Computer Science

Mike Nkongolo

A Web-Based Prototype Course Recommender System using Apache Mahout

Project Report

GRIN

Bibliographic information published by the German National Library:

The German National Library lists this publication in the National Bibliography; detailed bibliographic data are available on the Internet at http://dnb.dnb.de .

Imprint:

Copyright © 2017 GRIN Verlag
ISBN: 9783668554351

This book at GRIN:

https://www.grin.com/document/375739

Mike Nkongolo

A Web-Based Prototype Course Recommender System using Apache Mahout

GRIN Verlag

A Web-Based Prototype Course Recommender System using Apache Mahout

Mike Nkongolo

A research report submitted to the Faculty of Science for the
BSc (Hons) in Computer Science degree.
School of Computer Science and Applied Mathematics
University of the Witwatersrand
Johannesburg, South Africa

Contents

Abstract

Most universities offer a wide range of courses in which students can enrol. As a result, students may feel overwhelmed with the many possibilities and large amount of information, resulting in having a difficult time deciding what to sign up for. To this end, there is a need for a system that can assist students in this crucial process. Collaborative recommenders are known to be useful for finding similarities between students and course combinations that they have taken. Thus, we set out to develop a web-based recommender application that could generate a list of valuable, accurate course recommendations, taking into account a student's likelihood of succeeding academically. We used a historical dataset containing data about past Computer Science I students at the University of the Witwatersrand, including the combinations of courses they took and their respective marks. We calculated the relationships between courses and performance, to find which courses students did well or badly in. The dataset was also used with Apache Mahout, a free library of recommender algorithms, in order to generate course recommendations. This was done by using the Spearman correlation to determine similarities between past students in order to recommend courses to new students that other students had performed well in. The web components of the system were developed with JSP and Servlet software. We evaluated the recommender system on the basis of coverage and accuracy. We found there to be strong correlations between individual course marks and overall year marks, indicating that recommending courses past students did well in was likely to increase chances of doing well overall. The implemented system was found to have a coverage of 100%, indicating all students in the test dataset were able to have recommendations generated for them. The accuracy of the system, measured by the F1 metric, was found to be 0.66 (reaching as high as 0.72 at smaller user neighbourhood sizes), meaning that the recommendations generated by the system were accurate in the majority of cases. This allowed us to determine that the size of the dataset used to train the system was sufficient. Finally, the web application that was developed was intuitive, easy-to-use and incorporated the elements of the recommender system successfully in order to convey recommendations to students. Thus, it was possible to conclude that a collaborative recommender approach, developed in a web-based environment using Apache Mahout, is suitable in order to suggest relevant courses to students, while striking a balance between the students' own interests and crucial field-related material in order to ensure academic success. Such a system would be an asset to a university, increasing its students' chances of passing and thus increasing its own reputation as a result.

Acknowledgements

I would like to thank my supervisor, Mr. Michael Mchunu, for his helpful assistance and guidance with my research topic. I am grateful to my friend, Nicolas Scott Telford for his constant encouragement and support.

List of Figures

List of Tables

Dedicated to my mom (Melanie Masengu Ngandu) and dad (Prof. Jean-Paul Nkongolo Mukendi). All my achievements are simply extensions of their love and support!

Chapter 1

Introduction

In modern-day society, countless opportunities are open to young adults leaving secondary education. For many of them, going straight to work after finishing high school is seen as the most attractive option, for whatever reason one might have. One of the reasons may be the need to support themselves and their families financially, or they may simply be less academically inclined and therefore not keen on studying further. On the other hand, for other high school leavers, pursuing a higher level of education is seen as a safer option and a worthwhile investment for the future. A wider choice of jobs and career paths, and in many instances higher potential salaries, are generally open to people with a degree and a higher level of specific subject knowledge.

Typically, universities and other tertiary institutions are commonly associated with higher levels of study. Universities typically offer academic programs encompassing a range of disciplines, with the promise of the conferment of a degree, diploma or certificate at the end of the academic program. Popular fields of study most students are interested in include the arts, sciences, humanities, engineering and commerce. To qualify in any one of these fields, students must work consistently throughout the program. They must display a thorough understanding of their course content and what they have learnt in order to be recognised as worthy of these prestigious honours.

In addition, the combination of courses that a student must take during their academic program must match their academic programme. The selected combination of courses must offer content at a high enough difficulty level in order for the qualification received to be meaningful and worth the time invested acquiring it. For a student doing a degree program in computer science, for example, it would make no sense for the majority of courses the student is doing to be related to biology or engineering. Courses should also

follow on from each other, from year to year, to enable students to use the knowledge gained in a previous year in order to appreciate and solve more challenging problems in the years still to come. Various university faculties therefore have a tremendous task creating a degree structure that not only makes the degree competitive nationally and globally, but also ensures that it is completed within a reasonable amount of time, to enable the students to enter the workplace, should they so wish.

Selecting which courses to do in an academic programme cannot be left solely at the discretion of decision makers in a university. While it is accepted that some courses are absolutely necessary in a particular field in order to gain crucial knowledge and skills, students must also be allowed some level of freedom in selecting the courses they would like to do. Students may have their own personal tastes and interests, and may favour doing certain optional courses over others. Unfortunately, they might not have a full understanding of what each course entails, in terms of its breadth and depth of content coverage. They would thus not be able to make an informed decision in order to maximise both their enjoyment of their studies and their chance of successful academic performance.

This introductory section serves to provide a greater understanding of this issue. In Section 1.1, we describe the subject choice issue that students face in a more detailed manner. This is followed by a rigorous justification of the importance of the issue at hand in Section 1.2, and why it is necessary to carry out involved research on such a topic. Next, an outline of the methods intended to be used in this research is given in Section 1.3. Finally, the overall structure of this paper is described in Section 1.4, and the chapter is ultimately concluded in Section 1.5.

1.1 Definition of Problem

As mentioned earlier, choosing an effective and stimulating set of courses is not an easy task for a student. There are several factors at play when it comes to choosing courses that one must study. One of these factors may be the assumed difficulty of a course that a student is considering to take (O'Mahony and Smyth, 2007). Of course, if the course is compulsory, then the student has no choice but to enrol in it. However, in the situation in where there are many different subjects to choose from, the student may shy away from taking optional courses that might pose a significant challenge, in terms of workload or being unable to fully understand the course content.

These courses would clearly have a negative effect on academic performance. However, there may also be some students who are looking to be challenged, and for whom choosing more difficult courses would be an exciting challenge.

Another important consideration is to be aware of and take into account the student's own personal interests (O'Mahony and Smyth, 2007). University study does not necessarily have to be restricted to only courses within the same subject area. Since a student will already have taken a number of courses that are compulsory, he or she may decide that they would like to take a course or a number of courses from a different field in order to broaden their skills and make their knowledge more well-rounded. However, this may come at the cost of students not learning some topics that may be useful for their understanding in later courses in their specific field. Thus, it is clear that there is a trade-off between interests and required subject knowledge that a student has to make when deciding what subjects to enrol for.

In addition, there are other external factors to consider that may not be directly related to a student's personal preferences. For example, a very wide selection of possible courses may impose a burden on students. A student may not make a fully correct decision as a result of having too many courses to choose from (O'Mahony and Smyth, 2007). Students may miss out on crucial courses relevant to their academic interests, and may thus not be able to do the courses they wish to do. Also, the relative importance of certain courses may not be fully understood, perhaps due to insufficient course descriptions. In the case where students want to take courses closely related to their particular career path, certain courses in that field may be more relevant than others which may not be immediately apparent to the student, resulting once again in poor decisions. The chances of students failing as a result of poor decision making on their part would increase. Hence, a more effective way of selecting courses is needed, both for student satisfaction in terms of what they would like to learn, as well as to guarantee their likelihood of good academic performance, in terms of marks and grading. Undoubtedly, the problem discussed is extremely important and relevant to any particular university that accepts new students. A more detailed discussion of the importance of this problem is provided in Section 1.2.

1.2 Importance of Problem

Following the discussion of the problem in Section 1.1, the issue discussed can be summarised in two ways: the happiness of students with their courses, and

the likelihood of them being successful in their studies. Clearly, these two factors have implications not only for students, but for the university they attend as well. A student who is unhappy with the material being taught in a course may choose to deregister from it. This may result in the student having to wait until the next semester or year in order to register for another course, which results in more time being wasted. From a university's perspective, any self-funding students who choose to leave the university would result in a loss in fees that would have been paid, not only in that particular year but in future years as well. Furthermore, negative feedback from dissatisfied students may cause potential students to decide not to enrol in a course or at a particular university. This sentiment is reflected in (O'Mahony and Smyth, 2007), where it was shown that friends' opinions have a significant impact on a student or potential student's course and university selection. This negative publicity could have wide-reaching consequences.

In terms of academic performance, effective course selection is of utmost importance in ensuring a student is able to succeed in her or his studies and obtain their qualification(s). Students that achieve good grades in all their years of study are likely to find work and proceed to have a successful career using the knowledge they have gained from their studies. This is also important for a university, since it is able to advertise the success stories of its students and continue to attract future students. On the other hand, selecting irrelevant courses or courses that do not combine well together can cause students to fail or drop out of university, especially if they cannot afford to repeat due to financial problems. A large number of failures or dropouts reflects badly on a university, causing it to lose its reputation and its ability to stand out from other institutions. Hence, any improvement in the way students select courses is likely to have a profound effect on their success rate and the standing of their university, and thus all stakeholders would benefit.

1.3 Overview of the Research

In order to come to a solution for the issue of course selection, it is important to consider what data is available that could potentially be useful, and how to interpret and interact with this data. Clearly, there would have been a large number of students that have registered and succeeded in the same fields in the past. By looking at the combinations of courses some of these students took, and their performance in these courses, it is possible to determine certain course combinations that worked particularly well, and which course combinations were likely to result in failure. Information on courses

that were often selected together by similar students in the past can also be of use in determining the combinations of courses students were generally interested in. This information can then be used to enable students to select courses in a less cumbersome manner.

To implement this approach, we used a historical dataset containing information about past Computer Science I students. This dataset, sourced from the School of Computer Science and Applied Mathematics, contains relevant data pertaining to past academic performances of students over a number of years and the courses that each student enrolled in. We processed this data using various transformation procedures in order to obtain a clean, working dataset on which analysis could be performed. Using the valuable facts gained, we then developed a collaborative course recommender system, which can be used by students to make more informed decisions on the courses that they will potentially register for, providing them with detailed information on relevant courses they can take as well as their chances of passing when selecting particular course combinations. This recommender system was developed as a web-based Java program, which students can access from their computers or mobile devices with ease, in order to make their choices in a timely manner.

1.4 Structure of the Document

The following chapter, Chapter 2, takes a detailed look into the background knowledge required for understanding the important concepts of recommendation and recommender systems, including the design and implementation of course recommender systems. We detail each type of commonly-used recommender system, including advantages and disadvantages, and provide examples of their application. A collaborative recommender system was implemented in this study. The various algorithms that are used to implement recommender systems are explained in detail, in order to provide an understanding of their inner workings and why they are used. We highlight the use of these algorithms in some recent papers in order to demonstrate their effectiveness. Finally, we go into detail about Apache Mahout, the platform we used to create the recommender system. The platform is Apache Mahout.

In Chapter 3, the motivation for this research is presented. We also state, in detail, the research questions that we answered in this research, and justify the necessity of investigating them. The system design for the end user is outlined, and the methods with which we implemented our intended system

and as a result answered the research questions are also given.

In Chapter 4, we provide an in-depth look at the results of our research. We describe the statistics of the dataset we used, and we then move to the results for each specific research question we posed in Chapter 3. The design of the web application we created is also examined in detail.

Chapter 5 discusses the obtained results, going into detail about why certain results came about. We also discuss what effects the implementation of the system would bring.

Finally, Chapter 6 concludes the report, with an overall look at the study we carried out, as well as the types of future work that could add value to the research that was done.

1.5 Conclusion

Universities play an important role in preparing their students for their future careers. One of the most important considerations to be made is that of course selection, not only in terms of students' individual learning desires but also for practical reasons related to the required knowledge in order to succeed, both in their university courses and in later life. An ill-informed combination of courses can impact negatively on the students themselves, and also on the university. It is thus in the university's best interests to implement a system that can recommend effective course choices to students, in order to ensure that success is achieved.

Chapter 2

Background and Related Work

2.1 Introduction

The issue of course selection in university can pose a major challenge for students. Students may have to choose between courses restricted to their subject area and optional courses they may be interested in, for example. There is a need to develop a system that can assist students in making these decisions. For this purpose, recommender-based approaches are a worthwhile consideration for implementing such a system. There are different types of recommender systems, each having their positives and negatives. In addition, the platform on which to implement a chosen recommender system is also important to investigate. It is therefore important to discuss and understand the different types of recommender systems, and the platform that was used to implement the type of recommender system selected for this research.

In Section 2.2, we provide a comprehensive overview and discussion of the different types of recommender systems available, with subsections 2.2.1 to 2.2.3 each focusing on a different type of recommender system. Next, Section 2.3 looks at applications of recommender systems in various fields. In Section 2.4 we describe in detail the inner workings of collaborative filtering systems, with subsections 2.4.1 to 2.4.4 describing the workings of the algorithms involved. Section 2.5 looks at past work on collaborative recommender systems, with a focus on their use in educational settings. Section 2.6 uses the concepts introduced in earlier sections to describe and discuss Apache Mahout as the platform for the implementation of the recommender system. Finally, Section 2.7 concludes the chapter.

2.2 Overview of Recommender Systems

Generally, the main purpose of recommender systems is to make useful determinations on possible links between the user of the system and some objects or items based on some form of cleaned input data, and thus to output these particular findings. These outputs may be in the form of an ordered list or a singular value, depending on whether the person is looking for a list of recommendations or a prediction (Vozalis and Margaritis, 2003). Such outputs or findings can then be used in such a way as to assist the user in making decisions.

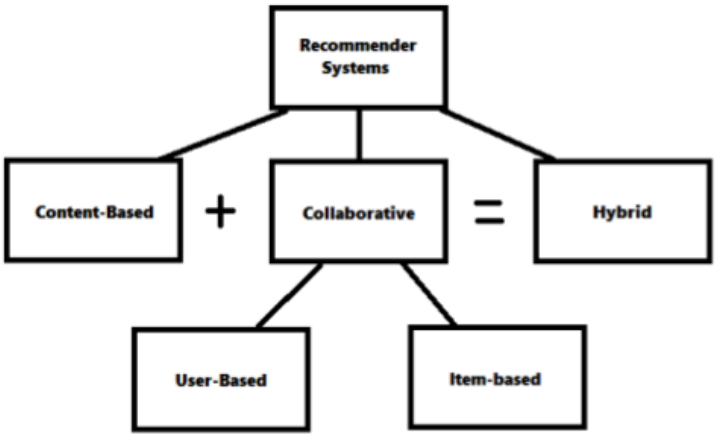


Figure 2.1: Types of recommender systems (Author's own work)

Figure 2.1 above displays an outline of different types of recommender systems. There are content-based, collaborative and hybrid recommender systems. In this research a web-based collaborative recommender system will be implemented to recommend courses to first time first year Computer Science students at the University of the Witwatersrand.

2.2.1 Content-based Systems

One of the most important and widely used types of recommender systems is the content-based filtering system. In this recommendation approach, the system uses data based on previously-recorded facts about a particular user, or any other relevant data currently contained by the system about the user. The system attempts to find similarities between the available data in order to effectively recommend a certain object or item to the user (Adomavicius and Tuzhilin, 2005).

In the context of course selection systems implemented in universities, for

example, these types of recommender systems may look at courses the student has taken in previous years in order to provide a better recommendation of current courses. The system may also use keyword selection to look at course descriptions to find courses that are similar to those already selected. O'Mahony and Smyth (2007) demonstrated a similar approach in creating a (More Like This) recommender system, in which course descriptions were scanned and compared with descriptions of other courses within the course database, with the most similar courses being selected and ranked.

Clearly, this type of recommender system has the potential to be very effective in performing recommendations, as it offers a relatively simple and yet practical means of gathering and comparing information. However, there are also a number of limitations. The approach will work well with more simple features such as text. However, other types of data such as real world data may be difficult to process and understand accurately by the system without significant time and processing power, due to the very dynamic nature of the information (Adomavicius and Tuzhilin, 2005). Also, the system may only recommend a set of courses from one field or faculty, and thus may miss out on some courses that could be valuable but are not directly related to the subject field. In this case, the system may be made to filter out the most similar suggestions in order to introduce a wider range of options.

2.2.2 Collaborative Systems

In contrast to content-based filtering, collaborative filtering methods do not rely solely on historical data about the user of the system and what their past decisions were. Instead, collaborative systems recommend items to the current user by using data containing the preferences of similar users, or similarities between different items in terms of ratings given to them by users. These constitute the two different types of collaborative recommender system-user-based recommendation and item-based recommendation.

In user-based recommendation, similarities between the current user and other users are examined in depth (Adomavicius and Tuzhilin, 2005). For example, this may include similarities related to types of goods purchased at an on-line store, or similarities in the area of education and similarities between courses taken by different students. Users that give similar ratings to the same products or services are grouped together, and the system determines that other items that one of the users liked are also likely to be enjoyed by the second user. Useful, informative correlations between these users are thus found. Using the data obtained from this, it is then possible for

the system to make an informed prediction or recommendation that is likely to be in some way helpful to the current user (Adomavicius and Tuzhilin, 2005). Using the previous examples, goods that the consumer would likely be interested in, or courses that a student might want to enrol for would be suggested. Thus, the end result is that the user is therefore be assisted in making an effective decision of their own, based on the particular field that the system is being used in.

In contrast, item-based recommendation methods take items themselves into consideration, rather than users (Adomavicius and Tuzhilin, 2005). The similarities in ratings between different items are examined and items that are rated similarly are grouped together. The system then compares these items to any items the user has already rated in order to find similarities, and if possible, recommend these items to the user.

Figure 2.2: Example of Item-Based Collaborative Recommendation

Figure 2.2 [1] shows an example of the results displayed by an item-based collaborative recommender used by Amazon to recommend similar items to the user, by considering the item the user is currently viewing (Linden *et al.*, 2003). The goal is to maximise sales and profits by predicting what products current customers will be interested in, based on past purchases made by similar customers. This method of advertising has had a significant impact in growing Amazon to a leading on-line retailer (Linden *et al.*, 2003).

Adomavicius and Tuzhilin (2005) state that one of the major advantages of using collaborative systems is the fact that, due to their approach of solely considering similarity with other users' choices, the recommender system does

[1] https://www.google.co.za/search?q=amazon+recommender+engine

not require a baseline understanding of the actual content of what is being recommended. This is in contrast to content-based systems, which look at actual keywords in text-based objects, as mentioned earlier. The variety of item categories that the collaborative recommender system can be applied to is generally much greater compared to content-based systems, since the raw content of the items is not being considered (Adomavicius and Tuzhilin, 2005). Collaborative filtering approaches do have some drawbacks. Considering the fact that collaborative filtering can be applied to a larger number of disciplines, certain disciplines with a very large number of users and/or items would require a large amount of processing power in order to scan through the available data and find effective recommendations (Vozalis and Margaritis, 2003). Also, the approach may potentially leave out a large number of items which would be relevant to the user. The filtering method only considers objects that have been favoured or selected by other users in the past, and so any items that have not been selected before are not considered (Vozalis and Margaritis, 2003). Collaborative systems may therefore not always be preferable over content-based approaches and the specific problem in question must be analysed carefully before selecting a type of system to use.

We outline the implementation details of these systems in greater detail in a later section.

