifiers, as summarized in the Table 8 for CB-35 below.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>F-measure</th>
<th>AUC</th>
<th>Accuracy %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naive Bayes</td>
<td>0.675</td>
<td>0.834</td>
<td>66.7</td>
</tr>
<tr>
<td><strong>Random Forest</strong></td>
<td><strong>0.738</strong></td>
<td><strong>0.862</strong></td>
<td><strong>73.3</strong></td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>0.617</td>
<td>0.753</td>
<td>61.7</td>
</tr>
<tr>
<td>J48</td>
<td>0.516</td>
<td>0.659</td>
<td>56.6</td>
</tr>
</tbody>
</table>

Table 8: Final Performance Evaluation for Each Classifier on CB-35

Table 9 below summarizes the top 35 features chosen by CB filter algorithms where "**" indicates the most important feature (Emotional Stability) and "--" indicates the least relevant feature (Group meetings). A more appropriate ranking for CB-35 can be seen in Appendix 8.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Feature Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Profile and Experience</strong></td>
<td>UL</td>
</tr>
<tr>
<td>Age</td>
<td>PE2</td>
</tr>
<tr>
<td>Health</td>
<td>PE5</td>
</tr>
<tr>
<td>Work2</td>
<td>PE8</td>
</tr>
<tr>
<td>Emotional Balance</td>
<td>PE10**</td>
</tr>
<tr>
<td><strong>Previous Academic Background</strong></td>
<td>Programming Concepts</td>
</tr>
<tr>
<td>College Performance</td>
<td>PAB11</td>
</tr>
<tr>
<td>Background knowledge</td>
<td>PAB12</td>
</tr>
<tr>
<td>Social Status</td>
<td>PAB13</td>
</tr>
<tr>
<td><strong>Previous Experiences</strong></td>
<td>EX18</td>
</tr>
<tr>
<td>Python</td>
<td>PC43</td>
</tr>
<tr>
<td>Variable Declarations</td>
<td>PC40</td>
</tr>
<tr>
<td>Conditional Operators</td>
<td>PC41</td>
</tr>
<tr>
<td>Loop Operations</td>
<td>PC42</td>
</tr>
<tr>
<td>Pointers References</td>
<td>PC43</td>
</tr>
<tr>
<td>Collections</td>
<td>PC44</td>
</tr>
</tbody>
</table>
We can conclude that a student’s emotional stability (PE10) is the most influential attribute with respect to their performance in school. Understanding python programming (EX18) is another influencer in terms of programming experience. Programming concepts are also influential, especially with a student’s ability to understand how to use variables with a program construct.

Another interesting result was the bottom three features which highlight that group meetings, instructor counselling and handling error are not major deciders. This was different in other feature subsets as their choice of features varied respectively.

### 4.2.5 Question 6 Analysis:

- **How do the results gotten from building the classifiers on the final feature subset compare to the initial F-measure and AUC values gotten from the balanced dataset in Question 3? Is the difference in value statistically significant for each of the classifiers?**

To answer the first part of the question, we compared our results before and after the feature reduction. **Figure 37** and **Figure 38** below show an F-measure and AUC comparative results before and after the dimensionality reduction on the balanced dataset.

![Figure 37: F-measure Values Before and After Feature Reduction](image-url)
These results show that the effectiveness of a few of the techniques where improved for both the F-measure and AUC with the present subset CB-35. This is with an exception for the J48 whose performance decreased by 0.063 for the F-measure value. A reasonable interpretation for the higher F-measure scores for Naïve Bayes, Random Forest and Logistic regression is as listed below:

1. These classifiers are more precise in prediction than before the features were removed. It also shows that their precision values are larger than their corresponding recall.
2. Their higher precision values further indicate that these classifiers are very accurate when classifying the instances.
3. A lower recall than precision indicates that a significant number of instances to be classified were missed. This can be observed in the Random forest classification sample results in the Figure 36 above.
4. J48 is not as precise as the other three classifiers since its values showed a higher recall than precision.

