UNIVERSITY OF PERADENIYA

FACULTY OF SCIENCE

UNIVERSITY COURSE RECOMMENDATION ENGINE

DECLARATION

We hereby declare that the Project Summary Report entitled University Course Recommendation Engine is an authentic record of our own work as a requirement of the three-months project under the course of ‘Independent study in Data Science (DSC3263)’ during the period from 03rd of November 2023 to 19th of February 2024 for the award of the degree of B.Sc. Honors Study in Data Science from the Department of Statistics and Computer Science Faculty of Science University of Peradeniya, under the guidance of Dr. Roshan D. Yapa and Dr. Sachith P. Abeysundara.

[illegible]
....
K.M.P.Kanchanamala
S/18/412

[illegible]
....
H.R.N.B.Meththananda
S/18/445

[illegible]
....
U.G.D.Sashinka
S/18/505
Date: 19-02-2024

Certified by:

1. Supervisor (Name):Dr.Roshan D. Yapa.

   Date: …………………..

   (Signature) :………………………..

2. Head of the Department (Name): Dr. Sachith P. Abeysundara.

   Date: …………………..

   (Signature):………………………..

   Department Stamp:

ABSTRACT

Recommendation systems are widely used in many fields. These systems work by recommending a personalized list of items to users based on their interests and thus helping users to overcome excessive information offered to them. For users such as students, selecting the right courses is a very challenging task while joining a new academic level. Picking the wrong courses may affect a student's academic life as well as their future career. This project develops a personalized course recommendation system for the Faculty of Science at the University of Peradeniya to address challenges in course registration and finding related courses. Using memory-based collaborative filtering with cosine similarity, similar students' data informs recommendations, enhancing the registration process within the Learning Management System (LMS). We experimentally evaluate the collaborative filtering approach on a real-life dataset and demonstrate the effectiveness of the proposed system using performance metrics and the coverage metric

Key words : Collaborative Filtering, Cosine Similarity, Course Recommender System, Machine Learning

ACKNOWLEDGEMENTS

We extend our sincere appreciation to Dr. Roshan D. Yapa, our internal supervisor, and Ms. Ishani Udheshika from Zone 24x7, our external supervisor, for their invaluable time, unwavering support, and guidance throughout this project. We are truly grateful for their expertise and mentorship.

We also wish to express our gratitude to the esteemed members of the panel and review committee, Dr. Ruwan D. Nawarathna, Dr. Jagath Senarathne, Dr. Hemalika T.K. Abeysundara, and Ms. B. R. Pavithra M. Basnayake, for their valuable insights and feedback, which significantly enriched this work.

Furthermore, special thanks go to Ms. Sathya K. Wijewardhana for her invaluable support in data collection and guidance throughout the project. Her contributions have been instrumental to the success of this endeavor. We also extend our gratitude to D.W.N.C. Bandaranayake for his support in data collection, which greatly facilitated the progress of this research.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 01

1.1 INTRODUCTION

1.1.1 Background

In today's educational landscape, technological advancements play a pivotal role in optimizing the learning experience. Recommender Systems are crucial in higher education for various applications, including e-learning, library services (Jomsri, 2018), research and publications. They help generate relevant materials, provide easy access to books, and recommend courses to students (Al-Badarenah & Alsakran, 2016; Rahman et al., 2022). These systems are essential for enhancing the learning experience and promoting innovation. The Learning Management System (LMS) is an e-learning platform which is considered as an important part of e-learning solutions from the university's viewpoint. This project aims to develop a personalized course recommendation engine for students, integrated into the Faculty of Science's LMS at the University of Peradeniya.

1.1.2 Problem Statement

The current LMS used by the Faculty of Science at the University of Peradeniya lacks a personalized recommendation engine, leading to challenges for students during course registration and seeking related courses based on their academic backgrounds. This project aims to address these issues by developing a course recommendation engine exclusively for science faculty students within the university's LMS, providing recommendations for courses that students can take for a semester.

1.1.3 Objectives

The main objective of this project is to develop a model for course recommendation that suggests a list of courses to a student based on similar senior students' data.

Specific objectives:

- To investigate how to incorporate past students' data to recommended courses.
- To investigate which type of recommender system can be used to recommend courses to students based on similar past students' data.
- To design a model for course recommendation to students.
- To generate a web application to effectively showcase the outcomes of the project.