2.2.3 Hybrid Systems

Content-based and collaborative systems have their advantages and disadvantages, some of which overlap with each other. A significant constraint with one filtering type, such as restriction to text-based information in content-based filtering, may be entirely negated by the other type of system, as in collaborative systems being able to process more diverse types of data. Some form of combination between these two types of systems may therefore have a positive influence on the outcome of the recommendation process. Systems that combine content-based and collaborative components are known as hybrid systems.

There are a number of methods used to implement hybrid systems (Adomavicius and Tuzhilin, 2005). The first and most obvious method is to simply transfer relevant features of one system into the other, creating a partially combined system and feature set. The system may compare similarity amongst all users for a set of items, but at the same time, can also take into account the past decisions made by the user in order to arrive at a

more well-rounded approach. In this way, common problems associated with collaborative systems can be eliminated, such as when items that have not been selected by other users before are being ignored.

Another approach is to simply obtain the results of a content-based filter on the dataset and environment, and thereafter get results from a distinct collaborative approach as well. These two sets of results can be considered together to arrive at a final recommendation set (Adomavicius and Tuzhilin, 2005). An effective method when concatenating results would be to set different weights for each set of results based on the current environment of data (Vozalis and Margaritis, 2003). Depending on the number of items in the database that have been selected by the user-base, the system can decide whether to place more or less weight on the results of the content-based or the collaborative filtering, as collaborative results would be more meaningful at a stage when the vast majority of items in the database could be accessed via their relation to users having previously selected them, for instance. It can thus be seen that there may be good reason to use hybrid systems in certain situations.

Having discussed the different recommender systems, in Section 2.3 we focus various applications of recommender systems in different fields.

2.3 Applications of Recommender Systems

Content-based, collaborative and hybrid recommender systems are all used extensively in a wide variety of disciplines. Lee and Hosanagar (2016) have noted the impact of recommender systems on conversion rates (causing potential customers to decide to buy products) in electronic commerce areas. They found that rates of conversion are indeed increased after recommender systems have been implemented in companies' web-pages, citing a near-6% increase, which is a significant amount especially for large businesses. In addition, the possible use of recommender systems in place of traditional customer reviews to help customers in decision-making was found to be viable. It can be seen from this that the use of recommender systems in different scenarios is worth considering. In the following subsections, we look at successful implementations of recommender systems in different applications.

2.3.1 Applications in Video Streaming

Perhaps the most common area where recommender systems are relevant is the Internet sector. Amazon has already been discussed as being a prominent example of successful recommender system implementation. Another major Internet-based application of recommender systems is Netflix, a subscription-based video streaming service (Gomez-Uribe and Hunt, 2016). As users browse the selection of movies and television shows that Netflix offers, they are given recommendations of particular shows to watch. This recommendation procedure is based heavily on the hybrid approach discussed in Section 2.2.3. Netflix takes into account both the user's own past viewing preferences, as well as the similarities between different users based on common shows watched. Effective recommendations are thus able to be generated based on this information. Gomez-Uribe and Hunt (2016) as well as Lee and Hosanagar (2016) have made similar findings regarding the effects of recommender systems on business success. They explained that, as a result of recommendations, users that would have otherwise chosen not to renew their subscriptions to the service have instead carried on paying for the service due to having a continuous flow of interesting shows to watch. Thus, extremely large amounts of income continue to be generated by the service.

2.3.2 Applications in Music Selection

Another field with possible applications of recommender systems is the Internet-based music market. In particular, helping users to choose songs they would like to listen to or purchase is a major focus. iTunes is an example of a successful implementation of such a system (Yoshii *et al.*, 2008). A collaborative approach is used, wherein the purchase decisions by users who bought sets of music tracks are looked at when proposing products to similar customers. Yoshii *et al.* (2008) proposed an update to such traditional systems, by introducing a hybrid system. Before, recommendation was based primarily on content-based features of music, in particular, the sound waves contained within the musical piece. Aspects of the sound waves were broken up into distinct features for the system to analyse, and when combined with the genre of music the piece was sourced from, recommendations were able to be generated. In addition, collaborative-based features, such as common music choices between users were also examined and factored in to increase the accuracy of recommendations. This hybrid approach was found to be able to generate reliable recommendations, and thus would likely be useful in business-related applications similar to those discussed earlier, in order to boost profits.

2.3.3 Applications in Public Transport

Recommender systems also have the potential to be successfully applied in the field of public transport. Yuan *et al.* (2013) proposed a system whose purpose was to make the process of locating a taxi more streamlined both for the end user and for the driver, economically as well as logistically. Major tasks included finding approaches that made taxi driving more financially viable for drivers, as well as increasing convenience to customers. A method for doing this was described, which involved finding optimal routes to the customer's pick-up location as well as factoring in elements that are likely to result in both the driver and the customer waiting for the smallest amount of time possible. Since this is a collaborative recommender system approach, the main data used was obtained from past data about similar situations, looking at which routes proved to be the most effective in the largest number of cases. From tests carried out in real-world situations, Yuan *et al.* (2013) found the system to perform reliably in terms of generating effective route and pick-up location suggestions, allowing both the driver and customer to benefit.

It is thus easy to appreciate how valuable recommender systems, in general, can be in a variety of disciplines. In Section 2.4, we provide a detailed discussion of the collaborative systems approach, which will be used in this research.

2.4 Algorithms in Collaborative Systems

2.4.1 Practical Use of Collaborative Systems

The limitations of collaborative systems were discussed in Section 2.2.2. Considering course enrolment systems, some of these limitations are mitigated, making the use of collaborative systems an attractive option in this respect. To be more specific, a university is unlikely to introduce a significantly large number of new courses into the system in any given year, meaning that the problem of new courses being ignored due to not having been selected by students before is relatively minimal. The system could even be modified to give higher priority to the recommendation of these new courses, an approach that could combat this problem.

2.4.2 Association Rule Mining

Before delving deeper into collaborative systems, we start by discussing the concept of association rule mining. Vozalis and Margaritis (2003) provide a concise discussion of this topic, while Lee and Cho (2011) give a more detailed explanation. Essentially, an association rule describes the association or relationship between a number of items in a dataset based on the frequency with which they are selected or otherwise chosen at the same time by the person interacting with the system. To explain further, two or more items in a dataset that are often selected simultaneously by a user are said to be associated (Lee and Cho, 2011). Association rules can be particularly useful in collaborative recommender systems, since recommendations can be made based on the association rules generated using a large number of items and their related user-base.

These associations are created based on a few relevant criteria. The first that will be detailed here is the support value (Lee and Cho, 2011). The support value of an item or set of items correlates to how often that particular item is actually chosen by any user, out of all instances in which users have chosen items. It is useful to obtain these support values since they can be compared with some minimum support value that has been selected as the threshold for purposes of making recommendations. Any items whose support value falls below this threshold are disregarded. In other words, low-scoring items that are extremely unlikely to be related to the selections of the user can be effectively removed, making the process more efficient.

Another important value to calculate in association rule mining is the confidence value (Lee and Cho, 2011). This value defines the likelihood that a user selects two or more particular items or sets of items simultaneously, i.e. the likelihood of association between the two items. Similar to the support value metric described above, a minimum confidence level is also chosen arbitrarily. After selecting items above the support value threshold, confidence values are then calculated for these items and items with a high enough confidence are then selected for recommendation. The recommendation algorithm produces a final shortlist of items (i.e. a set of 10 items, for example) by ordering them in descending order, based on their confidence levels (Vozalis and Margaritis, 2003).

2.4.3 User-Based Collaborative Algorithms

User-based collaborative filtering methods focus on the similarity between different users and their selected items, as opposed to some methods that focus on the similarity between items. Memory-based methods such as collaborative filtering utilise a matrix containing the user/item associations from past data, thus resulting in predictions that are likely to be accurate. These types of methods do not use probabilities and expected values, as is commonly found in model-based methods (Vozalis and Margaritis, 2003). Calculations are performed with the assumption that not all users have a preference for all items. To create more reliable recommendations, any of these (empty) associations are either replaced with an arbitrary constant similarity value or an average value is inserted based on the user's preferences for other items (Vozalis and Margaritis, 2003). Once this pre-processing is complete, calculations of user-user similarities can be performed.

Table 2.1: A typical set of users and their ratings for items

	Amy	Bob	Chris	Dean
Apple	4	2	3	1
Banana	5	2		
Cake		7	7	
Donut	8	6	5	
Egg	9	5	6	2

Table 2.1 above illustrates the type of data that would be processed for a typical user- or item-based collaborative system. Each user has a rating for a particular item, and users with similar ratings are expected to be matched. The purpose of these matchings is to recommend a new item to a user that the system expects the user would like, based on the fact that the user this user is paired with likes this item. The absence of some ratings for items increases the risk of inaccurate recommendations being produced, due to lack of data.

The most widely-used similarity metric that would be relevant in this context is the Pearson correlation similarity metric. The Pearson correlation similarity coefficients can be calculated using the following formula presented by Vozalis and Margaritis (2003):

$$sim_{xz} = \frac{\sum(r_{xy} - \bar{r_x})(r_{zx} - \bar{r_z})}{\sqrt{\sum(r_{xy} - \bar{r_x})^2}\sqrt{\sum(r_{zx} - \bar{r_z})^2}} \tag{2.1}$$

where r refers to the user's rating for an item, and sim is the calculated correlation coefficient, or similarity. The similarities between the active user and every other user within the sample space are calculated using this formula, based on the ratings given to items by each user.

Ekstrand *et al.* (2011) highlight a key issue concerning this approach when used in a collaborative filtering environment. This formula does not take into account the number of ratings when comparing ratings of one user to another. It will mark users as being related in terms of correlation even if they have only given a very small number of items a similar rating. Clearly, this could in some cases create inaccurate predictions or recommendations. If more data was obtained about these users in the future, it could reveal that there are actually large differences between the preferences of the two users in question. As a result of this, Ekstrand *et al.* (2011) propose a solution to eradicate this problem. A minimum number of similar ratings required for users to be marked as similar can be introduced. Ekstrand *et al.* (2011) cite a value of 50 similar ratings being a sufficient number. However, this would need to be modified depending on the actual size of the dataset, and could either be increased or decreased. As is always observed in other environments, the larger the dataset, the better the accuracy.

A better similarity measure to use when the relationship between user ratings does not follow a normal distribution is the Spearman correlation (Mukaka, 2012). In this correlation approach, the ratings that a user has given for each item are compared in order to determine the order of that user's ratings. The items are then arranged in descending order and given ranks in ascending order. Spearman correlation coefficients are calculated using the ranks assigned and the Pearson correlation formula (Ekstrand *et al.*, 2011).

The calculated Pearson or Spearman coefficients can then be used to create a matrix containing values demonstrating the aforementioned correlation between every user and every other user and their respective choices (Vozalis and Margaritis, 2003). From this matrix, the similarity between any two or more users can be easily determined. Thus, using this matrix, users that are very similar to the active user can be detected and grouped into a set, commonly referred to as a neighbourhood. This is performed up to a certain limit (ordered by the user with the highest similarity first). Thus, the best

possible predictions or combinations of items for the active user can be more easily found as a result.

In order to generate predictions, Vozalis and Margaritis (2003) describe a method that uses the average similarities of users within a neighbourhood. They use the following formula:

$$p_{ax} = r_a + \frac{\sum sim_{ay} * (r_{xy} - r_x)}{\sum |sim_{ay}|} \tag{2.2}$$

where r is the rating of an item, a and i are the active user and users in the neighbourhood, respectively, $\sum sim_{xy} * (r_{xy} - r_x)$ refers to the similarities of users in the neighbourhood multiplied by their respective item ratings, and $\sum |sim_{ay}|$ is the sum of the absolute similarity values. These combine finally into p_{ax}, which is the corresponding prediction on a new item to be made for the active user.

In order to generate recommendations, the top-N recommendation approach is used (Vozalis and Margaritis, 2003). Association rules are found for the users being considered. Support and confidence values will be calculated as described in Section 2.4.2. When all the confidence levels have been worked out with respect to the active user, N items ranked highest-first will then be selected (hence the top-N in the name of the method). As a result, accurate recommendations will be tabled and submitted to the user.

The item-based collaborative foldering approach is discussed next, in Section 2.4.4.

2.4.4 Item-Based Collaborative Algorithms

In direct contrast to the user-based filtering approach discussed in Section 2.4.3, item-based collaborative filtering focuses on similarities between items, rather than on similarities between users (Vozalis and Margaritis, 2003). Ekstrand *et al.* (2011) describe this approach in further detail, explaining the item similarities as being calculated depending on their favourability to two or more of the same users.

The item-based approach follows a similar set of steps to the user-based approach. Once again, similarities are computed using two or more items selected by the same user rather than separate users. Vozalis and Margaritis (2003) suggest a slightly modified version of the Pearson correlation formula to deal with this approach of generating recommendations. However,

Ekstrand *et al.* (2011) contrast the decision to use the Pearson coefficient, stating that it is not the optimal method for calculating accurate similarities in this particular case. Instead, they suggest using cosine similarities, as given by the following formula (Ekstrand *et al.*, 2011):

$$sim_{xy} = \frac{r_x * r_y}{\|r_x\|^2 \|r_y\|^2} \tag{2.3}$$

where r_x and r_y are ratings given by user x and user y, respectively. Item similarities are thus generated for each pair of items using this formula.

Finally, recommendations are made using association rules. In the case where predictions are desired rather than recommendations, a weighted sum is calculated using the following formula by Vozalis and Margaritis (2003), which is slightly different from the previous prediction formula seen in user-based recommendation:

$$p_{ax} = \frac{\sum sim_{xy} * r_{ay}}{\sum |sim_{ay}|} \tag{2.4}$$

where $\sum sim_{xy} * r_{ay}$ refers to the sum total of the active user's (a) ratings multiplied by similarities between other users x and y, $\sum |sim_{ay}|$ is the sum of the absolute similarity values, and p_{ax} is the corresponding prediction on a particular item to be made for the active user, based on ratings given by other users. Accurate predictions of a user's preference for an item are generated using this method.

2.5 Related Work

The wide-ranging applications of recommender systems were discussed in Section 2.3. This section focuses on a discussion of related work, in which recommender systems have been used in educational settings or contexts.

In Section 2.4.2 we discussed the application of association rules in recommender systems. Lee and Cho (2011) made extensive use of association rules in their work. They created a mobile course enrolment system based on the recommendation approach. Students and their course selections were assessed. Specifically, the instances of students taking a particular course and also taking another course were recorded. In testing their implemented recommender system, minimum support levels of between 35 percent and 50 percent were used, with relatively unpopular courses being removed from consideration. A confidence level of 80 percent was chosen, meaning that a

relatively high probability of students choosing a specific course combination was being considered. A large numbers of association rules were discovered, both for compulsory and optional courses, enabling recommendations to be generated. The system proposed by the authors had a high accuracy rate on recommendations. Thus, the work by Lee and Cho (2011) demonstrates that association rules can be effectively used in recommender systems.

A slightly similar set of methods were used by Aher and Lobo (2012) to develop an effective course recommender system. The collaborative method used was based mainly on student interests, and did not take into account other factors such as chances of academic success. Data about each student as well as the courses they were interested or not interested in was collected. The data was then processed using the k-means clustering algorithm, in order to determine which courses were commonly grouped or selected together. Specifically, courses that were commonly liked or disliked simultaneously by multiple students were considered. This allowed for courses that were associated to be identified and analysed. The discovered association rules could then be applied to a recommender system to persuade or dissuade students from taking certain courses based on courses they had selected so far.

Another study was carried out by Sacin *et al.* (2009) related to implementing a collaborative recommender system. In contrast to the study by Aher and Lobo (2012), this particular paper had a greater focus on the potential academic success of students, rather than their specific interests. In the paper by Sacin *et al.* (2009), the system developed was intended to determine the likelihood of success of students in the courses they were taking, based on the historical data obtained from other students with similar course combinations. After preparing the data for analysis, the study then determined correlations between students, first using a portion of the available data (the training dataset) in order to develop correlation parameters, and then testing the system on another data portion (the test dataset) to evaluate its effectiveness. Upon testing, it was found that the system was able to generate accurate predictions of academic performance in almost 78% of cases. This research is significant in presenting results on the more useful academic performance metric.

Much work has been done on the application of item-based collaborative systems, particularly in the education sector. O'Mahony and Smyth (2007) proposed a university course enrolment system based on the principles of item-based collaborative filtering. The need for this system was justified by the fact that the large number of available courses made it difficult for

students to select the most suitable courses, based on their own personal interests versus the diversity of knowledge required in their particular degree path. The recommender system implemented involved comparing and finding similarities between sets of elective courses chosen by particular students in combination with their core courses. This was weighted with past choices of previous students within the same field, eventually producing a set of top-N recommendations, listed in order of the system's estimated chance that a particular course will be an effective choice for a student. The majority of recommendations made by the system were found to be in line with the expected choices, when run on test data. The system was able to make accurate recommendations to the user based on the core subjects they had chosen, highlighting the effectiveness of this item-based approach. Both user-based and item based systems are effective, as shown by on the results obtained by Sacin *et al.* (2009) and O'Mahony and Smyth (2007).

A related approach was adopted by Thai-Nghe *et al.* (2010). However, the chosen form of system was a prediction generator. The purpose of the system was to predict the academic performance of students in on-line tutoring environment. The approach attempted to predict the first attempt success rate of students on questions given by the tutoring system. Past students and their successes on questions (items in this case) were measured and used to generate predictions for the active user. The results of this test were compared with predictions generated using the traditional logistic regression method. The system performed significantly better than the logistic regression-based method. The positive results from O'Mahony and Smyth (2007) and Thai-Nghe *et al.* (2010) demonstrate the usefulness of recommender systems in educational applications.

Our literature review has demonstrated the overwhelmingly positive findings related to the use of various types of recommender systems in educational settings. There is therefore a case for using such systems in educational environments that do not currently have similar systems implemented. In Section 2.6 we discuss the Apache Mahout platform, which was selected to implement the collaborative course recommender system.

2.6 A Platform for Recommender Creation: Apache Mahout

Apache Mahout is a Java-based platform created for the purpose of implementing common machine learning algorithms (Ingersoll, 2009). It is a freely extensible environment, allowing for improvements and other general modifications to the algorithms it contains. It is intended to be relatively easy to use and commercially friendly, enabling the creation of programs for business and other related purposes. Within the realm of machine learning, Mahout focuses heavily on aspects related to recommender systems, particularly of the collaboration type. Mahout can be expected to be an appropriate library to use in creating effective course selection applications for the educational sector.

Regarding the collaborative-related algorithms contained in Mahout, Seminario and Wilson (2012) provide a detailed breakdown. From their investigation, it was found that Mahout does implement algorithms critical for the creation of collaborative recommender systems. These include the Pearson correlation similarity formula for calculating similarities between users and items, as well as the respective user-based and item-based prediction and recommendation generation formulae, as described in the previous sections in this chapter.

2.7 Conclusion

This literature review has provided a detailed discussion of the key concepts used in the area of recommendation and recommender systems. The research related to recommender systems and their implementation in different applications, including the Apache Mahout platform, was also discussed in great detail. The major types of recommender systems were discussed, with content-based systems only using data and keywords about the user, and their past choices. This made using non-text-based data an issue, as well as resulting in only a smaller range of possible courses being considered. Collaborative systems, in contrast, were defined as comparing similarity between different users, which solved the problem of different data types being an issue. However, items not selected by users previously have a risk of not being considered. Types of hybrid systems, combining parts of collaborative and content-based filtering, were discussed, including systems that simply combine the results of both, as a means of solving nost of the problems mentioned earlier.