![Figure 38: AUC Values Before and After Feature Reduction](image)

We can see from the figure above that J48 and Logistic Regression showed some improvement for their AUC value. Naïve Bayes and random forest had their performances reduced marginally by 0.011 and 0.004 respectively. The slight decline in the AUC values for these indicates that there was a slight deviation from the initial TP Rate. The reverse is the case for J48 and Logistic regression as their models showed better alignment to the maximum of 1. We can visualize this in the before (Figure 38) and after ROC Curves (Figure 39) plotted in WEKA. Based on these results, we can answer the first part of question 6 with the following observations from our experiment on CB-35:

1. Feature reduction improved the prediction accuracy Logistic Regression and Random Forest from 60 to 61% and 68.3 to 73.3 respectively. It reduced the accuracy of J48 from 58.3 to 56.6% and the accuracy of Naïve Bayes remained the same at 66.7%.
2. The F-measure values for Naïve Bayes, Random Forest and Logistic Regression was improved. This proves that the classifiers where more precise than they were initially. J48 had a lower F-measure value indicating a lower precision during classification than initially.
3. The AUC improved for J48 and Logistic Regression which indicates an improvement in the classifiers performance when making predictions. It was marginally reduced for Naïve Bayes and Random forest, indicating a slight drop in the performance of the classifiers.

![Figure 39: AUC before Feature Reduction for All the Classifiers](image)

![Figure 40: AUC after Feature Reduction for All the Classifiers](image)

The second Part of this question will be answered using a statistical test such as the t-test to see if the above changes are statistically significant or not [11]. We shall do this by comparing the values gotten by excluding each feature from the initial value till CB-35. A p-value of below 0.05 represents a significant difference.
Applying the statistical t-test to each classifier resulted in the p-values both for F-measure and AUC Value in the table below:

<table>
<thead>
<tr>
<th>Metric</th>
<th>Naïve Bayes</th>
<th>Random Forest</th>
<th>J48</th>
<th>Logistic Regression</th>
</tr>
</thead>
<tbody>
<tr>
<td>F-measure</td>
<td>0.0407</td>
<td>0.0035</td>
<td>3.7584 * 10^-5</td>
<td>4.0263 * 10^-5</td>
</tr>
<tr>
<td>AUC</td>
<td>0.1948</td>
<td>0.0055</td>
<td>0.4569</td>
<td>5.0604 * 10^-6</td>
</tr>
</tbody>
</table>

Table 10: p-values Obtained from the t test experiment for each classifier

We can observe from this test with respect to the benchmark of 0.05, only Naïve Bayes and Random Forest had significant changes to its F-measure value. We demonstrated this from our earlier analysis that addressed question 3 as features were being removed iteratively. This explains the improvement in the preciseness of Naïve Bayes and our chosen classifier Random forest. However, the same cannot be said about J48 and Logistic regression which showed a significant decline in performance for F-measure and AUC.

Furthermore, the AUC value only showed a significant change for Random forest. The changes for Naïve Bayes, J48 and Logistic regression where statistically insignificant. This implies that reducing the features for Random forest reduced its performance with respect to TP-Rate. Furthermore, we can see that the other classifiers would still have had a similar performance if more features were retained.

In conclusion, the above table and discussion show that random forest had a statistically significant change for both measurements. We therefore reject the null hypothesis in favor of the alternate hypothesis stated in chapter 1.1.

4.2.6 Question 7 Analysis:

- How will each classifier model perform on unseen data applied on the most optimal feature set obtained from question 5?

It is important to point out that there was not enough data to split our data into training and testing else we may not have enough training data. However, we have managed to use our original balanced data source to test our model without compromising the integrity of our experiment. This was achieved by splitting the balanced data into training and test set. The classifiers were then built on the new training set and tested on the new test set. This will ensure that the model does not know about the new data (test data).