1.2 Literature Review

Information workload has become an increasingly important issue these days. No matter what information items a person is interested in knowing more about, there are some recommender systems (RSs) online that recommend relevant items. Recommender systems have shown great potential to help users find interesting and relevant items within a large information space.

In (Itmazi & Megías, 2008), authors emphasizes the importance of recommendation systems in everyday internet usage, highlighting how people rely on recommendations from others. It discusses the ability of recommendation systems, particularly within Learning Management Systems (LMS), to support students' needs. Additionally, it explores the suitability of different recommendation system approaches for recommending learning objects. It elaborates on various approaches, including collaborative filtering, content-based filtering, demographic-based systems, and rule-based filtering.

A content-based recommendation system is a type of recommendation system that suggests items to users based on the characteristics or attributes of those items and the user's preferences. Instead of relying on similarities between users or items, as in collaborative filtering, content-based systems focus on the content of the items being recommended. One drawback of this approach is that it demands in-depth knowledge of the item features/description for accurate recommendation. This knowledge or information may not always be available for all items. Also, this approach has limited capacity to expand on users' existing choices or interests. However, this approach has many advantages. As user preferences tend to change with time, this approach has the quick capability of dynamically adapting itself to the changing user preferences (Roy & Dutta, 2022).

Collaborative filtering is a technique used by recommendation systems to predict a user's preference for items or products based on the preferences or behavior of similar users. Traditionally, collaborative filtering-based systems suffer from the cold-start problem and privacy concerns as there is a need to share user data. However, collaborative filtering approaches do not require any knowledge of item features for generating a recommendation (Roy & Dutta, 2022).

In (Ray & Sharma, 2013), authors propose a course recommendation system that helps students in making choices about courses that have become more relevant out of a wide range of different elective courses. The proposed system uses the collaborative filtering approach to predict students' an accurate prediction of the grade they may get if they choose a particular course, which will be helpful when they decide on selecting elective courses, as grade is an important parameter for a student while deciding on an elective course.

Collaborative approaches are divided into two types: memory-based approaches and model-based approaches. Model-based systems use various data mining and machine learning algorithms to develop a model for predicting the user’s rating for an unrated item. Memory-based collaborative approaches recommend new items by taking into consideration the preferences of its neighborhood. Memory-based collaborative approaches are again subdivided into two types: user-based collaborative filtering and item-based collaborative filtering (Roy & Dutta, 2022).

The main difference between item-based CF (Sarwar et al., 2001) and user-based CF (Zhi-Dan Zhao & Ming-Sheng Shang, 2010) is that item-based CF generates predictions based on a model of item-item similarity rather than user-user similarity.

In the user-based approach, the user receives products that similar users like. On the other hand, in the item-based approach, the user receives recommendations based on the similarity between the items he likes. There are many similarity measure methods: Pearson correlation-based similarity, constrained Pearson correlation-based similarity, cosine-based similarity, or even adjusted cosine-based measure (Estrela et al., 2017; Fkih, 2021).

Some of the drawbacks in pure content-based and collaborative filtering recommendation systems are Cold start Sparsity, Grey-sheep, Scalability, handling high-dimensional data, and handling heterogeneous data types. Hybrid RSs have widely shown improved outcomes rather than any standalone filtering technique. Several hybrid combinations between CF and DF techniques have been proposed. Such hybridization minimizes the limitations of CF, such as the sparsity and the cold start problem (Lahoud et al., 2022).

One of the most challenging problems in collaborative filtering is handling users whose previous item purchase behavior is unknown (e.g., new users) or products for which user interactions are not available (e.g., new products) (cold start problem). (Panteli & Boutsinas, 2023) proposed a methodology to handle the cold-start problem and sparsity problem in recommender systems based on frequent patterns which are highly frequent in one set of users but less frequent or infrequent in other sets of users.

(Wei et al., 2017)Proposed models integrate CF with deep learning, outperforming baselines on cold start items, with potential for broader application.

In (Estrela et al., 2017), authors proposed a recommendation system for online courses which utilizes both collaborative filtering and the content-based to offer more accurate recommendations and only consider the interest of each user. The hybrid technique overcomes the disadvantages of traditional recommendation techniques such as the cold start problem and the sparseness and problems of scalability. (Vizine Pereira & Hruschka, 2015).