Examples of recommender systems used in various disciplines were given. The review then went into the finer details of collaborative systems, first explaining association rules and their generation using the support and confidence metrics. Their application and implementation in both user-based and item-based algorithms was discussed, including the use of metrics such as the Pearson correlation and Spearman correlation in determining similarities between users and between items, for the purpose of prediction or recommendation. Following this, examples of recommender systems used in the context of education were examined, with the conclusion being made that such systems were definitely suitable in course enrolment environments.

Finally, the review focused on the use of Apache Mahout in the implementation of collaborative systems. The observation that Mahout contains the necessary algorithms for our research was made, including calculations of correlation as well as user- and item-based recommendation generation algorithms (Seminario and Wilson, 2012).

Chapter 3

Research Method

3.1 Introduction

The currently available knowledge on the use of an effective course enrolment platform has been obtained, through a thorough review of the literature in Chapter 2. In particular, we reviewed the relevant recommender systems, focusing in detail on their inner workings and the type of system that will be implemented in this project. In the context of a university enrolment system, the potential usefulness of a collaborative recommender cannot be overstated. As (O'Mahony and Smyth, 2007) and (Thai-Nghe *et al.*, 2010) discovered in their research, this type of recommender performs substantially better than any generic, non-student-focused method of providing course information to students. Since it examines the relationships between past students and the combinations of courses they selected, new students with similar characteristics are likely to select similar courses on registration. Also, using a web-based approach in developing such a recommender system is likely to result in a relatively easy-to-understand end product that is flexible, and that students are likely to find worth using.

There are a number of factors to consider regarding the development and testing of a course recommender system. The system must be able to recommend a set of courses that achieves a number of goals. The primary focus is, of course, to improve student understanding of the options available to them as well as to narrow these down to their areas of interest. However, the likelihood of achieving good results based on such a set of selected courses must be factored in as well. The design of the system must thus be robust enough to deal with the problem at hand from all different angles and to provide a solution with minimal drawbacks. In this research several research

questions are posed, which focus on different aspects related to the design, development, implementation and evaluation of the collaborative recommender system. The methods that were used to answer these questions are also discussed.

Section 3.2 presents the questions that have been formulated for the purpose of this research. Next,Section 3.3 presents the proposed system's design. Section 3.4 goes into detail regarding the methodology that was used in order to answer the research questions posed, including data preparation, correlation finding, and the methods used to implement the recommender and the web application. Section 3.5 discusses the specific tests that were performed to assess and evaluate the accuracy and reliability of the system and its environment in order to answer questions about its suitability. Finally, we conclude this chapter in Section 3.6.

3.2 Research Questions

The successful implementation of a recommender system requires several factors to be considered, as well as a significant amount of preparation. All the planned stages and the information available beforehand must be considered in order to arrive at final answers or decisions that determine whether the project is worth carrying out initially, and whether it has succeeded after analysing results post-implementation.

It is thus useful to look at the goals of the project as a whole. The exact requirements of the research being proposed, and the results expected by the end of the design phase are important to consider. The following main research question was posed in order to provide a clear focus and direction on conducting this study:

Main Research Question:

- *Is a collaborative recommender approach, developed in a web-based environment using Apache Mahout, suitable in order to suggest relevant courses to students, while striking a balance between the students' own interests and crucial field-related material in order to ensure academic success?*

At best, a recommender system can only feasibly provide an estimate of what the best courses for a student to choose are, based on the historical educational dataset provided to the system. If there is not sufficient data on

certain combinations of courses, then the system may fail to give a particular student a good set of recommended choices. In the most severe of cases, which are obviously avoided where possible, the system may even recommend courses that would possibly lead to failure. The question of whether the system can be implemented is relative, depending on the number of times the system is able to make a good recommendation. A large dataset results in a greater percentage of suitable recommendations.

There are other factors involved in this research question. One of the main components to the system is the commitment to ensuring academic success. To do this, the marks obtained by past students must be examined. It is then necessary to find which courses students scored lower marks in and which courses students excelled in. The system would need to take instances where students failed and decrease the likelihood of recommending such courses. Thus, when finally generating recommendations for prospective students, the system must be able to calculate a prediction of academic performance and relate it to links between courses.

In order to answer the main research question in a focused manner the following related sub-questions need to be answered:

Research Sub-Question 1:

- *What are the relationships between courses and performance during the year?*

Research Sub-Question 2:

- *Are there courses in which students perform particularly badly, in comparison to others?*

Research Sub-Question 3:

- *Which first year courses are the strongest determinants of a student's success or failure?*

Research Sub-Question 4:

- *What combinations of courses should be recommended in the students' first year of study?*

Research Sub-Question 5:

- *How good is the coverage of the implemented recommender system?*

Research Sub-Question 6:

- *How good is the accuracy of the generated recommendations?*

Research Sub-Question 7:

- *How sufficient was the amount of available data in terms of allowing good quality recommendations to be produced?*

In terms of data volumes, the sample data that is available is from a relatively wide range of years (2010-2015), with a large number of students registered each year. This means that a large number of course combinations are available for observation. The system is able to process this data and thus has a large knowledge base with which to work. Taking all this data into account, the system was expected to be able to use the algorithms in Apache Mahout (Seminario and Wilson, 2012) in order to calculate an accurate set of recommendations. Whether the system was able to achieve this is discussed in Chapter Chapter 4.

In addition to describing the questions that are involved with our work, we must also describe the methods that we used to answer these research questions.

3.3 End-User System Architecture

Figure 3.1 shows the architecture of the web-based application we intended to implement from the end user's perspective.

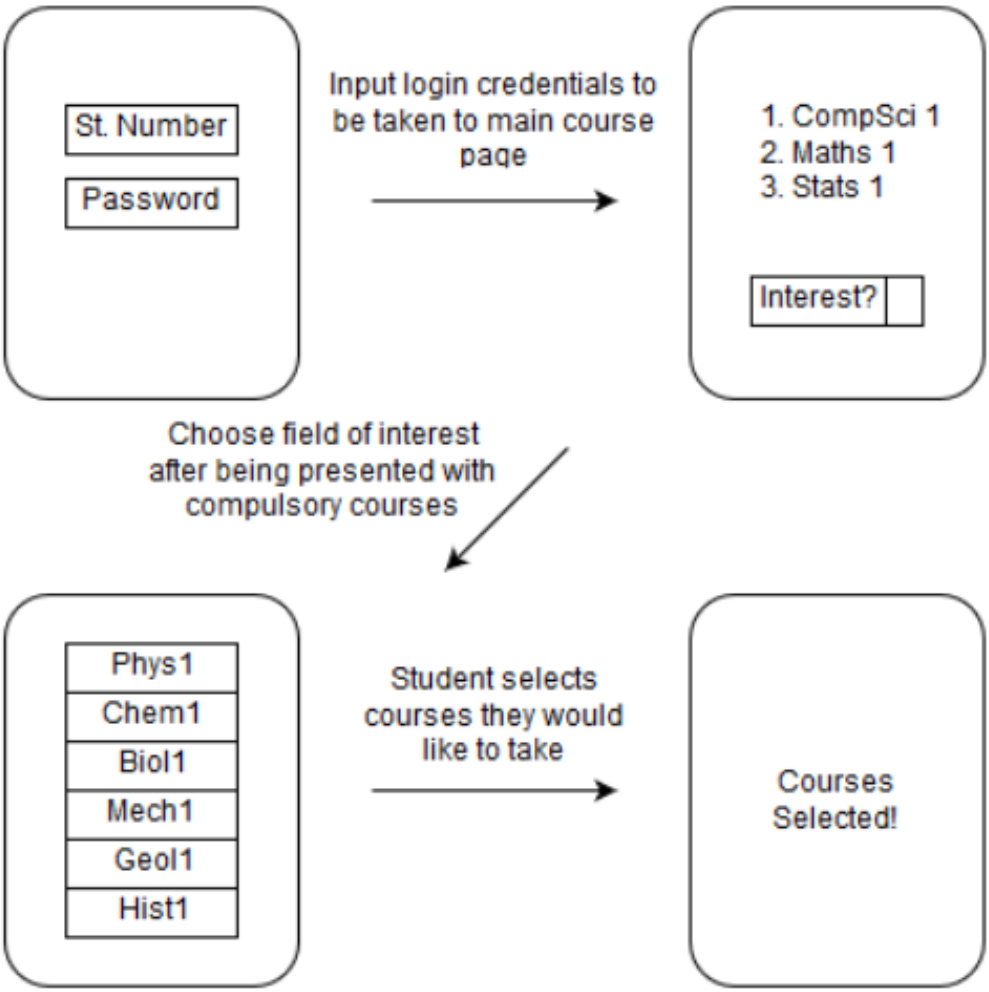


Figure 3.1: System design as seen by the end-user (Author's own work)

We designed the system with the following functionality in mind. A student is first required to log into the system, using a unique student number and password. Following this, the student is presented with compulsory courses related to their specific field. Once enrolled in these courses, the system then asks the student to choose a field in which they would like to select elective courses in. The system then looks at the student's core field and their chosen elective field and generates recommendations of elective courses, taking into account other students who have in the past selected similar combinations. However, the student is still allowed to choose other courses if they would like to do so.

3.4 Research Methodology

We now go into detail about the methods we used to develop the course recommender system and answer our research questions. Figure 3.2 below shows the components of the system that we intended to implement and how they link together:

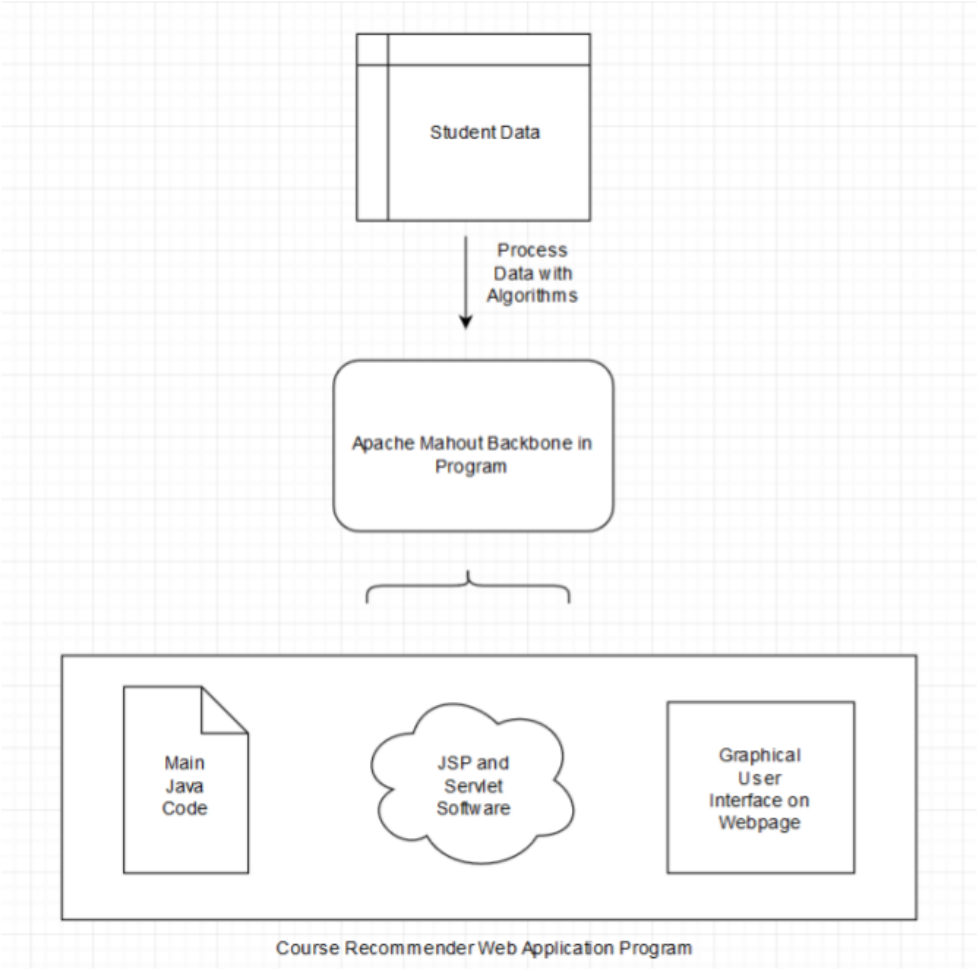


Figure 3.2: Proposed structure of system (Author's own work)

The historical student data, after being subjected to data preparation, was to be stored in a database and loaded for use by Apache Mahout. Collaborative filtering algorithms were to be applied to the data in order to generate a list of recommendations and grade predictions. These would then be sent to the relevant web-page of the application.

We now examine the methods used to create each component of the system in more detail.

3.4.1 The CRISP-DM Methodology

The successful implementation of our recommender system depended on the data being used to make predictions or recommendations. The word (data) is often thrown around loosely, commonly being used in inappropriate contexts. It is important to acknowledge the true meaning of the word, in order to eventually fully appreciate the true worth and necessity of data, not only in the field of recommendation but also in other contexts.

Ackoff (1989) describes data as being a set of facts that are related to some sort of real world construct, that the data originates from. However, data is not sufficient on its own. An uncategorized segment of numbers or letters, for example, does not hold much meaning or worth. There is thus a need to process and classify the data to create information. Information reveals the actual value contained within the data. It gives context to data, allowing it to be analysed and used for its intended purpose. Therefore, in terms of recommender systems, data about students must be interpreted and understood fully in order to perform meaningful actions on it, allowing the full effectiveness of recommender systems to be demonstrated (Ackoff, 1989).

One prominent model used for data mining and analysis is the CRISP-DM methodology (Shearer, 2000). This model breaks the process of data mining up into six distinct phases - business understanding, data understanding, data preparation, modelling, evaluation and deployment. We mainly focus on the first three of these phases for this research. The first of these, the business understanding, deals with understanding what the project aims to achieve and the specific business context the final product is being used for. In the case of this research, this would refer to the course recommender system that was developed for an educational context.

The data that was processed in this research is the student data containing multiple features about students that were enrolled in Computer Science I. This leads into the second phase - data understanding (Shearer, 2000). In order to extract valuable information from the data, a careful investigation into the data was carried out. Specifically, the consistency and reliability of the data was examined. Zhang *et al.* (2003) describe the main issues involved in this process. First, it was important to look at the number of missing values within the dataset. The presence of a large amount of missing data may skew the statistical analysis on the data. Also, in a dataset, there may be attributes contained within the data that are either not relevant or provide little additional information to the purpose for which the data is intended to

be used for (Zhang *et al.*, 2003). This is particularly relevant when considering the implementation of recommender systems and, as will be seen later, only very specific portions of the data are required. Thus, strategies must be developed to deal with these problems.

The next major phase of the CRISP-DM methodology - data preparation - deals with processing the student data that is used in this research (Shearer, 2000). This data has been obtained from the School of Computer Science and Applied Mathematics. The structure of the dataset is as follows:

Table 3.1: Dataset Characteristics

Attribute Name	Description	Range of Values
id	Individual identification of each student	1 and upwards
gender	Gender of student	M or F
race	Race of student	B, W, I, C
regYear	Year the student registered	09-15
finAid	Whether the student is on financial aid	Y or N
apsScore	Matric score	Integer value
nscAggregate	Average matric mark	Integer value
numCourses	Number of courses the student is taking	Integer value
accn1000 - stat1005	The student's mark in a particular course	0-100
wam	Weighted average mark	0-100
am	Average mark	0-100
cpAchieved	Course points achieved	Integer value
cpAttempted	Course points attempted	Integer value
cpAchievedCum	Cumulative course points achieved	Integer value
cpAttemptedCum	Cumulative course points attempted	Integer value
outcome	Year-end result code	PCD/RET/MBR/MBP...

As shown in Table 3.1, the dataset includes information on courses selected by previous Computer Science I students, as well as personal details about students which include age, race and student number. Each individual observation contains the corresponding final mark for each course that the student took, and there is a (-1) in the case where a student did not take a particular course. Methods for dealing with missing values are covered in detail in Pyle (1999). Essentially, certain values may be inserted in place of the missing data. Careful consideration must be taken to decide what

data values are used for this purpose. Pyle (1999) stated that calculating the standard deviation of a particular attribute's available values, and then inserting values that do not change the standard deviation significantly is an effective method for accounting for missing data. In addition, any attributes that depend on the attribute with missing values must also be considered when calculating these values (Pyle, 1999). Lastly, observations with missing values could also be removed completely if no effective solution is found when trying to replace values. An entire attribute could potentially be removed as well in case there are more missing values than can be reasonably dealt with. There is also the possibility of erroneous data or outliers being present in a dataset. This is data that lies outside of the expected minimum or maximum values of the attribute (Pyle, 1999). For example, in an educational dataset, a student's mark may be recorded as 150 when the actual value for that particular attribute only falls within the 0-100 range. Outliers can display similar characteristics, being within the correct range but occurring far away from most other observations. These abnormal values can be dealt with in a similar manner as missing values, where they are either removed or replaced with recalculated numbers. However, when talking about outliers in particular, unusual values may actually be genuine observations, as mentioned by (Osborne and Overbay, 2004). The authors describe a feasible method of dealing with such situations, which involves truncating the observations. Values are intentionally modified downwards or upwards to become equal to the maximum value of the major cluster of normal data values. This allows some of the weight of the outlier to be retained, while preventing it from affecting analysis of the data significantly. Finally, the last factor mentioned earlier is the potential presence of insignificant attributes or variables in the dataset (Pyle, 1999). To determine which variables do not provide extra information depends heavily on the problem the dataset is being used to solve. A dataset about students at a university might have various non-numeric values, such as name and surname, which cannot be used in any calculations. Thus, these particular attributes can be removed completely from the dataset. In cases where an extremely large amount of data is being used, removing these variables will significantly streamline the data. This will lead to faster processing times when algorithms are run on the data, allowing for results to be obtained more quickly.

3.4.2 Using SAS for Data Processing

Different software packages are capable of performing simple data cleaning and variable partitioning tasks. We selected the Statistical Analysis System (SAS) suite for this purpose (Delwiche and Slaughter, 2012). SAS has

built-in functions for data preparation, including functions to easily drop columns/variables from the dataset, as well as to detect any missing data within the observations. Rows with missing data can then be loaded with default or dummy variables or removed entirely to avoid skewing the results (Delwiche and Slaughter, 2012). Functions also exist to split the data into different tables based on certain conditions, such as when a student's grade is above or below a certain number. Finally, SQL queries can also be run on the dataset from within SAS for more advanced functionality if it is required (Delwiche and Slaughter, 2012). In terms of removing variables, it is clear to see that the most important variables in the data set are the subjects that each student is enrolled in, as well as their final grades in each subject. Features such as student number, age and race were removed from the dataset. The major focus was on courses and grades only for the purposes of the research. It was then necessary to transpose each of the variables in order to transform the data so that each observation included the student's identification, one course they took and the respective final grade. This format was necessary for use as input to Apache Mahout's algorithms (Ingersoll, 2009). The final table structure was as shown in Table 3.2 below:

Table 3.2: Transformed Dataset Characteristics

Attribute Name	Description	Range of Values
id	Individual identification of each student	1 and upwards
course	Courses the student has taken	ACCN1000 - STAT1005
mark	Student's mark in a course	0-100

3.4.3 Finding Correlations in the Data

In order to determine the correlations between different courses and performance, we used the functionality of SAS to calculate the average grades of each course. The *proc means* option in SAS takes the dataset and calculates the mean and standard deviation for every course that is within the data (Delwiche and Slaughter, 2012). We then used the *proc freq* functionality in order to produce frequency distribution plots of the marks in the compulsory courses, as well as values of skewness to see whether the data followed a normal distribution or not. Finally, we used the *proc corr* functionality in order to produce scatter plots of the relationship between the marks in every course and the weighted average marks of the students (Delwiche and Slaughter, 2012). The correlation values here allowed us to determine whether there

is a positive or negative relationship between these marks, and also to gauge the strength of such a relationship.