Suppose we apply the same configuration for the initial model built in question 5 to the new test data without training on a separate training data, it will give us a much better reading like in Figure 41. We have set up a test data of 10% (6 instances) of the original 66 instances and applied the random forest classifier configuration of CB-35 to this test data. We can see that its prediction accuracy, AUC and F-measure values are a perfect 100%. This result is not accurate because the model was built on all this information previously. We must therefore build new classifier models to perform this test.
Having identified this, the following steps describe the path to answer the question from using our initial training data:

1. **Splitting the data into training and testing set**

WEKA does not explicitly offer a feature to separate your data into training and testing. It does, however, offer a *percentage split* option to specify how much of the data should be used for training and testing. The limitation of this approach is that, a standard training option like cross-validation cannot be utilized to train the model. A much better and coordinated approach where we can utilize cross-validation to train our model was utilized.

WEKA offers an unsupervised instance learning method called resampling (see section 2.7.5.3). We have resampled the original data set without replacement. This was done to avoid duplicates in our training and test data. It is also a standard ML practice that ensure integrity in our data. We have used 90% (54 instances) of the total 60 instances as training data and the remaining 10% (6 instances) as test data (see Appendix 7 for the configuration). The reason for this ratio is so that we can have more instances (54 instances) than attributes (53 attributes) and avoid an over-fitted model. An unsupervised-randomization technique (explained in section 2.5.2.1) was applied to the new training and test data to further enhance the capabilities of our cross-validation process as performed earlier in section 4.2.2. A sample extract from the *arff* testing file can be seen in the figure below.

---

Figure 41: Applying the Model from Question 5 on 10% of the same data it was built from
Figure 42: Test File with 6 instances as an unseen data

2. Rebuilding the model on the new training set using the same configuration of Correlation-based filtering techniques while retaining 35 features.

We used the features obtained from CB-35 from this experiment to train our models from our training data (54 instances) using 10-fold cross-validation. These features were already identified to be the most relevant features. The table below represents the results of the F-measure, AUC and prediction accuracy for all our models. We can see that random forest had a better average performance across all three metrics than the other classifiers using CB-35 configuration.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>F-measure</th>
<th>AUC</th>
<th>Accuracy %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naïve Bayes</td>
<td>0.581</td>
<td>0.819</td>
<td>57.4</td>
</tr>
<tr>
<td><strong>Random Forest</strong></td>
<td><strong>0.60</strong></td>
<td><strong>0.798</strong></td>
<td><strong>59.3</strong></td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>0.618</td>
<td>0.758</td>
<td>61.1</td>
</tr>
<tr>
<td>J48</td>
<td>0.506</td>
<td>0.654</td>
<td>51.9</td>
</tr>
</tbody>
</table>

Table 11: Showing the Classifier Evaluation Metrics Obtained from the New Training data

3. Applying each model’s new configuration on the unseen test data set that was not originally used for training

This step gives us a final view on the performance of our models on unseen data. We have applied the model built from step 2 above to our test data (6 instances). The results are summarized in the table below:

<table>
<thead>
<tr>
<th>Classifier</th>
<th>F-measure</th>
<th>AUC</th>
<th>Accuracy %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naïve Bayes</td>
<td>0.667</td>
<td>0.917</td>
<td>66.7</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.667</td>
<td>0.917</td>
<td>66.7</td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>0.667</td>
<td>0.917</td>
<td>66.7</td>
</tr>
<tr>
<td><strong>J48</strong></td>
<td><strong>0.822</strong></td>
<td><strong>1.000</strong></td>
<td><strong>83.3</strong></td>
</tr>
</tbody>
</table>

Table 12: Showing the Classifier Evaluation Metrics Obtained from the New Unseen Test data

From our experiments and results, it is safe to answer the question and conclude that all these classifiers perform well with unseen data using CB-35. It is clear to see that all the classifiers performed very well across all metrics for the unseen data. J48 performed much better than the rest. However, when presented with more data (90%) when training in step 2, J48 had the worst performance out of all the classifiers. The scarcity of data for this experiment is clearly a limitation to give a more accurate conclusion.
4.3 Threats to Validity

It is evident to see that our experiments produced interesting results showing how convenient it is to use EDM techniques. However, it is ethical to highlight some of the threats that may undermine our findings in this report.