The above literature review focuses on various recommendation systems including course recommendations in higher education. The studies discussed different approaches, such as collaborative filtering , content-based filtering, and hybrid methods, to build personalized course recommendation systems. The proposed course recommender system for university students aims to leverage the insights and techniques discussed in the literature review. Specifically, the proposed system will use collaborative filtering method with cosine similarity to provide personalized course recommendations based on similar senior students' data. The system will analyze data from 976 students to identify courses. Macro-average precision, recall and F1 score will be used to evaluate the system's performance, as it has been used in previous studies. The proposed system has the potential to assist students in selecting the most appropriate elective courses, improve their academic performance, and enhance their educational experience.

CHAPTER 02

2.1 Methodology

The main objective is to develop a specialized recommendation engine tailored to the Science Faculty's Learning Management System (LMS). This involves offering students personalized course recommendations to streamline the course registration processes. Furthermore, the goal is to enhance the overall student experience within the LMS by providing tailored course suggestions. Additionally, provision is made for supplementary and compulsory courses separately, considering pre-requirements and catering to individual student needs. After creating the recommendation engine, a web application is developed to effectively showcase the outcomes of the project.

The generic workflow diagram presented in Figure 2.1 outlines the main steps of this study. Initially, the dataset is acquired from the LMS database and undergoes preprocessing to ensure it is in the correct format. Next, the memory-based collaborative filtering model, utilizing the cosine similarity algorithm, is implemented. The model's performance is evaluated through various methods on the test set, and adjustments are made to enhance it. Finally, the model is deployed on the web application, showcasing the ultimate project outcome. The overall steps involved in this work are as follows:

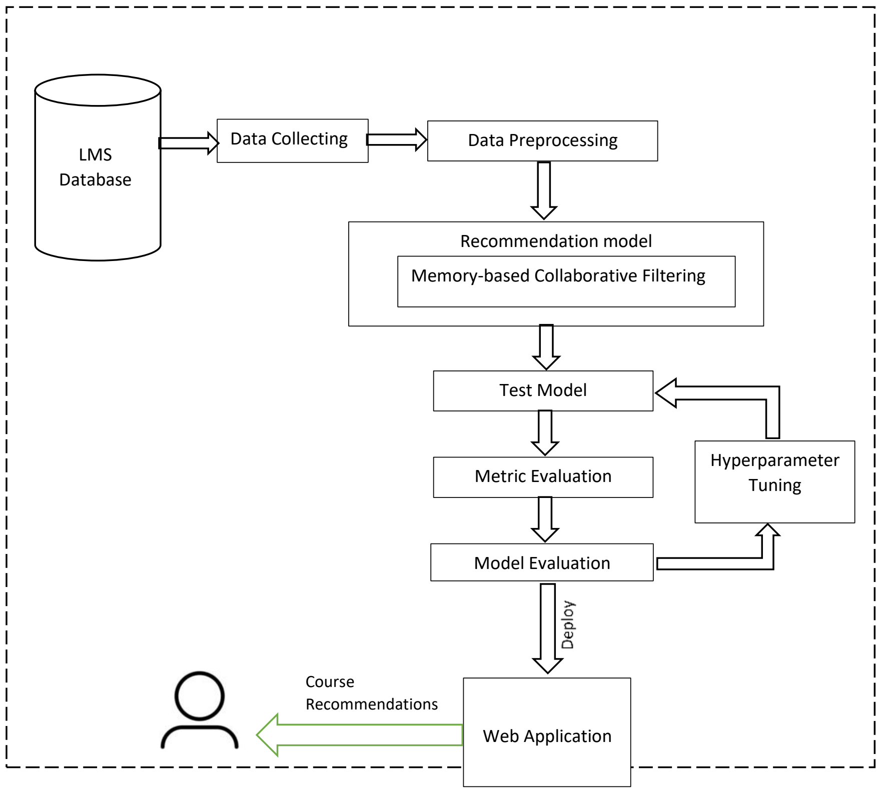


Figure 2.1 Generic workflow diagram

2.1.1 Data Collection

In this study, two datasets were obtained from the Course Unit of the Faculty of Science at the University of Peradeniya. The first dataset comprises student enrollment data, offering comprehensive information about course registrations. Attributes in this dataset include Registration Number (RegNo), Course Code, Course Name, Grade, Remark, Academic Year, Semester, and Credits. Notably, the student enrollment data covers three batches: 16, 17, and 18, and it includes records for a total of 976 unique students.