3.4.4 Installation of Apache Mahout

The most recent version of Apache Mahout, v0.12.3, was installed and used in analysing the data. The installation of Mahout on a Linux-based platform has a number of requirements (Ingersoll, 2009). Since Mahout is developed on the Java platform, the first step was to install the latest version of the Java Development Kit (JDK) onto the machine that was used for the research. Following this, the software framework Apache Hadoop was installed on the system. With these two components already installed, Mahout was then installed and used in the development of the proposed recommender system. Having set up the required software to produce our recommender system, we now move to the actual development of the application.

3.4.5 Storage of Data in a Database

A SQLite database was set up in order to store student data. We chose SQLite for this purpose as it is easy to use with the Java programming language and supports the major SQL functions that are needed, such as creating tables, inserting and updating data (Zhao *et al.*, 2015). It was necessary to create a (student) table in order to store data about new students such as their student numbers and personal information. A (courses) table was also necessary in order to store a list of all courses as well as each of their respective descriptions. These were intended to be displayed in the final web application. The students and courses tables could then be linked via an intermediate table in order to record which courses each student chose. Finally, a (history) table was required in order to store the dataset used in the project containing the previously prepared data.

3.4.6 Using the Functionality of Apache Mahout

The application development environment that was used in this research is the Eclipse IDE (Dai *et al.*, 2007). Eclipse provides an attractive interface in which a myriad of Java-related programs can be developed, including those that require external libraries such as Apache Mahout.[1] We

[1]The transformed historical dataset is loaded into Mahout and fed to the *PearsonCorrelationSimilarity*() and *SpearmanCorrelationSimilarity*() classes, which determine how similar each previous student is to every other student (Ingersoll, 2009).

tried both of these classes in order to see which would produce more accurate recommendations. The similarities obtained are then loaded into the $NearestNUserNeighbourhood()$ class in order to create a neighbourhood of similar users (Ingersoll, 2009). We experimented with neighbourhood sizes of 5, 10, 20 and 30 in order to see what effect this size had on the final recommendations. These neighbourhoods were then used with the $GenericUserBasedRecommender()$ class in order to generate a list of course recommendations for the new student (Ingersoll, 2009). Since we do not have previous course history about new students, it is not possible to compare course performances to find students with similar grades. Instead, we created a dummy student with high marks in compulsory courses, so that the system would find other similar students with high marks and recommend the elective courses they did well in.

3.4.7 Developing the Web-Based Components of the Application

The Eclipse IDE also provides support for a wide range of plug-in software, through which it can extend its functionality to support developers. One such extension is the Java Web Tools Project (WTP), which supports the development of web applications within the Eclipse environment (Dai *et al.*, 2007). We made use of this tool-set in our development process, as it contains both JavaServer Pages (JSP) and Servlet technology which allow for the creation of a web application. The Mahout functions that we implemented in Section 3.4.6 were integrated into a separate Servlet for each different page of the web application. We then designed the layout of each web page using JSP combined with CSS. We created separate web pages for logging in, displaying compulsory courses, displaying the recommendations generated by Apache Mahout and an additional page for listing all courses should the student decide not to follow the recommendations. In order to test the system, we created a testing server using Apache Tomcat in order to display the pages in a browser and evaluate their design.

3.5 Evaluating the System

As stated in Section 3.4.6, we tried both the Pearson correlation and the Spearman correlation to see which would produce better recommendations. The testing phase was split up into the following steps:

3.5.1 Evaluating Coverage

The coverage of a recommender system refers to the percentage of the total population that the system is able to make recommendations for (Vozalis and Margaritis, 2003). A high coverage value is desired to make sure that every student that uses the system is able to have recommendations generated for them. The historical data was used in order to evaluate the coverage of the system. We attempted to generate recommendations for each previous student in the dataset. We recorded a count of each student for which the system was able to generate recommendations, and then divided this by the total number of students. The formula for this is shown below

$$coverage = \frac{number\ of\ users\ that\ predictions\ were\ generated\ for}{number\ of\ users} \tag{3.1}$$

By using this metric, we were able to determine the reliability of the system in terms of making recommendations.

3.5.2 Evaluating Accuracy

In order to evaluate accuracy, we use the metrics of precision and recall. Precision calculates how many recommendations that were made were consistent with courses that students actually chose. Recall compares the number of correct recommendations made to the total number that could have been made based on a testing dataset. Both a high coverage and a high recall are desirable for our system. The formulae for precision and recall are as follows (Shani and Gunawardana, 2011)[2]:

$$precision = \frac{TruePositives}{TruePositives + FalsePositives} \tag{3.2}$$

$$recall = \frac{TruePositives}{TruePositives + FalseNegatives} \tag{3.3}$$

The $GenericRecommenderIRStatsEvaluator()$ class from Mahout was used as it has functionality to split the dataset into separate training and testing sets in order to calculate values for precision and recall (Said and Bellogín, 2014).

[2]Where true positives are the number of generated recommendations that were actually chosen by students in the test set, false positives are the number of generated recommendations that were not in the test set, and false negatives are the number of courses the system decided not to recommend that were actually chosen by students in the test set (Shani and Gunawardana, 2011).

3.5.3 Evaluating Volume of Data

Using the results received during testing, we were able to determine if there was sufficient data to perform these experiments, based on the coverage and accuracy of the system. Recommendations are generated based on the likelihood of students passing their chosen courses. A relatively high accuracy level would indicate that the data was indeed sufficient. Secondly, the effect that calculated academic prediction grades have on the final recommendations made was examined, enabling us to be able to judge the capability of the system in terms of making both accurate and reliable recommendations of courses to choose.

3.6 Conclusion

Course recommender systems have the potential to be useful in easing the course selection process. Thus, it is therefore useful to examine the feasibility of such a recommender system in order to determine whether its implementation is justified. In order to achieve this, a number of research questions were posed. It was necessary to examine the reliability of the system in terms of producing recommendations that depend on both student-related and academic success-related factors. We then questioned whether the available data was sufficient in combination with academic score predictions in order to achieve reliable recommendations. This was considered in combination with correlations between particular courses and academic success as well as general trends observed in courses. The importance of the data used in recommender systems was discussed; as well as the methods that could be used to clean and prepare it and to render it suitable for use in analysis. We created a database in order to store the data, and generated recommendations using Apache Mahout. These were displayed on the web pages of the system, making use of JSP and Servlet software. By using training and testing datasets, we examined whether the system can make recommendations that match expected results from the testing set, hence proving such recommendations to be accurate.

Chapter 4

Results

4.1 Introduction

In Chapter Chapter 3, we presented the methodology with which we proposed to implement our collaborative course recommender system. We first detailed the research questions we hoped to answer with the results obtained from the research. We set out to investigate which courses should be recommended to Computer Science I students (and the level of correctness of such recommendations), while considering which courses were more or less likely to lead to success or failure for students. We then presented a diagram of what we envisioned the overall course recommender system to look like for the end user, and an outline of the ways in which students can interact with the system. The various stages involved in the CRISP data mining methodology were described, and the necessity of data preparation and analysis was justified. We described the creation of the database used to store student data, and went into detail about the classes in Apache Mahout we used to generate recommendations. The use of JSP and Servlets in the creation of the proposed web-based application were discussed. Finally, methods for the evaluation of the results obtained from the recommender system were outlined, including the metrics of coverage, precision and recall. In In Chapter 4, we now present the results of the research. We will present the course enrolment system that has been created, taking a detailed look at how the system functions and what features are included. We also detail the results of evaluation procedures performed on the data and highlight how effective the system has been in producing effective recommendations. Section 4.2 gives an overview of the dataset statistics, including the numbers of students for each course and what proportion passed the year. Section 4.3 to Section 4.5 give the results regarding the relationships between courses and performance,

including which courses students performed particularly badly in and which courses were the strongest determinants of success or failure. Section 4.6 provides a list of typical courses that would be produced by the implemented system. Sections 4.7 to 4.9 give the results of the coverage and accuracy tests performed on the recommender system, which are used to determine whether the volume of data in the dataset was sufficient. Next, Section 4.10 provides a look at the web interface of the application and what its functionalities are. Section 4.11 concludes the chapter.

4.2 Overall Dataset Statistics

Table 4.1 below displays the courses included in the data as well as the percentages of students that passed the year overall while taking each course. For a more detailed look at the descriptive statistics of the dataset, please see Appendix A.

As seen in Table 4.1, both the numbers of students taking each course as well as the percentage of students passing were widely varied across all the possible courses. For the compulsory courses (COMS1000, MATH1034, MATH1036), the number of students enrolled either equalled or came close to equalling the total number of students (having enrolment numbers of 572, 550 and 565 respectively). Enrolment numbers for elective courses ranged from 1 to 447. Pass percentages were generally consistent for the compulsory courses, but varied more for elective courses. Elective courses and their effect on determining performance are discussed in more detail in Section 4.5. In Section 4.3, we looked at the compulsory courses in more detail.

Table 4.1: Overall course results from all years

Course Name	Number of Students	Number of Passes	Pass Percentage (%)
ACCN1000	17	5	29.41
APPM1006	447	223	49.89
ARCL1000	4	2	50.00
BIOL1000	6	3	50.00
CHEM1012	71	32	45.07
COMS1000	572	273	47.73
ECON1000	40	21	52.50
ECON1008	88	39	44.32
ECON1009	61	35	57.38
GEGO1000	8	2	25.00
GEOL1000	2	0	0.00
INFO1000	17	9	52.94
INFO1003	17	10	58.82
MATH1034	550	267	48.02
MATH1036	565	271	47.80
PHYS1000	157	81	51.59
PHYS1001	79	30	37.50
PHYS1026	2	1	50.00
PHYS1027	1	1	100.00
PSYC1001	12	4	33.33
PSYC1002	15	4	26.67
PSYC1009	21	9	42.86
STAT1002	4	4	100.00
STAT1003	11	10	90.91
STAT1005	2	2	100.00
Overall	572	273	47.73

4.3 Relationships Between Courses and Performance

In this section, we provide the results pertaining to the following research question:

Research Sub-Question 1:

- *What are the relationships between courses and performance during the year?*

Table 4.2 below shows the numbers of students enrolled in each compulsory course, as well as the average and standard deviation of the marks in each course and pass rates:

Table 4.2: Compulsory Course Results

Course Name	Students	Mean Mark	Std. Deviation	Year Passes	Year Passrate (%)
COMS1000	572	57.93	18.72	273	47.73
MATH1034	550	49.80	15.98	267	48.02
MATH1036	565	46.36	16.36	271	47.80
Overall	572	52.18	14.91	273	47.73

As shown in Table 4.2 above, COMS1000 has a higher average mark than both MATH1034 and MATH1034, with its mean mark being 57.93 versus 49.80 and 46.36, respectively. The standard deviation across all three subjects is relatively similar, being within the range of 15 - 19. Pass rates across all subjects were similar as well, being under 50%, owing to the vast majority of students taking all 3 courses as they are compulsory.

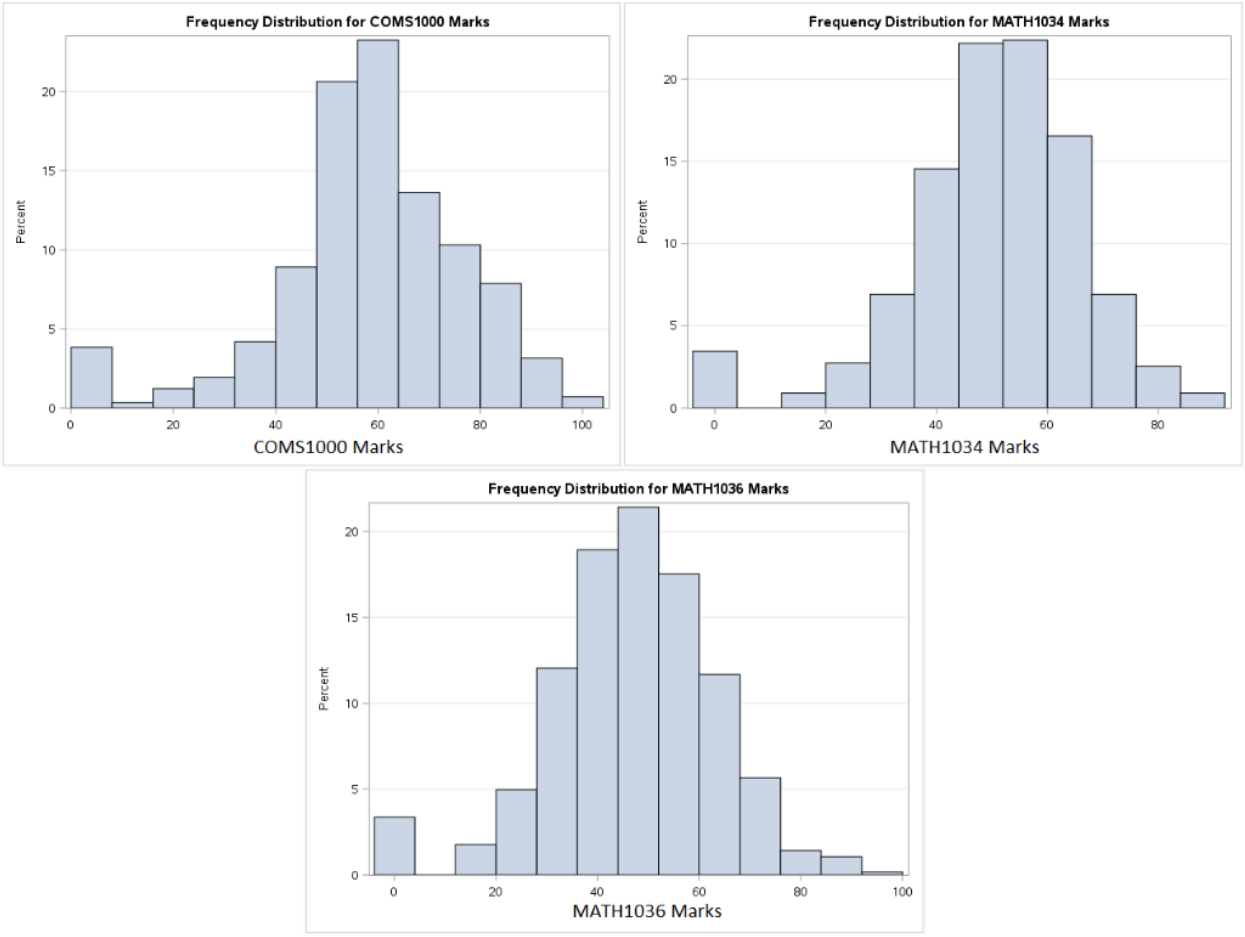


Figure 4.1: Frequency Distribution of Marks of All Students (Author's own work)

Skewness:

- COMS1000: -0.9272
- MATH1034: -0.8440
- MATH1036: -0.4362

Figure 4.1 above shows the frequency distributions of the final marks of the students for the compulsory courses. The majority of the marks fell between 40 and 60 for all three of the subjects. The skewness values for the graphs were -0.9272, -0.8440, and -0.4362. This indicates that the graphs were skewed to the left and so the data was not normally distributed.

We now examine the relationships between these courses in more detail.

4.3.1 Relationship Between Computer Science I and Algebra I

In order to determine the relationship between Computer Science I and Algebra I, we opted for a Spearman correlation approach, due to the data not conforming to a normal distribution (Mukaka, 2012). We plotted the final marks of COMS1000 against MATH1034 in a scatter plot, as shown in Figure 4.2 below:

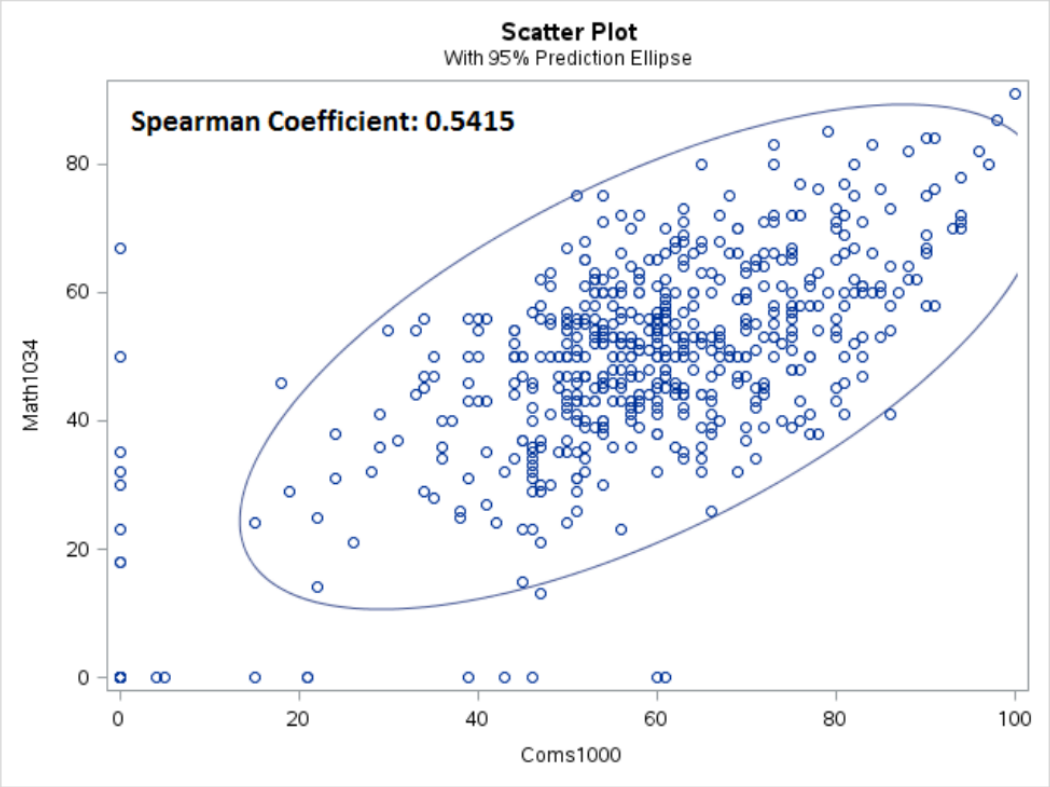


Figure 4.2: Scatter Plot of COMS1000 Marks vs MATH1034 Marks (Author's own work)

Also shown in Figure 4.2 is a prediction ellipse that estimates the area on the graph that any new observations are likely to fall in, at a 95% accuracy. It does this using the mean and standard deviation of the marks in order to obtain the center and boundaries of the ellipse (Friendly *et al.*, 2013).

As can be seen in Figure 4.2, there is a positive correlation between the final marks of COMS1000 and MATH1034. This means that as the final marks get larger for COMS1000, so too do the marks for MATH1034. It is

desirable to calculate the Spearman correlation coefficient in order to understand how strong or weak the correlation between the variables is. In order to do this, we utilised the *proc corr* function with the *spearman* option in SAS in order to automatically generate the correlation coefficient. The calculated value for the correlation coefficient, as can be seen in Figure 4.2, is 0.5415. The possible values for the correlation coefficient fall within the range of -1 to +1 (Mukaka, 2012). Values that are closer to -1 indicate strong negative correlation, and values close to +1 show strong positive correlation between variables. The strength of the correlation changes linearly as the coefficient value moves from -1 to +1. Since the calculated coefficient is close to +1, this indicates a strong positive correlation between the final marks of COMS1000 and MATH1034.