1. This research used data from one source, a combination of first and second year students in Heriot-watt University in Dubai campus. Most students across the different campuses were not considered at this point. This implies that our results are specific to this data and are not general.

2. The total number of instances from the original data was 44. We had to use SMOTE to balance the majority and minority classes to perform our experiment. Clearly, this approach yielded better results when we compared results. Thus, we acknowledge that there were not enough scenarios to train the classifiers on initially.

3. The final feature subsets were chosen to be the best performing features because they performed best with the classifiers. For the fact that “students’ group meeting” was the least significant feature using CB-35 does not necessarily mean the other filter classifiers agree on that. This is another clear indicator that a limited amount of data may not be suitable when training your classifier.

4. Furthermore, there was not enough data to be split into training and test set. We validated the effectiveness of the classifiers on new data by sensibly using the same dataset. We split the balanced dataset into training and testing data. This reduced the amount of training scenarios we had for our experiment. We also tested on only CB-35 as this was chosen to be the best set of features from our earlier experiments. It is possible that the other feature subsets may produce better or worse results.

5. Finally, we used F-measure and AUC to determine the effectiveness of our classifiers as is the norm in most EDM literature. We also used prediction accuracy and RAE to make critical decisions on a few occasions particularly in section 4.2.3. However, there are other measures that can still be used such as Kappa coefficient. This is a statistical measure that is used to determine the degree of concurrence amongst the other scoring metrics [82].

4.4 Chapter Summary

We observed that the best features from our original feature set was gotten from CB-35. This means that 35 features were retained from the original 53 features using CB filter techniques. Emotional balance, experience in python programming and usability of variable declaration were the top most influential features respectively. Random forest performed better than the other 3 classifiers for prediction accuracy, F-measure and AUC values. Its p-values for F-measure and AUC showed a statistically significant change from its initial value.

Finally, we performed our experiment on unseen data from the few instances we had. A judicious utilization and application of ML and data mining techniques ensured that we avoided overfitting our model. All the models performed well to the unseen data and indicates that these techniques are suitable for a real-world application.
CHAPTER 5: PROFESSIONAL, LEGAL, ETHICAL AND SOCIAL ISSUES

5.1 Professional Issues

The main concern about professionalism in any research field, is the integrity of the work done. The computer law is guiding the use of data under Data Protection Law Policies. This research has ensured that the work is addressing a specific issue faced within the university. This was achieved by conducting an in-depth research by using materials and online resources, relevant to the problem situation. The work done, diagrams and other external sources from other researchers that was used in the literature survey were cited appropriately.

Furthermore, the quality of the people-data and methodology that was used is also an issue of professionalism rightly highlighted by [83]. This is due to its impact on the outcome of the project. This was handled carefully by considering each attribute within the questionnaire and adopting the CRISP-DM methodology where applicable. It has been used in many data mining processes and was applied to this project. It involves setting project goals, objectives and model evaluation. This aligns the dissertation to a set of standards which improves the quality of the work done. This was reviewed with the supervisor and an agreement was reached before data was gathered. Considering these constraints ensures that the report abides by the code of conduct necessary to avoid professional issues.

5.2 Legal Issues

As earlier discussed, there are numerous software available for data mining. The problem with most of these software is in their legal, social and ethical barriers that make them difficult to adopt in a new context [84]. This research uses WEKA 3.8 to perform the data mining tasks. WEKA is an open source software and is licensed under GNU General Public License (GPL 3.0). The terms and condition to align with the terms of the use have been reviewed. The applicability of this software was tested before applying it to this report. There is no need for obtaining a license for using WEKA because it is explicitly declared as an open source tool [51]. The software was downloaded directly from the website and not through any other vendor site.

Another issue that can have legal implications is with the storing of people-data. We are all exposed to different types of cybercrime and one can carelessly implicate oneself. The data collected does not include social media exchanges like emails and online questionnaires. The information from the students does not also include personal information. This was avoided so that their identity was protected. The questions within the questionnaire was structured in such a way that the student is comfortable to answer. This ensured anonymity amongst the participants in the unlikely event of data theft.