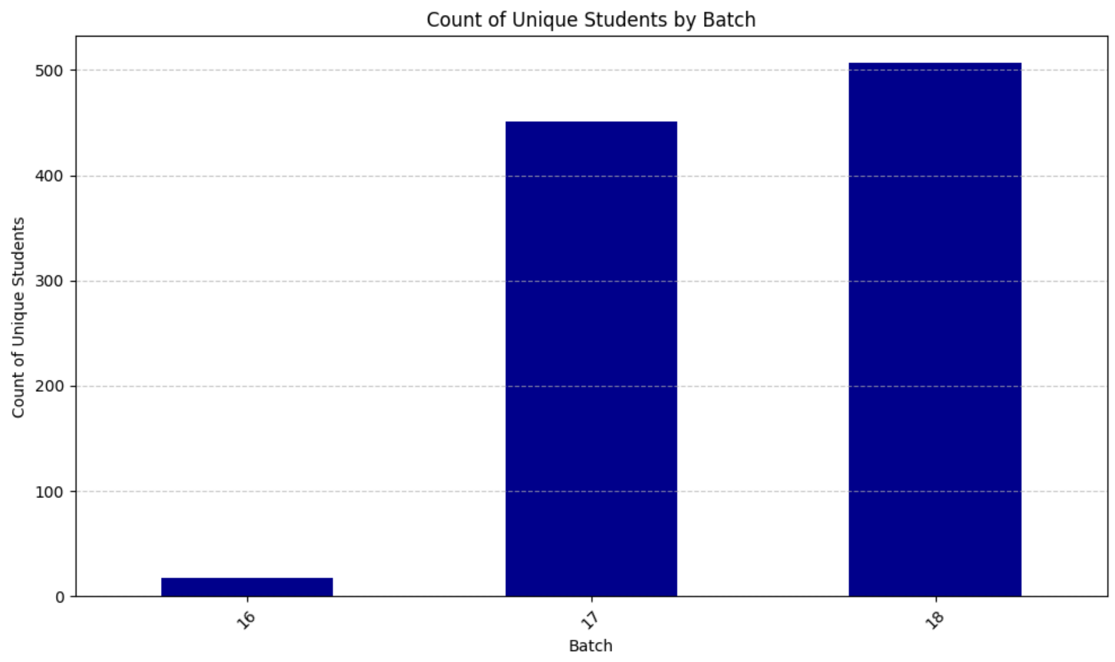


Figure 2.2 Number of students by batch

The second dataset pertains to course data, furnishing details about the courses available within the faculty. This dataset encompasses information on 737 unique course codes and includes attributes such as Course Code, Level, Course Name, Department, Credits, and a Description of each course. These datasets serve as the foundation for our analysis and exploration of student enrollment patterns and course offerings within the Faculty of Science.