4.3.2 Relationship Between Computer Science I and Calculus I

We plotted the final marks of COMS1000 against MATH1036 in the scatter plot shown in Figure 4.3 below:

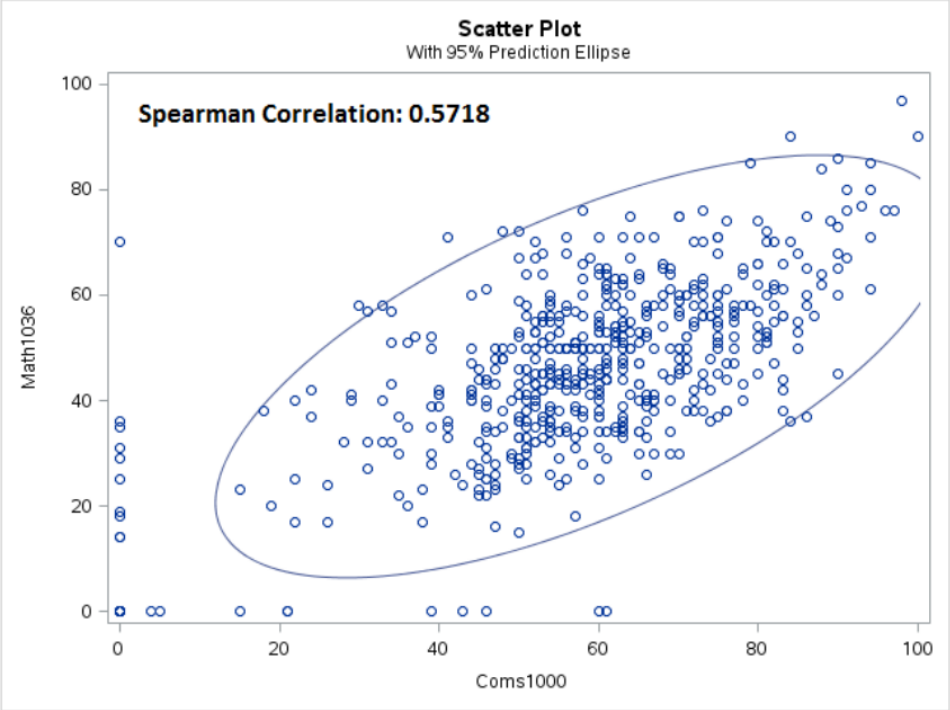


Figure 4.3: Scatter Plot of COMS1000 Marks vs MATH1036 Marks

Figure 4.3: Scatter Plot of COMS1000 Marks vs MATH1036 Marks (Author's own work)

As can be seen in Figure 4.3, there appears to be a positive correlation between the final marks of COMS1000 and MATH1036. In general, as the marks for COMS1000 got higher, the marks for MATH1036 increased as well. To determine the relationship between Computer Science I and Calculus I, we once again used the *proc corr* function in SAS to find the Spearman coefficient. The value of the correlation coefficient was found to be 0.5718. Since this value is close to +1, it indicates a strong positive correlation between the final marks for COMS1000 and MATH1036.

4.3.3 Relationships Between Elective Courses and Performance

To get a sense of the effect of each of the different elective courses on the performances of students, we plotted scatter plots of the final marks of students of each of the elective courses versus the weighted average marks of each student. Prediction ellipses were also included in these plots. The graphs are shown in Figure 4.4 below. Looking at the graphs in Figure 4.4, correlations between final marks of elective courses and the weighted average marks of students seem to be generally positive. As the weighted average marks increase, the final course marks increase proportionally. It can be seen that some of the relationship plots in Figure 4.4 are near-empty or have few data points plotted on them.

This is a result of certain courses having very few instances of students enrolled in them over the six-year period that the data is taken from. For courses that have small numbers of students enrolled over the course of the six years, it is impossible to observe a meaningful trend coming from the graphs. Any correlation observed for these subjects cannot be seen as useful, as the amount of data is too small and hence the correlation cannot be extended to predictions for the marks of a whole population or for that of future students. Thus, no conclusions can be made on the basis of these particular graphs.

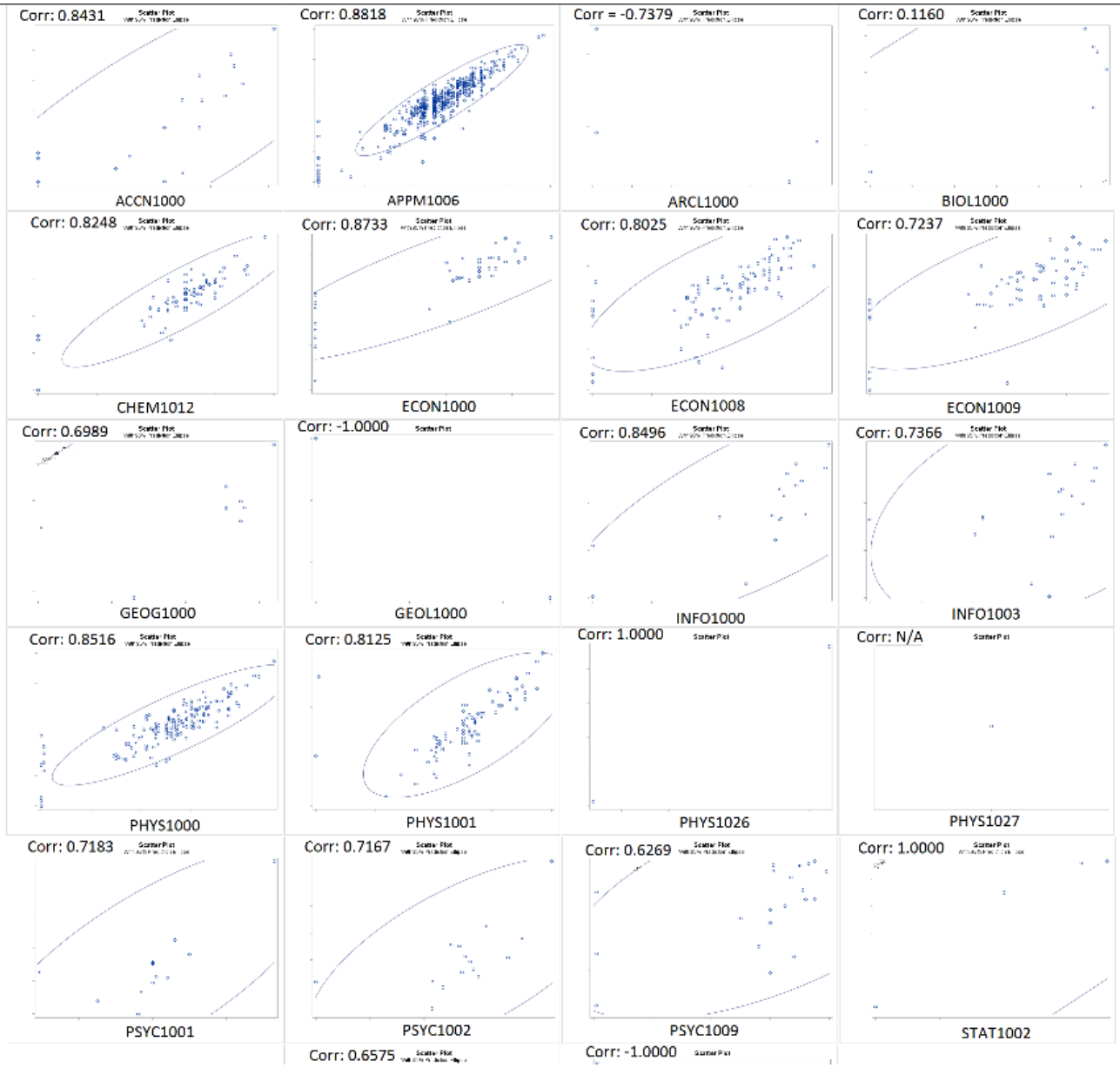


Figure 4.4: Scatter Plot of Elective Course Marks vs Weighted Average Mark (Author's own work)

We now examine the courses with enrolment numbers large enough to make meaningful predictions from them. Using *proc corr* in SAS, correlation coefficients were once again calculated for all the courses in the scatter plots. All of these courses demonstrated a positive correlation coefficient that was close to +1. The strongest correlation for courses with higher enrolment numbers was found to be between Applied Mathematics I (APPM1006) and the students' weighted average marks, with a correlation coefficient value of 0.8818. There are no correlation values that fall below 0.6, indicating strong correlations across all subjects. Hence, there is a general trend that students are more likely to do well on average as the marks of their respective

elective courses increase. There were no instances of scatter plots that did not conform to this trend.

4.4 Courses with Low Grades

In this section, we provide the results pertaining to the following research question:

Research Sub-Question 2:

- *Are there courses in which students perform particularly badly, in comparison to others?*

Figure 4.5 below displays the average final mark for each subject, along with the number of students that enrolled in those courses:

	ACCN1000	APPM1006	ARCL1000	BIOL1000	CHEM1012	ECON1000	ECON1008	ECON1009
No. of Students	17	447	4	6	71	40	88	61
Mean Mark	45.24	53.33	27.25	44.50	48.51	39.68	44.77	44.75
	GEOG1000	GEOL1000	INFO1000	INFO1003	PHYS1000	PHYS1001	PHYS1026	PHYS1027
No. of Students	8	2	17	17	157	79	2	1
Mean Mark	44.88	16.50	57.59	68.82	51.87	51.18	64.50	37.00
	PSYC1001	PSYC1002	PSYC1009	STAT1002	STAT1003	STAT1005		
No. of Students	12	15	21	4	11	2		
Mean Mark	60.08	54.47	56.95	61.00	51.09	52.16		

Figure 4.5: Average Marks Across All Elective Courses (Author's own work)

- **Overall Mean Weighed Average Mark: 52.18**

In Figure 4.5 above, courses with particularly low marks include ARCL1000, ECON1000, GEOL1000 and PHYS1027, all of which have mean marks below 40%. This is much lower than the mean weighted average mark, ranging from GEOL1000's average mark being approximately 35% lower (16.50) to ECON1000's average mark which is approximately 12% lower (39.68). However, we note that ARCL1000, GEOL1000 and PHYS1027 have low numbers of students enrolled. Due to this, we cannot definitively say that the average

number of students passing can be extended to a hypothetical larger population of enrolled students. Therefore, this data does not hold much meaning.

On the other hand, the one course that had both low average marks and a reasonable number of students enrolled was ECON1000. This is the one course for which we can state that students performed particularly badly, owing to the average mark being much less than the mean weighted average. Other courses with relatively low marks include ECON1008 and ECON1009, with mean marks falling within the 40-45% range. However, the magnitude of the differences between these marks and the weighted average mark is not as great as that of ECON1000.

4.5 Strongest Determinant of Performance

In this section, we provide the results pertaining to the following research question:

Research Sub-Question 3:

- *Which first year courses are the strongest determinants of a student's success or failure?*

Taking from Table 4.1 in Section 4.2, we note the following set of courses, that correspond to higher and lower year pass percentages:

Table 4.3: Elective course results from all years

Course Name	Number of Students	Number of Year Passes	Year Pass Percentage (%)
ACCN1000	17	5	29.41
ECON1008	88	39	44.32
ECON1009	61	35	57.38
GEGO1000	8	2	25.00
GEOL1000	2	0	0.00
INFO1003	17	10	58.82
PHYS1001	79	30	37.50
PHYS1027	1	1	100.00
PSYC1001	12	4	33.33
PSYC1002	15	4	26.67
PSYC1009	21	9	42.86
STAT1002	4	4	100.00
STAT1003	11	10	90.91
STAT1005	2	2	100.00

As seen in Table 4.3, courses with a sizable number of students enrolled include ECON1008, ECON1009 and PHYS1001. Students taking ECON1008 and PHYS1001 were shown to have a low rate of passing the year, with a pass percentage of 44.32% for ECON1008 and an even lower pass percentage of 37.50% for PHYS1001. It can therefore be seen that students that enrol for these two courses have a much lower possibility of overall year success.

In contrast, students taking ECON1009 recorded a higher rate of passing the year, with a value of 57.38%. This is the one course that, when taken, significantly increases the chance of success for students. The effect on pass rate when taking ECON1009 is not as great as that when taking PHYS1001, however, it still represents a noteworthy increase in pass rate.

For the sake of completeness, we will also mention courses with lower numbers of students, however these courses cannot be definitively proved to have a large effect on pass rate. Students taking PSYC1002 had an extremely low year pass rate of 26.67%. The highest year pass rate in this range was 90.91%, which was observed for students taking STAT1003. Classes with enrolment numbers below 10 showed year pass percentages of as low as 0% and as high as 100%, however as stated, these do not warrant significant consideration.

4.6 Combinations of Courses to be Recommended

In this section, we provide the results pertaining to the following research question:

Research Sub-Question 4:

- *What combinations of courses should be recommended in the students' first year of study?*

The course recommender system uses a collaborative recommendation approach, in which course combinations selected by past students are considered together with courses taken by the active user of the system in order to provide recommendations for the user (Adomavicius and Tuzhilin, 2005). First-year computer science students are required to take the following courses:

- Computer Science I (COMS1000)
- Algebra I (MATH1034)
- Calculus I (MATH1036)

These courses are thus given to the student by the system by default. In addition, the system also produces a list of elective courses to choose from that can be taken in combination with the compulsory courses. However, students enrolling for the first time do not have any course history. This means that there are no courses or ratings for the system to compare other students' choices to in order to generate recommendations. Thus, as explained in Section 3.4.6, a dummy student is created, and is enrolled in the compulsory courses. The grades given to the dummy student in the compulsory courses are intentionally made to be high, so that students that are similar in terms of having high grades are identified by the system and the courses taken by those students are considered. With this in mind, Table 4.4 below shows a typical set of courses and ratings generated by the course recommender system, with rows arranged in descending order according to rating. As we can see in Table 4.4, a set of four courses have been generated by the system. These are Chemistry I (CHEM1012), Computational and Applied Mathematics I (APPM1006), Economics I (ECON1000), and Physics I (PHYS1001). The courses that were found to have the highest ratings were selected by the system and listed. An additional course was found

Table 4.4: A typical set of recommendations produced by the course recommender system

Compulsory Courses	Elective Courses	Predicted Rating
Computer Science I (COMS1000)	Chemistry I (CHEM1012)	1.71
Algebra I (MATH1034)	Applied Mathematics I (APPM1006)	1.625
Calculus I (MATH1036)	Economics I (ECON1000)	1.56
	Physics I (PHYS1001)	1.5

by the recommender system (PHYS1000), however the predicted rating for this course was too low and so it was not listed by the system.

Note the predicted rating scores in Table 4.4 above. The predicted grades for the student for each subject were normalized to fall between 0-3, with 0 being the lowest rating and 3 being the highest. All students had their marks converted to this rating system, with marks between 0-25% being converted to 0, marks between 25-50% being converted to 1, marks between 50-75% being converted to 2 and marks between 75-100% being converted to 3. This means that in all the recommendations shown in Table 4.4 above, the system predicted a pass for the student using the system. In order to generate these recommendations, the dummy student mentioned earlier was given ratings of 3 for each of the compulsory courses in order to find similar students that passed and performed well. We now detail the results of various evaluation metrics for the implemented recommender system.

4.7 Coverage of Generated Recommendations

In this section, we provide the results pertaining to the following research question:

Research Sub-Question 5:

- *How good is the coverage of the implemented recommender system?*

As stated in Section 3.4.6, we attempted to use both the Pearson correlation as well as the Spearman correlation in order to determine the similarity between different users of the system. After determining the nearest N neighbourhood of the active user, the system then generates a list of recommended

courses for the user. In order to determine the coverage of the system, we made use of the vast amounts of sample data, as described in Section 3.5.1. Each past student was run through the recommender system in order to see if recommendations could be generated for them. The number of students out of the total that successfully had recommendations generated for them was recorded. The results are shown in Table 4.5 below:

Table 4.5: Coverage Values for Different Correlation Methods and Neighbourhood Numbers

		Coverage	
		Pearson Correlation	Spearman Correlation
Number of Users in Neighbourhood	5	$\frac{368}{573} = 64.2\%$	$\frac{437}{573} = 76.3\%$
	10	$\frac{405}{573} = 70.7\%$	$\frac{555}{573} = 96.9\%$
	20	$\frac{421}{573} = 73.5\%$	$\frac{573}{573} = 100\%$
	30	$\frac{421}{573} = 73.5\%$	$\frac{573}{573} = 100\%$

In Table 4.5 it can be seen that coverage values generally increased as the number of users in the neighbourhood increased. The lowest coverage value for the Pearson correlation was 64.2% at a neighbourhood of 5 users, while the highest recorded value for the Pearson correlation was 73.5% at neighbourhood sizes of 20 and 30. This means that 64.2% and 73.5%, respectively, of the users used in testing successfully had recommendations generated for them. For the Spearman correlation, the lowest coverage value was 76.3% at a neighbourhood of 5 users. The highest value was found to be a full 100% coverage at neighbourhood sizes of 20 and 30 neighbours. When comparing the Pearson and Spearman correlations, the Spearman correlation displayed higher coverage values for all user neighbourhood numbers in comparison to the Pearson correlation.

4.8 Accuracy of Generated Recommendations

In this section, we provide the results pertaining to the following research question:

Research Sub-Question 6:

- *How good is the accuracy of the generated recommendations?*

We obtained accuracy readings for both the Pearson correlation and Spearman correlation and split these according to number of users in the neighbourhood. As explained in Section 3.5.2, in order to evaluate accuracy, we find the precision and the recall of the recommendation results (Shani and Gunawardana, 2011). The precision and recall are calculated using the following formulae:

To obtain readings for the precision and recall, we utilised the built in evaluation functionality of Apache Mahout.[1] Measures of accuracy were calculated for a number of different neighbourhood sizes. The results are shown in Table 4.6 below:

Table 4.6: Precision and Recall of Generated Recommendations for Different Correlation Techniques and Neighbourhood Sizes

		Pearson Correlation		Spearman Correlation	
		Precision	Recall	Precision	Recall
Number of Users in Neighbourhood	5	0.80	0.38	0.83	0.64
	10	0.77	0.40	0.67	0.67
	20	0.71	0.37	0.63	0.61
	30	0.67	0.35	0.66	0.66

As seen in Table 4.6, the precision and recall values generally decreased as the size of the neighbourhood increased for the Pearson correlation. The Spearman correlation did not show any particular trend. The highest precision value for the Pearson correlation was 0.80 at a neighbourhood size of 5, with the lowest value being 0.67 at a neighbourhood size of 30, representing a 0.13 decrease in precision over the range of neighbourhood sizes. These high precision values mean that the vast majority of recommendations made by the recommender system were the correct recommendations to make, i.e., consistent with courses that other students chose in the test set (Shani and Gunawardana, 2011). This indicates the possibility of high accuracy of recommendations. Similar trends were observed for the Spearman correlation,

[1]Specifically, the *GenericRecommenderIRStatsEvaluator*() class was used with values of precision and recall being calculated using the above formulae and returned upon calling it (Said and Bellogín, 2014).

where the highest precision value was 0.83 at a neighbourhood size of 5, decreasing to 0.63 at a neighbourhood size of 20 and increasing again to 0.66 at size 30. Thus, precision values were generally higher for the Pearson correlation than the Spearman correlation.