5.3 Ethical Issues

Access to free data on its own is not socially or ethically an issue. The ethical and social issue comes into play when personal data is mined directly from people (people-data). Using questionnaires, online or physical, can pose a real threat to the privacy of individuals. Hence privacy is a major concern. We sought the consent of the
participants before obtaining their data. Therefore, no participant was mandated to give information. The researcher must have also educated the participants about the purpose of the research before obtaining their data. Any information gotten from this dissertation will be kept confidential. Only stakeholders such as the project supervisor, can have access to this information. This is to ensure that ethical principles are considered.

5.4 Social Issues

The initial paragraph addressed the ethical issues concerned with the project, however the social issues must be addressed. The aims and objectives of this dissertation is in the best interest of the students. This research was carried out without bias and data evaluation will be done objectively. The data set we used was strictly from the respondents of the questionnaire. We were not tempted to source data from non-stakeholders to obtain optimal results. We have instead, applied industry standard techniques like SMOTE on the little amount of data we got.
CHAPTER 6: SUMMARY

6.1 Conclusion

A successful programmer is one who can transform ideas, oral and written, into codes than can be used to tackle our day to day challenges. This research focused on the identifying the most relevant features that can influence the performance of a student taking programming courses. The features were identified in chapter 2.2 and summarized in chapter 3.2.1. Another aim of the project was to identify which of our four algorithms selected gives the best performance.

Firstly, filter methods are more suited for datamining tasks because they can assess the individual worth of the attributes. We also showed from our comparison that they are they are faster than wrapper methods and are suitable for small data with fewer features. They have been used in most data mining literature as a preprocessing step so as not to directly optimize the performance of the class label. Wrapper methods, on the other hand, make the decision on their own. They eliminate features so that it can generate the best possible prediction accuracy. This is more suited for ML tasks where optimal performance of the classifier is of paramount importance.

Additionally, we can see from our investigation that a careful combination of AUC, F-measure and prediction accuracy for the CB (Correlation-based) filter method can yield optimal dimensionality for the feature set. Retaining 35 features resulted in better results than reducing more features. Random forest gave the best prediction accuracy, F-measure and AUC at 73.3%, 0.738 And 0.862 Respectively. Based on these results, applying this classifier to a fresh data set will give us optimal results with a good precision from F-measure. However, we noted a significant reduction in AUC value when the features were reduced.

Furthermore, we noticed a positive performance from all the classifiers when applied to unseen data. Although the same data was used for the second part of the experiment, the data models were trained and tested newly without using the old model. Random forest outperformed the other classifiers when trained on the training data. However, J48 gave a better overall performance on the test data. This might be an inaccurate performance since there was not enough training data. Also, the same classifier performed poorly when compared to others with more training data.

Finally, the best evaluation mode to be used when performing this task must be decided by the instructors. If the intention is to retain all the features, then it is best to consider a higher AUC value with a lower prediction accuracy. Otherwise, the F-measure value of the random forest classifier using 10-fold cross-validation should be considered with CB-35 features set.

6.2 Recommendations

The first point to consider is the final results from our experiment. Instructors are advised to take into special consideration, the effects of a student’s emotional state (PE10) in class. There is a famous saying that “Health
is wealth”. Our human nature makes it difficult to handle pressure. The focus on the wellbeing our students cannot be overemphasized as this translates to their performance academically.

Furthermore, there were some other interesting features that must be noted. Revision classes and feedback from lecturers are very helpful to students. Students need that truthfulness and openness from their lecturers to motivate their performance. Our results also highlight the influence of the lecture notes and class demos used to teach during lectures. A visual experience from a trusted source, such as the lecturers, is considered very helpful.

Finally, the various programming concepts were considered as major determinant during our test. In fact, all the features where retained after our experiment (see Table 9). Five features were in the top-fifteen most relevant features retained by Correlation base filter algorithm. Our lecturers must invest more time in ensuring that the students grasp the basic principles and activities happening with their program. This can be enforced from the foundation classes by pointing out the difficulties they are likely to face in future. Most of these features were considered important in the works by [8] and [9], further validating their work.

6.3 Future Works

Having identified the threats to validity in section 4.3, this study can be improved by firstly integrating a data from a wider source. This can be done by getting more data from students across all the campuses and from all levels. In fact, data from three-quarter of the campuses can be used to train the classifier while the remaining can be used for testing. WEKA provides a “use training set” and “supplied training set” function for these purposes. This is suitable for large amounts of data where we can afford to separate training from testing.

Secondly, the data collection mode was inefficient because it was done manually. A website can be built for this purpose. This way, a link can be shared to every student no matter their location, and with proper motivation, we can generate data of more than 150 instances. Additionally, there are free questionnaire tools can be used to make the process of data collation easy.

Furthermore, only 26 features were excluded from the total features. We can also investigate the effects of removing more features. This may affect the performances positively, negatively or neutrally.

The Final recommendation would be to see the effect of using other classifier models and comparing the results with this one. It is even possible to implements other filtering or wrapper algorithms to select our features. Every classifier and filtering technique has its strengths and weakness and a careful study may discover a better model for this same task. Additionally, it is possible to perform other data mining tasks on this data like clustering. The aim of this research is to support decision making at all level as it concerns the performance of students, staff and lecturers within the educational institution.
REFERENCES


Appendix 1  Questionnaire

**QUESTIONNAIRE**
Researcher-Made Questionnaire on The Factors Affecting the Performance of Students Taking Programming Courses in Heriot-Watt University. 2016-2017 Session

**I. Profile and Experience**
1. What is your present Year of Study?  
   - [ ] 1st Year  
   - [ ] 3rd Year  
   - [ ] PG  
   - [ ] 2nd Year  
   - [ ] 4th Year
2. What age bracket do you belong?  
   - [ ] Below 15 yrs.  
   - [ ] 20 to 25 yrs.  
   - [ ] Above 30 yrs.  
   - [ ] 15 to 19 yrs.  
   - [ ] 26 to 30 yrs.
3. What is your gender?  
   - [ ] Male  
   - [ ] Female
4. What type of student are you?  
   - [ ] Full-Time  
   - [ ] Part-Time
5. Do you think you are healthy Enough?  
   - [ ] Strongly Agree  
   - [ ] Neither Agree nor Disagree  
   - [ ] Strongly Disagree  
   - [ ] Agree  
   - [ ] Disagree
6. Do your parents give you enough money/ do you earn enough to take care of yourself?  
   - [ ] Strongly Agree  
   - [ ] Neither Agree nor Disagree  
   - [ ] Strongly Disagree  
   - [ ] Agree  
   - [ ] Disagree
7. Are you working presently?  
   - [ ] Yes  
   - [ ] No
8. Did you ever work on a programming project, assignment or task before coming to college?  
   - [ ] Yes  
   - [ ] No
9. Where you a science student in your pre-college education?  
   - [ ] Yes  
   - [ ] No
10. How well can you manage school and home and other social activities?  
    - [ ] Excellent  
    - [ ] Fair  
    - [ ] Very poor  
    - [ ] Good  
    - [ ] Poor

**II. Previous Academic Background**
11. How was your performance overall after your pre-college education?  
    - [ ] Excellent  
    - [ ] Fair  
    - [ ] Very poor  
    - [ ] Good  
    - [ ] Poor
12. How important were the foundation classes to your present knowledge of programming?  
    - [ ] Very important  
    - [ ] Neither important nor unimportant  
    - [ ] Very unimportant  
    - [ ] Important  
    - [ ] Unimportant
13. How influential are the universities social activities to your academic performance in school?  
    - [ ] Very important  
    - [ ] Neither important nor unimportant  
    - [ ] Very unimportant