The highest recall value for the Pearson correlation was 0.40 at a neighbourhood size of 10, with the lowest recall value being recorded as 0.35 at a neighbourhood size of 30 - a minor decrease. This means that the recommendations made by the system were a relatively small fraction of the total number of correct recommendations that could have been made (Shani and Gunawardana, 2011). In contrast, the highest recall value for the Spearman correlation was 0.67 at a neighbourhood size of 10, decreasing to 0.61 at a size 20 neighbourhood. Therefore, recall values were generally higher for the Spearman correlation than for the Pearson correlation. The values for the Spearman correlation indicate that the recommendations made were a higher proportion of the pool of correct recommendations of courses that could have been made (Shani and Gunawardana, 2011).

Clearly, both precision and recall are important for evaluating the system. To obtain a better idea of the accuracy from these values, we calculate the F1 metric, which combines both precision and recall and is calculated using the following formula (Vozalis and Margaritis, 2003):

$$F1 = \frac{2 * precision * recall}{precision + recall} \tag{4.1}$$

The F1 values are hence shown in Table 4.7 below. As seen in Table 4.7, F1 values remained relatively consistent across all neighbourhood sizes for the Pearson correlation. The highest F1 value was 0.52 at neighbourhood sizes of 5 and 10, decreasing slightly to 0.46 at neighbourhood size 30. For the Spearman correlation, the highest F1 value was 0.72 at a neighbourhood size of 5, with the lowest value being 0.62 at a neighbourhood size of 20. Thus, F1 values for the Spearman correlation were higher than those for the Pearson correlation. This means that the accuracy of recommendations was generally better when using Spearman correlation as opposed to using Pearson correlation.

Owing to a high F1 value and suitable coverage, the Spearman correlation and a neighbourhood of size 30 were chosen for use in the recommender system. This ensures that 100% of students will have recommendations generated for them, while the majority of course recommendations match up with those chosen by students in the past.

Table 4.7: F1 Values of Generated Recommendations for Different Correlation Techniques and Neighbourhood Sizes

		Pearson Correlation	Spearman Correlation
		F1 Value	F1 Value
Number of Users in Neighbourhood	5	0.52	0.72
	10	0.52	0.68
	20	0.49	0.62
	30	0.46	0.66

4.9 Volume of Data and its Sufficiency

In this section, we provide the results pertaining to the following research question:

Research Sub-Question 7:

- *How sufficient was the amount of available data in terms of allowing good quality recommendations to be produced?*

The attributes of the dataset that were used were as follows:

- Total number of (unique and non-unique) students: 572
- Total number of courses: 25
- Total number of enrolments made: 2590
- Minimum number of enrolments made in a single course: 1 (PHYS1027)
- Maximum number of enrolments made in a single course: 572 (COMS1000)

There were 2590 observations on which to train the recommender system, spread across a total of 572 students.

From Section 4.7 and Section 4.8 respectively, the best coverage that could be achieved from this data was 100%, and the best F1 value that could be achieved was 0.72. The coverage value obtained shows the potential of the trained system in terms of being able to generate recommendations for

any student. In addition, the high F1 value indicates a high accuracy of recommendation while making use of the dataset in order to form such recommendations. Considering these factors, it can be said that the volume of data that was available for use in this project was generally sufficient, however, greater volumes of data would obviously increase the accuracy even more.

4.10 Web-Based Course Enrolment and Recommender System

In this section, we provide the results pertaining to the main research question:

Main Research Question:

- *Is a collaborative recommender approach, developed in a web-based environment using Apache Mahout, suitable in order to suggest relevant courses to students, while striking a balance between the students' own interests and crucial field-related material in order to ensure academic success?*

A detailed examination of the web-based application that has been created is carried out in this section. We provide screenshots of the different parts of the application that are encountered on a typical run-through of the system by the user. By going through each section of the application, we can detail the background processes that are taking place and hence give an explanation of how the program works as a whole.

4.10.1 Web Browser Specifications

The system was developed with a range of web browsers in mind. The layouts of each individual page within the course recommender application were designed so that the pages would look presentable no matter which web browser is used to access the system. In addition, the elements on the web pages respond to the resolution of the monitor being used, ensuring that each part of the application can be viewed regardless of screen size.

The recommended browsers to use when accessing the course enrolment system consist of the following, as displayed in Table 4.8 below:

Table 4.8: Latest Versions of Recommended Browsers for the Course Enrolment System

Mozilla Firefox 49.0.1	Google Chrome	Microsoft Edge 25.1	Opera 34.0	Safari (OS X 9.0.2)

4.10.2 Application Design and Use

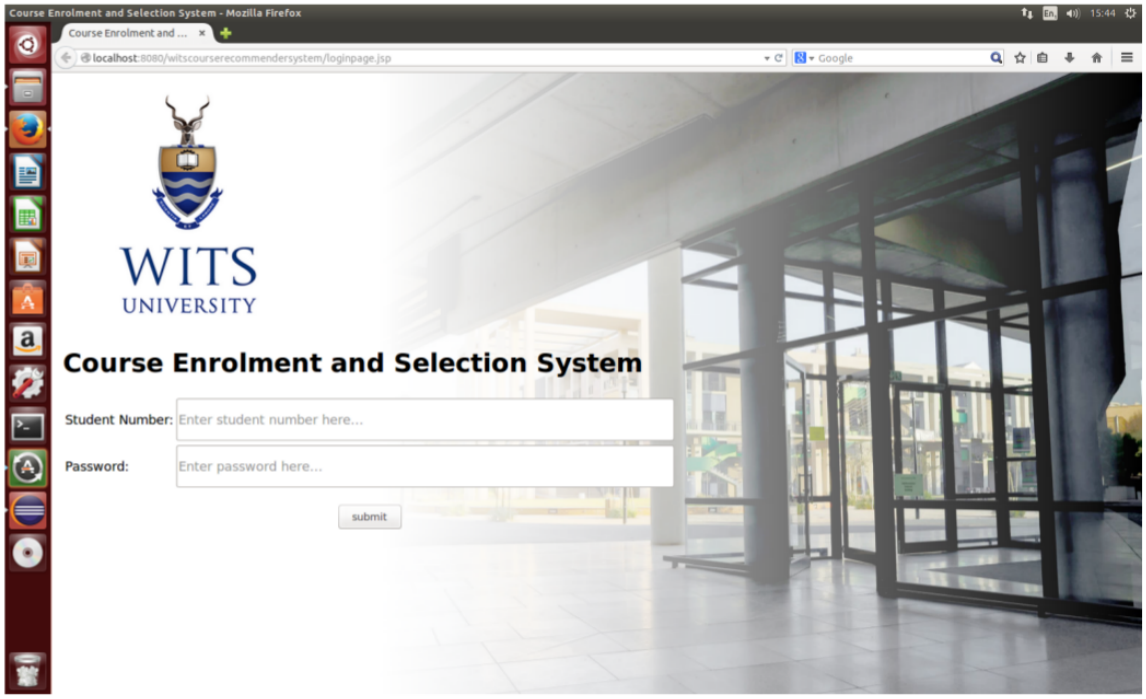


Figure 4.6: The log in screen of the web application (Author's own work)

Figure 4.6 above shows the initial screen of the application, the log in screen. At this page, the student is required to enter their unique student number and password in order to access the system. The student number and password would have been given to the student after they had been accepted into the University.

The backend SQLite database contains a table for all students that stores the values for both their student numbers and passwords. Once the student has entered information into the page, the system checks it against the stored values for the student. If both the student number and password match, then

the student is logged into the system. Otherwise, the student is presented with an error message and is asked to re-enter their credentials.

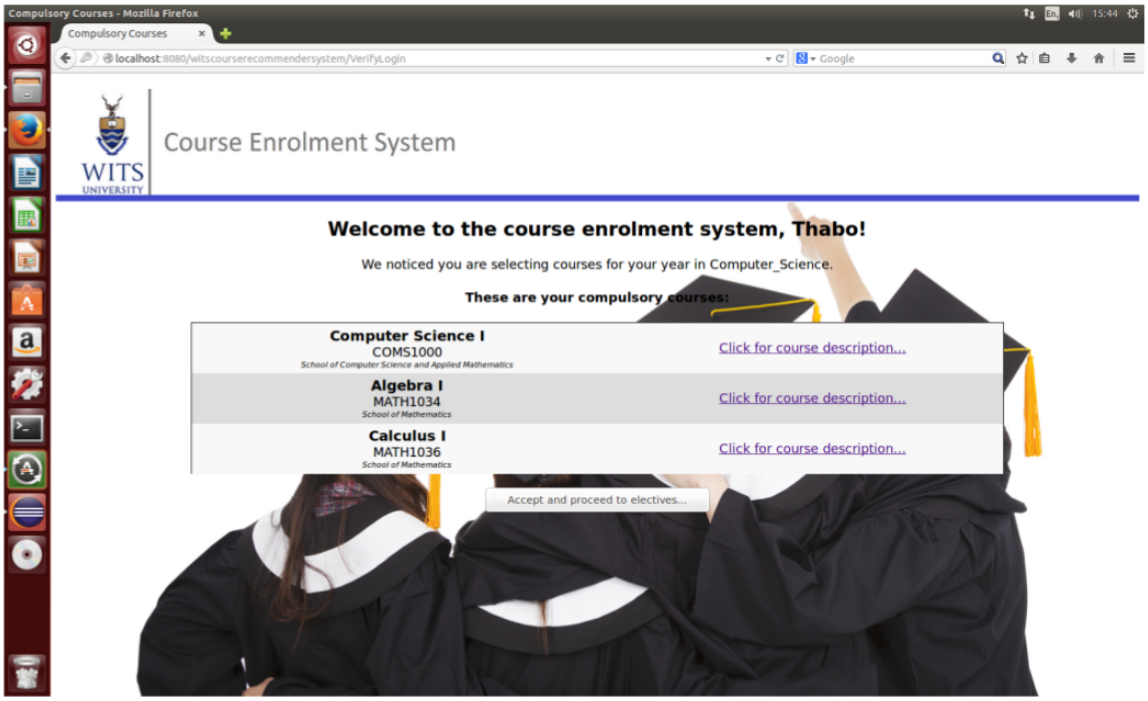


Figure 4.7: Compulsory course page (Author's own work)

Figure 4.7 shows the page displaying the compulsory courses that the student is taken to after logging into the system. The database has a record of the major the student applied for. Associated with the major, the student must take a certain number of compulsory courses. For Computer Science I students in particular, these compulsory courses are COMS1000 (Computer Science I), MATH1034 (Algebra I) and MATH1036 (Calculus I).

The system retrieves the compulsory courses relevant to the student's major and their descriptions from the database and displays them in a dynamic table on the page. In order to avoid clutter, the descriptions of the courses are collapsed and the user must click on (Click for course description...) to see the full description. When the user is satisfied they have read up about their courses, they can proceed to the next page.

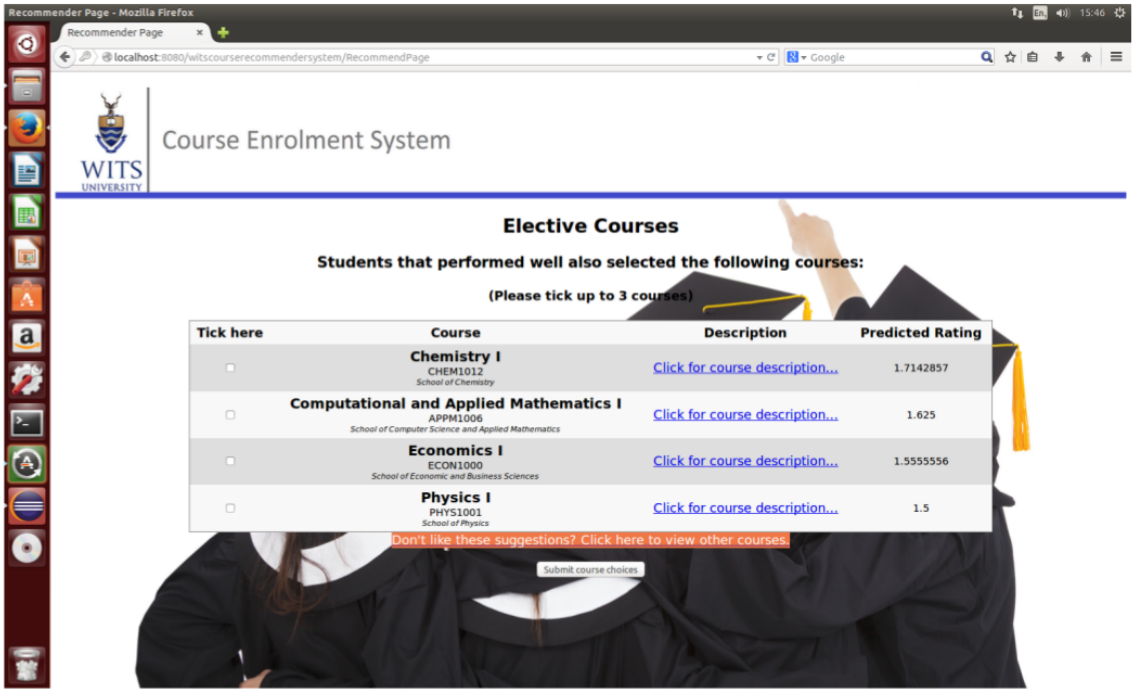


Figure 4.8: Recommended courses page (Author's own work)

Figure 4.8 displays the page showing the courses that the system has recommended for the student. The backend of the system checks the compulsory courses that the student must take and then calculates recommendations for the student using Apache Mahout functions. The predicted rating for each subject is calculated, which have been normalised to fall within a range of 0 to 3. A rating closer to 4 indicates that the student is more likely to perform favourably in the course. Once again, a dynamic table is created with the name of the courses, their descriptions as well as the rating on the right. The rows in the table are organised in descending order according to which courses have the highest ratings. The student may tick which courses they would like to take using the checkboxes to the left of the page. However, if the courses suggested do not pique the interests of the student, they may click a link to be taken to another page displaying all courses.

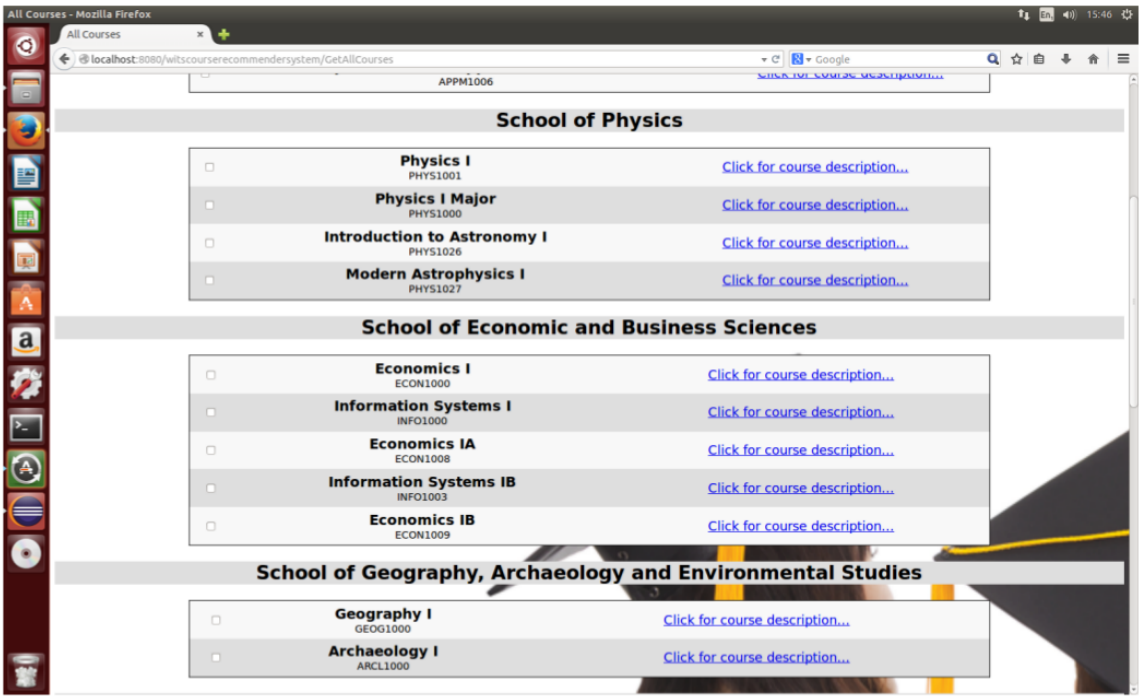


Figure 4.9: Page displaying all possible courses (Author's own work)

Figure 4.9 shows the page with all possible courses displayed. Once again, this is a dynamic table, with courses organised according to the respective school that each course is offered in. Once again, the student may tick which courses they wish to take using the checkboxes on the left of the page. The student may read a description of each course by clicking the (Click for course description...) button within the description field. As stated before, all possible courses are open to the student to select. Even if the system did not generate a favourable result prediction for the student for a particular course, it is still open to the student themself to make the final decision. Once the student has selected the courses they wish to take, they may proceed by clicking the (Submit course choices) button.

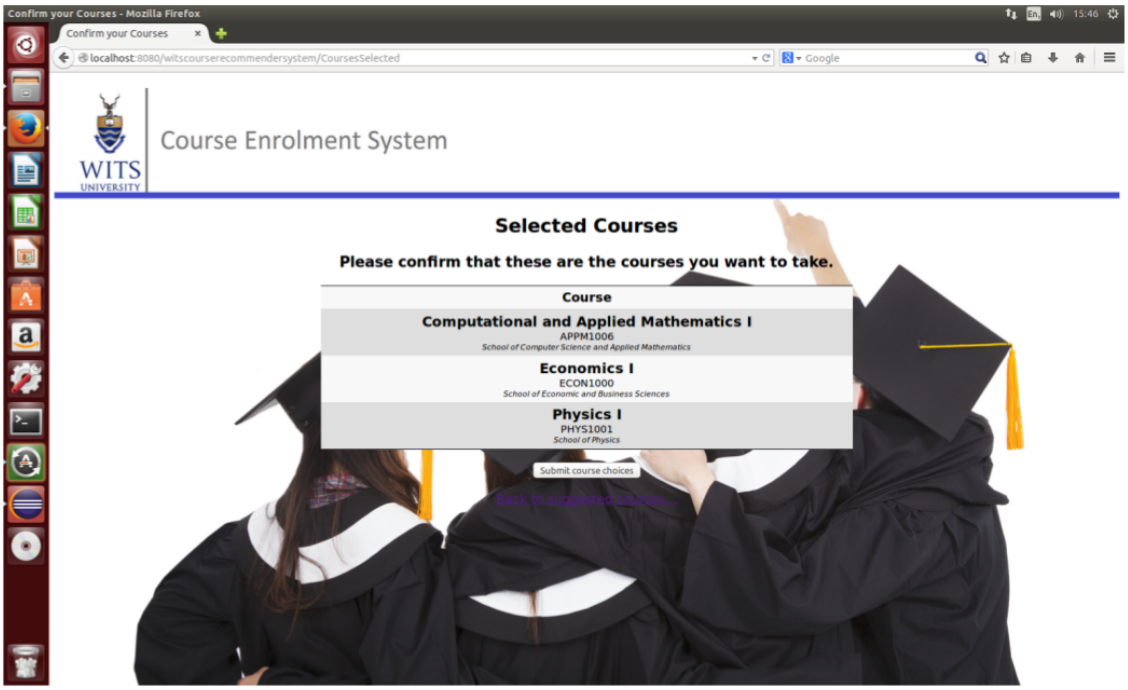


Figure 4.10: Confirmation of courses (Author's own work)

Figure 4.10 is a screenshot showing the page asking the student to confirm that the courses they selected are definitely the courses they want. Each course that the student selected previously is displayed in a list here. If the student is satisfied with their courses, they may proceed by clicking the (Submit course choices) button. Otherwise, if the student wishes to make an amendment to the courses they selected, they can click the (Back to suggested courses) link in order to go back to the recommended courses page. Once the student has decided to proceed, the SQLite database is updated and the choices that the student has made are inserted. A flag within the student's record in the (students) table is activated, signalling that the student has completed their enrolment and may not do so again. Every successive login attempt will notify the student that they have already registered.