**III. Previous Programming Experiences Before Coming To school**

**Direction:** Please check [ ] and honestly rate your performance using the following scales:

<table>
<thead>
<tr>
<th>5-Very Good</th>
<th>4-Good</th>
<th>3-Poor</th>
<th>2-Very Poor</th>
<th>1-No Experience</th>
</tr>
</thead>
<tbody>
<tr>
<td>How would you rate your Programming skills before coming into college in the following languages?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- [ ] C
- [ ] C++
- [ ] Java
- [ ] JavaScript
- [ ] Python

19. How Helpful have your previous experiences in programming been to your current performance in programming Courses?  
    - [ ] Extremely Helpful  
    - [ ] Somewhat Helpful  
    - [ ] Not Helpful At all  
    - [ ] Very Helpful  
    - [ ] Slightly Helpful
IV. Students Learning and Studying Mode

**Direction:** Please check ☐ and honestly rate the usefulness of the following using the scales below:

<table>
<thead>
<tr>
<th>How helpful are the following to your understanding of programming?</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 - Extremely Helpful</td>
</tr>
<tr>
<td>Studying Programming course books?</td>
</tr>
<tr>
<td>Watching online programming videos?</td>
</tr>
<tr>
<td>Student Group meetings?</td>
</tr>
<tr>
<td>Students Peer Tutoring sessions on programming</td>
</tr>
<tr>
<td>Programming sites for tutoring e.g. Code Academy etc.</td>
</tr>
<tr>
<td>Lecture attendance</td>
</tr>
<tr>
<td>Visit, mail or calling your course instructor for assistance Excluding lectures</td>
</tr>
<tr>
<td>Lab sessions</td>
</tr>
<tr>
<td>Materials on discovery (Library Resource)</td>
</tr>
<tr>
<td>Number of students in your class</td>
</tr>
<tr>
<td>Studying Lecture slides</td>
</tr>
</tbody>
</table>

V. Lecturers Mode of Teaching

**Direction:** Please check ☐ and honestly rate your understanding using the following scales:

<table>
<thead>
<tr>
<th>How helpful are the following Modes of teaching to your understanding of programming?</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 - Extremely Helpful</td>
</tr>
<tr>
<td>Reading from slides</td>
</tr>
<tr>
<td>Performing live Demos during lectures</td>
</tr>
<tr>
<td>Unsupervised Learning i.e. Making you research about a topic before the next class</td>
</tr>
<tr>
<td>On the Spot Quiz after each lecture</td>
</tr>
<tr>
<td>In class Exercises during lectures</td>
</tr>
<tr>
<td>Intense Lectures i.e. few breaks in between Lectures</td>
</tr>
<tr>
<td>Revision the last lecture before starting of the new one</td>
</tr>
<tr>
<td>Feedback from your coursework's after submission</td>
</tr>
</tbody>
</table>

VI. Programming Concepts

**Direction:** Please check ☐ and honestly rate your performance using the following scales:

<table>
<thead>
<tr>
<th>Rate your understanding of the following concepts?</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 - Excellent</td>
</tr>
<tr>
<td>Constructors</td>
</tr>
<tr>
<td>Variable Declarations</td>
</tr>
<tr>
<td>Conditional Operators like IF and ELSE</td>
</tr>
<tr>
<td>Loop Operations like FOR and WHILE loops?</td>
</tr>
<tr>
<td>Pointers and References</td>
</tr>
<tr>
<td>Collections (Array List, Linked List etc.)?</td>
</tr>
<tr>
<td>Input/output Operations</td>
</tr>
<tr>
<td>Error Handling</td>
</tr>
<tr>
<td>Programming Language Libraries</td>
</tr>
<tr>
<td>String Handling</td>
</tr>
<tr>
<td>Memory storage in your programs i.e. How your program affects the memory utilization of the system?</td>
</tr>
</tbody>
</table>