The web application that was created successfully integrates the different components of the system. It displays the accurate and all-covering recommendations generated by Apache Mahout. However, it also allows students to choose their own interests over courses recommended for them by the system. The presence of recommendations in the system increases the chances of students making choices that will lead them to success in the year ahead.

The web-based interface is simple to access and use. All of these factors combined together make a reliable and intuitive enrolment system.

4.11 Conclusion

Throughout this chapter, the results pertaining to the various research questions we posed in Chapter 3 were detailed. We first examined the relationships between compulsory courses, and via the use of the Spearman correlation coefficient, found there to be a strong positive correlation between the marks of Computer Science I (COMS1000) and Algebra I (MATH1034). In addition, there was a strong positive correlation between the final marks of Computer Science I and Calculus I (MATH1034). Next, examinations of the relationships between elective courses and performance found that as marks in elective courses increased, the final average marks of students generally increased as well. The strongest correlation was found to be between the APPM1006 marks and the weighted average marks.

We then considered the elective courses in order to determine if there were any courses with particularly low marks. Results indicated that ARCL1000, ECON1000, GEOL1000 and PHYS1027 all had average marks below 40%. Courses that strongly determined performance were also considered. For courses with significant enrolment numbers, ECON1008 and PHYS1001 were found to contribute to lowering overall year performance, while students taking ECON1009 passed more often overall.

We then detailed a typical set of recommendations that would be produced by the recommender system. Compulsory courses were recommended by default, while a set of elective courses were recommended based on predicted performance within those courses. Predictions were generated using high dummy marks in compulsory courses, with the final grade predictions normalised into a rating. The coverage of our implemented recommender system was examined and found to rise as high as 100% using a Spearman correlation and a large neighbourhood. Accuracy was also considered, and combining the precision/recall metric, F1 values were found to rise as high as 0.72 using the Spearman correlation with small neighbourhoods. The volume of data was found to be sufficient to create a reliable system based on the values of accuracy and coverage.

Finally, we provided a detailed overview of the design and workings of the web-based application. The login page was detailed, and the page displaying

the set of compulsory courses the student needs to take was also described. We showed the elective courses page, on which the student is presented with the set of courses generated by the recommender system and asked to select the courses they would like to take. Students unhappy with the courses recommended for them can instead go to a page with all possible courses listed. Lastly, the student will be expected to submit their final set of courses, which are recorded and stored in the backend SQLite database. The final web-based course enrolment system successfully integrates recommendations while also allowing students to follow their interests instead.

Chapter 5

Discussion

5.1 Introduction

In Chapter 4, we presented the results obtained during the course of our research. Findings involving correlations between courses and performance were given. Details about the implemented course recommender system, such as what combinations of courses would be recommended, as well as the coverage and accuracy of the recommender were disclosed. A brief run-through of the system, detailing its design and functionality was also included.

In this chapter, we discuss the findings that were presented in Chapter 4. Section 5.2 details the significance of the results, including the reasons the results came about. Next, Section 5.3 describes what the results mean for the University of the Witwatersand and why it would be beneficial for the University to implement the system. Section 5.4 briefly describes the limitations of the research. Finally, Section 5.5 concludes the chapter.

5.2 The Significance of the Results

In this section, the results relating to each research question are examined in more detail. We also discuss what factors could have lead to the results that were obtained and give reasons as to why certain results turned out or did not turn out as expected.

5.2.1 Courses and Performance

From the results concerning the relationship between COMS1000 and MATH1034, it was found that there was a strong positive correlation between the final

marks of the students taking COMS1000 and MATH1034. A similar finding was made regarding the relationship between the final marks in COMS1000 and MATH1036. The presence of high correlation coefficient values indicates that students that performed well in COMS1000 were very likely to perform well in MATH1034 and MATH1036 as well. This result is not particularly surprising, and there are a number of ways in which such a finding can be explained. Firstly, computer science and mathematics subjects are similar on the basis that they both require the development of sound logical thinking in order to be understood fully. The same logical thinking can be carried between the two subjects, so students that excel in this area are likely to do well in both types of subject. In addition to this, a strong work ethic is also applicable to these subjects and a hard working student is likely to perform well across the board. The strong correlations observed between elective courses and overall performance can be explained in a similar manner, where overall performance is a direct result of consistently studying and completing assignments.

It is difficult to say what the reason is for low performance when taking ECON1000. As we saw in Section 4.4, the average mark in ECON1000 was only 39.68%. It is possible that factors such as workload may have contributed to this. A combination of the pressure induced from having to complete work for the compulsory subjects as well as having to learn a largely different subject in Economics may have been too much for some students. It may be useful to look at what the overall average mark for ECON1000 was (including non-Computer Science students) and see if there is a significant difference from the average mark only including Computer Science students. Having a larger population of students (greater than 40) may have increased overall marks slightly, however low marks are still likely, judging from the results in similar subjects such as ECON1008 and ECON1009 (both with an average grade under 45%).

From these results, it can thus be identified that ECON1000 is one particular subject that should perhaps be avoided when students select their courses, unless they have a particular interest in that subject area. This further solidifies the need for a good course recommender system in University situations such as these - if it is identified that a student is not likely to perform well in a subject, then they should be notified of this beforehand and thus guided in making a better decision.

The use of a recommender system is further justified by the trends observed between courses and performance. Since it was observed that students' over-

all weighted average mark is directly correlated to how well they perform in individual subjects, it is in their best interest to select subjects that they are most likely going to obtain high grades in. Therefore, basing the recommendations of a recommender system on the likely performance in a subject is a suitable method in order to produce good recommendations.

5.2.2 Reliability and Accuracy of Recommendations

In Section 4.6 we saw a typical set of courses generated by the system. The courses displayed were those found to have the highest predicted grades for the student. This is consistent with what we have discussed in terms of increasing students' chances to succeed. However, as we outlined in Section 4.6, these courses could not be personalised for a particular student due to new students never having taken courses at the University before. The implemented procedure of using a dummy student with high marks in order to generate recommendations via similarities with other high-scoring students is an effective strategy for helping new students. It is able to assess which combinations of courses led to high scores across the board and thus recommend that the new student follows the same path, leading to a greater chance of good results.

Of course, in order to achieve good recommendations, the implemented recommender system is required to be both accurate, while also able to generate recommendations for all students using the system. As stated in Section 4.7, it was found that coverage of the system was generally good when using either the Pearson correlation or Spearman correlation; however, the Spearman correlation performed better with every different neighbourhood size - reaching coverage levels of 100% for higher neighbourhood sizes. The likely reason for this is that the Spearman correlation does not rely on the variables - in this case final marks - conforming to a normal distribution (Mukaka, 2012). This is seen in Figure 4.1, where the frequency distributions for the marks are slightly skewed. In addition, the scatter plots (Figures 4.2, 4.3 and 4.4) show a relationship that is not completely linear - one that may be better modelled by a Spearman correlation (Mukaka, 2012). As such, when the recommender system is attempting to find relationships between the active user and past users, it is likely to pick up certain courses as being correlated where a Pearson approach would not. Thus, a larger pool of selected courses allows for better coverage, and our choice of using the Spearman correlation is correct and justified.

In terms of the accuracy of the implemented recommender system, we ob-

served F1 values as high as 0.72 when using the Spearman correlation, as reported in Section 4.8. The reasoning for such high accuracy is similar to that for coverage - since the Spearman correlation more closely models the relationship between users and course combinations, users that are deemed to be similar have their course choices recommended to the active user, and these recommendations are thus more likely to be correct. The use of the Spearman correlation in terms of accuracy is once again justified by this.

One more important point to mention is considerations around the trade-off between accuracy and coverage. Different neighbourhood sizes result in higher or lower accuracy and coverage, and these two are inversely related, as seen in Sections 4.7 and 4.8. For example, a neighbourhood of size 5 resulted in a respectable F1 value of 0.72; however, the coverage of the recommender with this neighbourhood was diminished to only 76.3%. It is thus crucial that we selected a neighbourhood size that would balance the coverage and accuracy in order to create the best possible recommender system. It is for this reason that we chose a neighbourhood size of 10 - coverage for this neighbourhood size was shown to be 96.9%, meaning that the vast majority of students will get recommendations generated for them. Additionally, the F1 value only slightly decreased to 0.68, meaning that recommendations generated were still accurate, and the system in the majority of cases is able to find trends in the combinations of courses that students took. These high accuracy and coverage values are very promising, and thus show that the volume of data used to train the system was good enough to allow for a robust and reliable recommender system, as stated in Section 4.9. Had there been more data to train on, accuracies and coverage would be expected to increase even more.

5.2.3 The Web-Based Recommender Application

In Section 4.10, a detailed look into the workings of the course enrolment system was carried out. We examined each of the pages in the system, from the login page to the elective course selection pages, with recommendations generated and displayed by the backend recommender system. The overall layout of the web pages is expected to be simple for new users to understand, including those that may not have had significant amount of experience with using computers or web browsers before. Courses and their descriptions are listed in tables to give users a full understanding of what is available to them. The system is fast and responsive, and the generation of recommendations only takes a matter of seconds, thus the user should be able to complete their enrolment quickly and efficiently. The colour schemes and background

images are chosen so that no text is obscured or made hard to read due to contrasting colours. Instructions are provided on every page as to the intention of every page and what actions the user must perform to proceed. We therefore expect that were such a system to be implemented in the University, there would be relatively few issues encountered by users of the system.

The Apache Mahout backbone provides the necessary correlation and filtering algorithms in order to generate accurate course recommendations, as we have discussed in Section 3.4.6. These are presented to students in the hopes that the students will ensure their own success by selecting the best courses. However, students' own interests are considered as well in the design of the system, as they may look at other course options if they do not like what is recommended to them. All of this is organised in an easy-to-use web-based system.

We can therefore conclude that our collaborative recommender approach, developed in a web-based environment using Apache Mahout, is suitable in order to suggest relevant courses to students, while striking a balance between the students' own interests and crucial field-related material in order to ensure academic success.

5.3 Implications of the Results

It is important now to consider what the University of the Witwatersrand can do with this research, and what benefits the research may bring, both to students and to the University. For new students coming into the University, the implementation of this web-based course enrolment system would be a significant help in guiding them on the best course choices and making informed decisions. Students that take the advice of the system and select courses recommended for them can expect to perform better during the year than they would have if they had selected courses without any advice. This is particularly noteworthy for students who are on financial aid, as passing the year would put them on track to receive financial aid the next year and thus they would not need to be concerned about dropping out over a lack of funding.

A reduced rate of drop outs is also desirable for the University. An increased pass rate is helpful towards increasing the standing and ranking of the University amongst other universities in South Africa as well as worldwide. The Computer Science degree offered by the University will be seen

as more competitive and there will be a greater demand for it in the coming years. In addition, the University will be less at risk of losing fees from students that are not on financial aid, as fewer students would be forced to leave the University.

Finally, the costs to the University of implementing such a system would be relatively low. The database and web pages in the system would just need to be hosted on a server by the University so that students can access them via the Internet. The existing enrolment system is already Internet-based and accessible via the Student Information Management System, so implementing this system in its place should be relatively simple.[1] The benefits of having such a system in place would certainly outweigh any costs in the long term.

5.4 Limitations of the Research

There were a few limitations with regards to this project. The first limitation is the time constraints of the project. The project had to be completed within the year, meaning that some methods used may not have been optimal, Using metrics of accuracy and coverage to test the system may be feasible, but they do not provide real evidence of how good the recommendations of the system actually are. It would be desirable to test the system on real students to see how their marks are affected by the recommendations made.

Another limitation was the amount of data used in the project. There were many cases within the data in which certain subjects had very few enrolment numbers, meaning data related to those subjects was not particularly meaningful. Although the accuracy of the recommender system was found to be rather high, it would be desirable to obtain an even higher accuracy to optimise the recommendations generated. This could be done with a larger pool of data to work with.

5.5 Conclusion

In this chapter, we discussed the results obtained in our research. We attributed the positive correlations between subjects and performance to shar-

[1]University of the Witwatersrand Self-Service-Student Information Management System, 2016. Retrieved 17 October 2016, from https://self-service.wits.ac.za/psp/csprod/UW-SELF-SERVICE/.

ing similar logical thinking requirements, as well as surmising that if a student is a hard worker in one course, then they are likely to be hard workers in other courses as well. The fact that students did particularly badly ECON1000 was explained to be possibly as a result of work load. Courses such as these in which students perform badly provide further evidence as to why recommender systems are valuable - students would be steered away from taking these courses.

The results pertaining to the recommender system itself were also discussed. We stated that our method of using dummy scores to generate recommendations was feasible, due to it retrieving courses that have had high marks historically. We justified our use of the Spearman correlation method in generating recommendations on the basis that the student data was not normally distributed, and relationships between students and courses were more monotonic. This lead to high recommendation accuracy and high coverage, with an F1 value of 0.66 and a coverage of 100% for a neighbourhood size of 30, which is the size we decided to use. These accuracy and coverage values were promising and sufficient for a course recommender system, and this further indicated sufficient data was available for suitable recommendation. We also discussed the web application we developed, citing its ease of use, and stated that the integration of the aforementioned recommender system allowed for students to be guided in making appropriate enrolment decisions while also allowing them the freedom of pursuing their own interests.

Next, we discussed the implications of the results and the limitations of the research done. It is in the best interests of the students to implement the system to increase pass rates and reduce the chance of failure and dropping out. This is important for the University too, to improve its image and to contribute to its financial stability. It would have been preferrable to test the system on new students over the course of a year, but due to time constraints this was not possible. In addition, more data for the system to work with would have contributed towards better recommendations. However, accuracy and coverage of the current system is still relatively high.

Chapter 6

Conclusion and future works

> If we knew what it was we were doing, it would not be called research, would it?
>
> *Albert Einstein*

Enrolling at a university has become a widely popular option for an increasing number of high school students who qualify to enter university. Upon registering at university, the new students are presented with a wide variety of fields of study to choose from. Incorporated into these fields of study are a large selection of courses that students may opt to sign up for. While students may choose to take courses only within their main field in order to maximise their knowledge, they may also wish to enrol for optional courses they may be interested in. Whichever decision they take, the sheer number of courses and information about those courses is likely to be overwhelming and may result in making suboptimal choices about which courses to take. Incorrect decisions also reflect badly on a university if too many students fail as a result of bad decision making. Hence, there is a need to streamline the course selection process that takes place during registration.

To address this problem and to ensure that students select courses that are most likely to result in academic success, this research looks at recommender systems as a possible solution. These systems accept a large amount of preprocessed data and use it to make recommendations or predictions to users based on their own past choices (Adomavicius and Tuzhilin, 2005). The systems can also look at relationships between separate users and try to find links between their respective preferences. The former is a characteristic of content-based recommender systems, which are used mostly on text-based data, while the latter is more typical of collaborative systems, which are

more applicable to our research problem (Adomavicius and Tuzhilin, 2005). To this end, we proposed the design, development and implementation of a web-based application incorporating a collaborative based system in order to provide course recommendations for new students. It incorporates the interests of students in decisions, as well as predicting their likely academic performance, and uses this information to generate the best possible set of recommendations.

Having spoken about our assumptions, the finer details related to developing the system are given. After removing insignificant variables, dealing with missing values and splitting the data using SAS, the Java application code was developed in the Eclipse IDE and the underlying algorithms were investigated. Apache Mahout was selected as the framework of choice, which provides collaborative filtering algorithms that were applied on our dataset. Current students, course information and our historical dataset were all stored in a SQLite database linked to the application. The web-based aspects of the program, such as the layout of pages and integration of the recommender system code were developed using servlet and JSP technology. The system was run off an Apache Tomcat server for the purposes of testing the web application.

We then moved into the more analytical aspects of our research. We used SAS to generate scatter plots and histograms in order to allow us to understand the relationships between the course marks, and to find out which courses students performed badly in and which had a strong influence on chances of passing. To evaluate our implemented recommender system, we opted to use the metrics of precision and recall in order to determine the accuracy of recommendations, and we also checked the coverage of the system. This would enable us to determine whether the volume of data we worked with was in fact sufficient.

We detailed the results of our research in Chapter 4. We found there to be a strong correlation between the marks of the compulsory subjects (COMS1000, MATH1034 and MATH1036). In addition, there were also strong correlations between the marks in elective courses and the weighted average marks of students. We found that students performed particularly badly in ECON1000 - a course with a sufficient number of enrolled students to draw conclusions from. ECON1008 and PHYS1001 also contributed to students being less likely to pass the year, whereas students taking ECON1009 were more likely to succeed.

We then examined a typical set of courses recommended by the system, which showed that the system recommended courses based on which subjects students were likely to perform well in. Recommendations were accompanied by a rating corresponding to how well students were expected to do in the courses. When evaluating the coverage of the system, it was found that coverage rates went as high as 100% using a Spearman correlation and a large neighbourhood of users, which is an outstandingly good coverage rate and is very promising. We combined the metrics of precision and recall into F1 values to better understand the accuracy of the system, and found that the system was able to reach F1 values of up to 0.72 with small neighbourhoods - a good accuracy rating. We opted to use a neighbourhood of size 30, trading between accuracy and coverage and ending up with respectable values of 100% coverage and a 0.66 F1 value. These values helped us to determine that the dataset used was sufficient in size.

The design of the web-based application was also detailed. Students can move from a login screen to a page showing their compulsory courses. This then leads to another page displaying the list of recommended elective courses for the student. Students unhappy with the recommendations given to them can instead view the full list of courses and choose from there instead. Confirming their selections causes the system to store them in the backend database.

The significance and implications of these results were discussed in Chapter 5. Correlations between individual course marks and overall year marks likely come about due to strong logical thinking as well as good work ethics. The identification of subjects with particularly bad marks reinforces the need for recommender systems to guide students away from making bad choices. The system's method of making recommendations is suitable, as it resulted in the system recommending courses that students with high marks in compulsory courses took. The fact that accuracy and coverage values are higher when using the Spearman correlation was expected and made sense, due to the distribution of the data not being normal and correlations being monotonic. The web application this recommendation functionality was integrated into is easy to use for newcomers, and is suitable both for students that choose to follow the recommendations as well as students wanting to pursue their own interests. The use of this system by the University of the Witwatersrand would both aid the students in ensuring a good chance of performing well, as well as aid the University in improving its standing worldwide and make it less likely to lose out on student fees. We thus conclude that a collaborative recommender approach, developed in a web-based environment using Apache Mahout, is suitable in order to suggest relevant courses to students, while

striking a balance between the students own interests and crucial field-related material in order to ensure academic success.

There is potential for future work to be done in order to improve the system. One major way in which the system could be improved is to extend its functionality to provide recommendations to new students outside of the School of Computer Science. The design of the system is such that extending functionality to other disciplines is simple, provided that there is historical data with which to train the system. New fields simply have to be inserted into the database, and combined with the historical data, the system should be able to provide recommendations without too many issues.

Future work could also combat the limitations of the project. A better method of testing the effects of recommendations on pass rates and performance would be to implement the system for a year at the University and observe the results of the students at the end of the year. These results would then be compared with those from previous years in order to find out whether they were significantly higher than the previous results. This would provide more concrete proof as to how helpful the system would be. In addition, training the system on a larger database of past students, perhaps those from before 2010, would serve to improve the system as there may be more data about subjects with lower enrolment numbers. General trends about these subjects would be able to be observed, such as whether students are likely to perform well in the subjects. In addition, there may be data about subjects that have not been covered at all in the pool of data used, allowing the system to diversify its recommendations even further.

The research that was conducted during the course of this project showed the potential of recommender systems in educational settings. It is evident that collaborative recommendation, such as the recommender integrated into the web-based course enrolment system we developed, could have a notable effect on the performance of students at the University of the Witwatersrand. Our system gives students the guidance needed to make informed decisions about course choices and balance their interests with courses they are likely to perform well in. This puts students on the path to having a fulfilling and successful experience during their tertiary studies.

References

[Ackoff 1989] Russell L Ackoff. From data to wisdom. *Journal of applied systems analysis*, 16(1):3–9, 1989.

[Adomavicius and Tuzhilin 2005] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, 17(6):734–749, 2005.

[Aher and Lobo 2012] Sunita B Aher and LMRJ Lobo. Prediction of course selection by student using combination of data mining algorithms in e-learning. *International Journal of Computer Applications*, 40(15):1–7, 2012.

[Dai *et al.* 2007] Naci Dai, Lawrence Mandel, and Arthur Ryman. *Eclipse Web tools platform: developing Java Web applications*. Pearson Education, 2007.

[Delwiche and Slaughter 2012] Lora D Delwiche and Susan J Slaughter. *The little SAS book: a primer*. SAS Institute, 2012.

[Ekstrand *et al.* 2011] Michael D Ekstrand, John T Riedl, Joseph A Konstan, et al. Collaborative filtering recommender systems. *Foundations and Trends® in Human–Computer Interaction*, 4(2):81–173, 2011.

[Friendly *et al.* 2013] Michael Friendly, Georges Monette, John Fox, et al. Elliptical insights: understanding statistical methods through elliptical geometry. *Statistical Science*, 28(1):1–39, 2013.

[Gomez-Uribe and Hunt 2016] Carlos A Gomez-Uribe and Neil Hunt. The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)*, 6(4):13, 2016.

[Ingersoll 2009] Grant Ingersoll. Introducing apache mahout. *IBM developerWorks Technical Library*, 2009.

[Lee and Cho 2011] Youngseok Lee and Jungwon Cho. An intelligent course recommendation system. *SmartCR*, 1(1):69–84, 2011.

[Lee and Hosanagar 2016] Dokyun Lee and Kartik Hosanagar. When do recommender systems work the best?: The moderating effects of product attributes and consumer reviews on recommender performance. In *Proceedings of the 25th International Conference on World Wide Web*, pages 85–97. International World Wide Web Conferences Steering Committee, 2016.

[Linden *et al.* 2003] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003.

[Mehboob *et al.* 2012] F Mehboob, S Shah, and NA Bhutto. Factors influencing student's enrollment decisions in selection of higher education institutions (hei's). *Interdisciplinary Journal of Contemporary Research in Business*, 4(5):558–568, 2012.

[Mukaka 2012] Mavuto M Mukaka. A guide to appropriate use of correlation coefficient in medical research. *Malawi Medical Journal*, 24(3):69–71, 2012.

[O'Mahony and Smyth 2007] Michael P O'Mahony and Barry Smyth. A recommender system for on-line course enrolment: an initial study. In *Proceedings of the 2007 ACM conference on Recommender systems*, pages 133–136. ACM, 2007.

[Osborne and Overbay 2004] Jason W Osborne and Amy Overbay. The power of outliers (and why researchers should always check for them). *Practical assessment, research & evaluation*, 9(6):1–12, 2004.

[Pyle 1999] Dorian Pyle. *Data preparation for data mining*, volume 1. morgan kaufmann, 1999.

[Sacin *et al.* 2009] Cesar Vialardi Sacin, Javier Bravo Agapito, Leila Shafti, and Alvaro Ortigosa. Recommendation in higher education using data mining techniques. In *Educational Data Mining 2009*, 2009.

[Said and Bellogín 2014] Alan Said and Alejandro Bellogín. Comparative recommender system evaluation: benchmarking recommendation frameworks. In *Proceedings of the 8th ACM Conference on Recommender systems*, pages 129–136. ACM, 2014.

[Seminario and Wilson 2012] Carlos E Seminario and David C Wilson. Case study evaluation of mahout as a recommender platform. In *RUE@ RecSys*, pages 45–50, 2012.

[Shani and Gunawardana 2011] Guy Shani and Asela Gunawardana. Evaluating recommendation systems. *Recommender systems handbook*, pages 257–297, 2011.

[Shearer 2000] Colin Shearer. The crisp-dm model: the new blueprint for data mining. *Journal of data warehousing*, 5(4):13–22, 2000.

[Thai-Nghe *et al.* 2010] Nguyen Thai-Nghe, Lucas Drumond, Artus Krohn-Grimberghe, and Lars Schmidt-Thieme. Recommender system for predicting student performance. *Procedia Computer Science*, 1(2):2811–2819, 2010.

[Vozalis and Margaritis 2003] Emmanouil Vozalis and Konstantinos G Margaritis. Analysis of recommender systems algorithms. In *The 6th Hellenic European Conference on Computer Mathematics & its Applications*, pages 732–745, 2003.

[Yoshii *et al.* 2008] Kazuyoshi Yoshii, Masataka Goto, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G Okuno. An efficient hybrid music recommender system using an incrementally trainable probabilistic generative model. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):435–447, 2008.

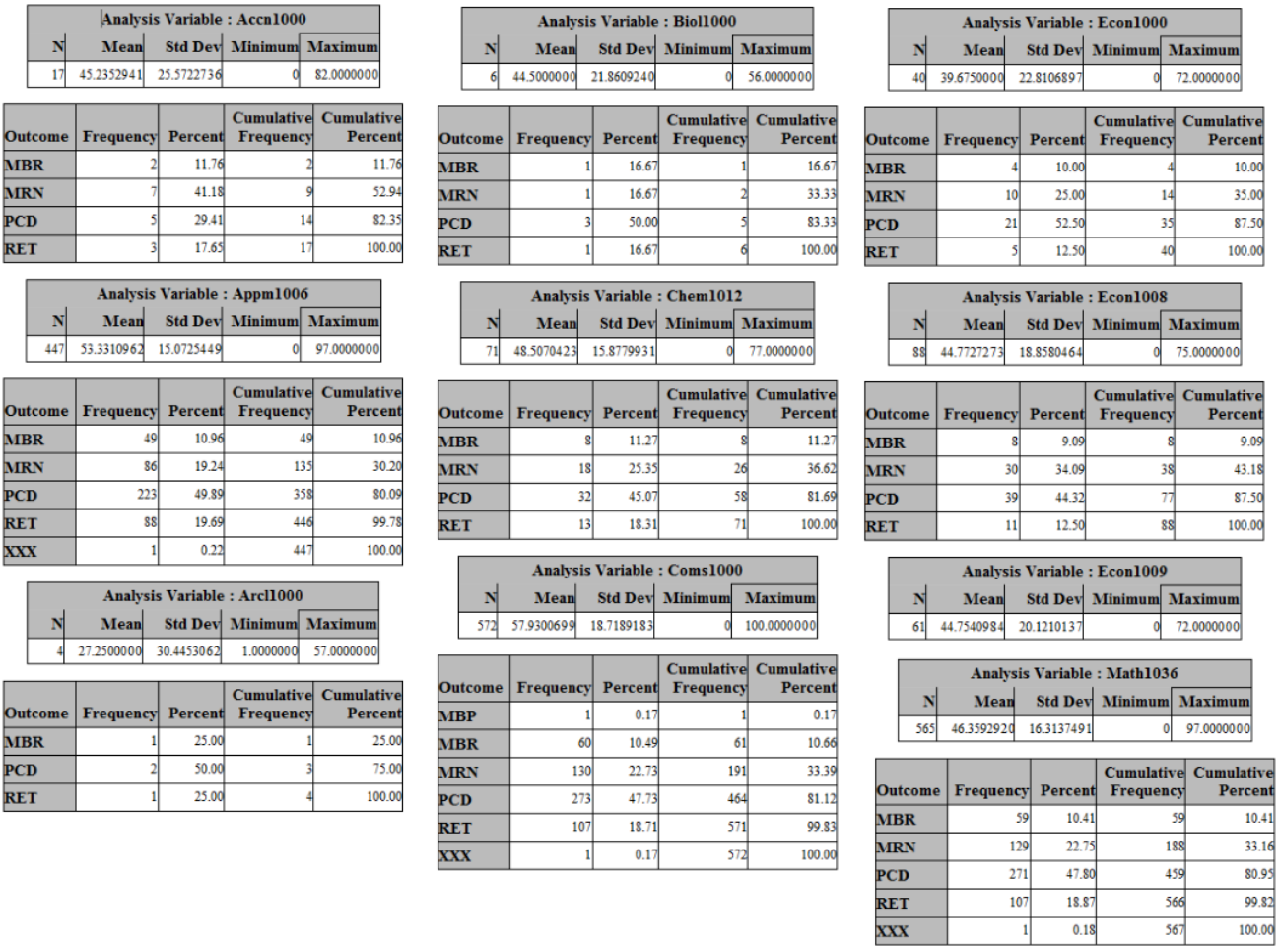
[Yuan *et al.* 2013] Nicholas Jing Yuan, Yu Zheng, Liuhang Zhang, and Xing Xie. T-finder: A recommender system for finding passengers and vacant taxis. *IEEE Transactions on Knowledge and Data Engineering*, 25(10):2390–2403, 2013.

[Zhang *et al.* 2003] Shichao Zhang, Chengqi Zhang, and Qiang Yang. Data preparation for data mining. *Applied Artificial Intelligence*, 17(5-6):375–381, 2003.

[Zhao *et al.* 2015] Aite Zhao, Zhiqiang Wei, and Yongquan Yang. Research on sqlite database encryption technology in instant messaging based on android platform. In *International Conference on Intelligent Science and Big Data Engineering*, pages 316–325. Springer, 2015.

Appendix A

Descriptive Statistics



Analysis Variable : Accn1000

N	Mean	Std Dev	Minimum	Maximum
17	45.2352941	25.5722736	0	82.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MBR	2	11.76	2	11.76
MRN	7	41.18	9	52.94
PCD	5	29.41	14	82.35
RET	3	17.65	17	100.00

Analysis Variable : Appm1006

N	Mean	Std Dev	Minimum	Maximum
447	53.3310962	15.0725449	0	97.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MBR	49	10.96	49	10.96
MRN	86	19.24	135	30.20
PCD	223	49.89	358	80.09
RET	88	19.69	446	99.78
XXX	1	0.22	447	100.00

Analysis Variable : Arcl1000

N	Mean	Std Dev	Minimum	Maximum
4	27.2500000	30.4453062	1.0000000	57.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MBR	1	25.00	1	25.00
PCD	2	50.00	3	75.00
RET	1	25.00	4	100.00

Analysis Variable : Biol1000

N	Mean	Std Dev	Minimum	Maximum
6	44.5000000	21.8609240	0	56.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MBR	1	16.67	1	16.67
MRN	1	16.67	2	33.33
PCD	3	50.00	5	83.33
RET	1	16.67	6	100.00

Analysis Variable : Chem1012

N	Mean	Std Dev	Minimum	Maximum
71	48.5070423	15.8779931	0	77.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MBR	8	11.27	8	11.27
MRN	18	25.35	26	36.62
PCD	32	45.07	58	81.69
RET	13	18.31	71	100.00

Analysis Variable : Coms1000

N	Mean	Std Dev	Minimum	Maximum
572	57.9300699	18.7189183	0	100.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MBP	1	0.17	1	0.17
MBR	60	10.49	61	10.66
MRN	130	22.73	191	33.39
PCD	273	47.73	464	81.12
RET	107	18.71	571	99.83
XXX	1	0.17	572	100.00

Analysis Variable : Econ1000

N	Mean	Std Dev	Minimum	Maximum
40	39.6750000	22.8106897	0	72.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MBR	4	10.00	4	10.00
MRN	10	25.00	14	35.00
PCD	21	52.50	35	87.50
RET	5	12.50	40	100.00

Analysis Variable : Econ1008

N	Mean	Std Dev	Minimum	Maximum
88	44.7727273	18.8580464	0	75.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MBR	8	9.09	8	9.09
MRN	30	34.09	38	43.18
PCD	39	44.32	77	87.50
RET	11	12.50	88	100.00

Analysis Variable : Econ1009

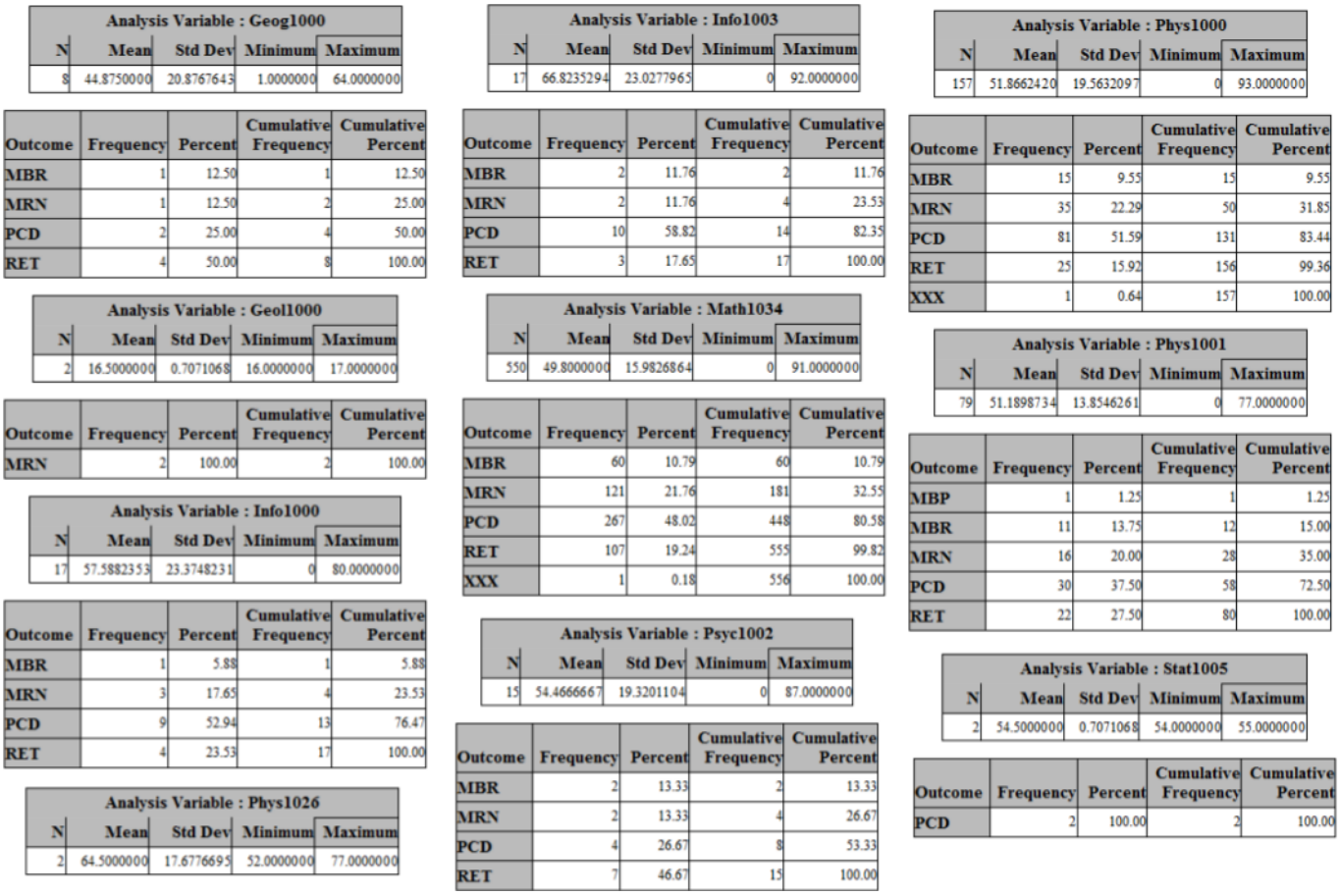
N	Mean	Std Dev	Minimum	Maximum
61	44.7540984	20.1210137	0	72.0000000

Analysis Variable : Math1036

N	Mean	Std Dev	Minimum	Maximum
565	46.3592920	16.3137491	0	97.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MBR	59	10.41	59	10.41
MRN	129	22.75	188	33.16
PCD	271	47.80	459	80.95
RET	107	18.87	566	99.82
XXX	1	0.18	567	100.00

Figure A.1: Descriptive Statistics for the Student Dataset (Author's own work)



Analysis Variable : Geog1000

N	Mean	Std Dev	Minimum	Maximum
8	44.8750000	20.8767643	1.0000000	64.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MBR	1	12.50	1	12.50
MRN	1	12.50	2	25.00
PCD	2	25.00	4	50.00
RET	4	50.00	8	100.00

Analysis Variable : Geol1000

N	Mean	Std Dev	Minimum	Maximum
2	16.5000000	0.7071068	16.0000000	17.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MRN	2	100.00	2	100.00

Analysis Variable : Info1000

N	Mean	Std Dev	Minimum	Maximum
17	57.5882353	23.3748231	0	80.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MBR	1	5.88	1	5.88
MRN	3	17.65	4	23.53
PCD	9	52.94	13	76.47
RET	4	23.53	17	100.00

Analysis Variable : Phys1026

N	Mean	Std Dev	Minimum	Maximum
2	64.5000000	17.6776695	52.0000000	77.0000000

Analysis Variable : Info1003

N	Mean	Std Dev	Minimum	Maximum
17	66.8235294	23.0277965	0	92.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MBR	2	11.76	2	11.76
MRN	2	11.76	4	23.53
PCD	10	58.82	14	82.35
RET	3	17.65	17	100.00

Analysis Variable : Math1034

N	Mean	Std Dev	Minimum	Maximum
550	49.8000000	15.9826864	0	91.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MBR	60	10.79	60	10.79
MRN	121	21.76	181	32.55
PCD	267	48.02	448	80.58
RET	107	19.24	555	99.82
XXX	1	0.18	556	100.00

Analysis Variable : Psyc1002

N	Mean	Std Dev	Minimum	Maximum
15	54.4666667	19.3201104	0	87.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MBR	2	13.33	2	13.33
MRN	2	13.33	4	26.67
PCD	4	26.67	8	53.33
RET	7	46.67	15	100.00

Analysis Variable : Phys1000

N	Mean	Std Dev	Minimum	Maximum
157	51.8662420	19.5632097	0	93.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MBR	15	9.55	15	9.55
MRN	35	22.29	50	31.85
PCD	81	51.59	131	83.44
RET	25	15.92	156	99.36
XXX	1	0.64	157	100.00

Analysis Variable : Phys1001

N	Mean	Std Dev	Minimum	Maximum
79	51.1898734	13.8546261	0	77.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MBP	1	1.25	1	1.25
MBR	11	13.75	12	15.00
MRN	16	20.00	28	35.00
PCD	30	37.50	58	72.50
RET	22	27.50	80	100.00

Analysis Variable : Stat1005

N	Mean	Std Dev	Minimum	Maximum
2	54.5000000	0.7071068	54.0000000	55.0000000

Outcome	Frequency	Percent	Cumulative Frequency	Cumulative Percent
PCD	2	100.00	2	100.00

Figure A.2: Descriptive Statistics for the Student Dataset-part 2 (Author's own work)