VII. Extracurricular activities

<table>
<thead>
<tr>
<th>How often would you attend programming conferences?</th>
</tr>
</thead>
<tbody>
<tr>
<td>☐ always</td>
</tr>
<tr>
<td>☐ Very Often</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>How many hours do you spend practicing programming in a day?</th>
</tr>
</thead>
<tbody>
<tr>
<td>☐ 1 hour</td>
</tr>
<tr>
<td>☐ Less than 2 Hours</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>How well can you construct a program from scratch on your own, applying all the concepts you know</th>
</tr>
</thead>
<tbody>
<tr>
<td>☐ Extremely well</td>
</tr>
<tr>
<td>☐ Very well</td>
</tr>
</tbody>
</table>

VIII. Current Performance

<table>
<thead>
<tr>
<th>Right on till now, how do you rate your performance, academically based on your current CGPA?</th>
</tr>
</thead>
<tbody>
<tr>
<td>☐ Excellent</td>
</tr>
<tr>
<td>☐ Good</td>
</tr>
</tbody>
</table>
Appendix 2  Show the visualization capabilities in WEKA GUI

Appendix 3  Sample .arff File

Appendix 4  Data Preprocessing in WEKA
Appendix 5  Building a classifier model in WEKA With 10-fold Cross Validation Test Option

Appendix 6  Applying Feature Selection Algorithms to a Data set in WEKA
### Appendix 7  
T-test Analysis table showing a p-value of less than 0.05

<table>
<thead>
<tr>
<th>Before Reduction</th>
<th>After Reduction</th>
<th>t-test: Paired Two Sample for Means</th>
<th>Before Reduction</th>
<th>After Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.866</td>
<td>0.86</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.866</td>
<td>0.853</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>0.856</td>
<td>0.8575555556</td>
<td>5.2204E-32</td>
<td>0.000126967</td>
</tr>
<tr>
<td>Variance</td>
<td>0.866</td>
<td>18</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>Hypothesized Mean Difference</td>
<td>0.855</td>
<td>df</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>t Stat</td>
<td>0.866</td>
<td></td>
<td>3.179518517</td>
<td></td>
</tr>
<tr>
<td>P(T&lt; t) one-tail</td>
<td>0.866</td>
<td>0.002741946</td>
<td>1.739606726</td>
<td></td>
</tr>
<tr>
<td>t Critical one-tail</td>
<td>0.845</td>
<td>0.005483892</td>
<td>2.106815578</td>
<td></td>
</tr>
<tr>
<td>t Critical two-tail</td>
<td>0.866</td>
<td>0.859</td>
<td>0.835</td>
<td></td>
</tr>
<tr>
<td>0.866</td>
<td>0.876</td>
<td></td>
<td>0.855</td>
<td></td>
</tr>
<tr>
<td>0.866</td>
<td>0.853</td>
<td></td>
<td>0.866</td>
<td>0.852</td>
</tr>
</tbody>
</table>

### Appendix 8  
Final Ranked Features By CB-35

**Attribute Evaluator (supervised, Class (nominal): 53 CL53): Correlation Ranking Filter**

Ranked attributes:
- 0.362 10 FE10
- 0.354 18 EX18
- 0.347 40 PC40
- 0.331 33 LM33
- 0.311 8 FE8
- 0.304 25 LS25
- 0.303 41 PC41
- 0.29 36 LM36
- 0.287 51 EC51
- 0.286 45 PC45
- 0.285 52 EC52
- 0.28 44 PC44
- 0.276 42 PC42
- 0.274 2 P2
- 0.274 37 LM37
- 0.272 27 LS27
- 0.272 48 PC48
- 0.269 31 LM31
- 0.263 11 PAB11
- 0.26 38 LM38
- 0.258 46 PC46
- 0.255 5 FE5
- 0.251 32 LM32
- 0.249 29 LS29
- 0.246 47 PC47
- 0.246 50 EC50
- 0.238 13 PAB12
- 0.235 43 PC43
- 0.235 26 LS26
- 0.234 13 PAB13
- 0.232 33 LS23
- 0.23 36 LS26
- 0.229 49 PC49
- 0.229 32 LS22
Appendix 9  Training and Test Data

Training Data: 90% (54 instances) of 60 instances

Test Data: 10% (6 instances) of 60 instances gotten by setting the invert Selection as True