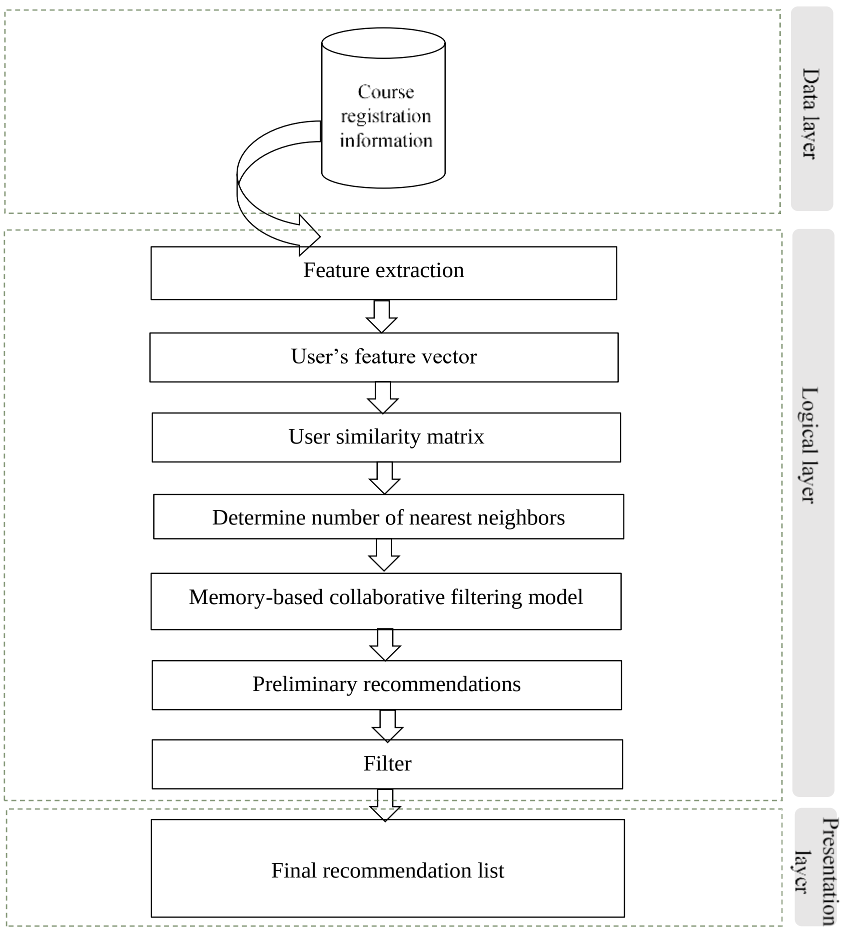
2.1.2 Data Preprocessing

During the Data Preprocessing phase, the initial focus was on data cleaning to ensure the dataset's quality and consistency. This involved identifying and handling missing values, anomalies, and outliers using established techniques and referencing relevant resources for best practices. Following this, the dataset underwent transformation tasks to standardize numerical features and enhance its readiness for analysis. Ultimately, after completing the feature engineering process, which involved creating new features or transforming existing ones to improve predictive performance, the dataset comprised essential attributes such as RegNo, courseCode, courseName, grade, remark, academic_year, semester, credits, year, subject, batch, Compulsory_for, Pre-requisites, Department, Combination, and degree. This phase aimed to refine the dataset, making it suitable for subsequent analysis and decision-making processes.

2.1.2 Proposed method

The memory-based collaborative filtering method (Roy & Dutta, 2022) utilizing cosine similarity(Estrela et al., 2017; Fkih, 2021), is specifically tailored to construct a course recommender system customized for the needs of students in the upcoming semester. This approach underscores the significance of leveraging crucial features extracted from students' course enrollment histories. By computing cosine similarity among students based on these features, the system identifies similar students who share common interests. It then utilizes this information to predict courses for the next semester. This method operates on the assumption that users with similar past behaviors are likely to exhibit similar future preferences. Consequently, by utilizing the behavior patterns of similar students, memory-based collaborative filtering effectively taps into the collective wisdom of the user base to generate personalized recommendations tailored to individual students' academic journeys.

This method works by inputting important features, calculating similarity, determining the number of nearest neighbors who share similar interests, and predicting the courses for the new semester.



Course registration information
Feature extraction
User’s feature vector
User similarity matrix
Determine number of nearest neighbors
Memory-based collaborative filtering model
Preliminary recommendations
Filter
Final recommendation list
Data layer
Logical layer
Presentation layer

Figure 2.3 Workflow diagram of memory-based collaborative filtering

- **Input features**

  Input features play a crucial role in calculating the user-to-user similarity matrix. The proposed model uses the 'courseCode', 'Combination', and 'year' as the input features. These features capture important aspects of student behavior, such as the courses they have enrolled in, the combination, and the academic years in which they have participated.

- **Calculate user-item matrix**

  The user-item matrix serves as the foundation for calculating the user-to-user similarity matrix in the the course recommender system. The user-item matrix encapsulates the interactions between student and input features('courseCode', 'Combination', and 'year'). Each entry in the matrix represents whether a user has interacted with the input features ('courseCode', 'Combination', and 'year'), encoded as binary values (1 for interaction, 0 for no interaction).

- **Calculate the user similarity matrix using cosine similarity**

  The user similarity matrix, derived from cosine similarity on feature vectors of the calculated user-item matrix.

$$Sim(A,B) = \cos\theta = \frac{A.B}{|A||B|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

Similarity of user A and B can be calculated as,

The similarity between users A and B can be calculated using the user-item interaction data represented in Table 2.1 An example of user-item matrix, which presents a matrix where users' interactions with items are recorded. In this matrix, each row corresponds to a user, labeled as User A or User B, and each column represents a course or combination or year. The values in the matrix indicate whether a user has interacted with a particular item, with 1 indicating interaction and 0 indicating no interaction.

Table 2.1 An example of user-item matrix

	course 1	…	course N	combination 1	…	combination M	year 1	…	year 4
User A	1	…	0	1	…	1	1	…	0
User B	0	…	0	1	…	0	1	…	1

To compute the similarity between users A and B, cosine similarity is applied to the rows corresponding to User A and User B in the user-item matrix. The resulting user similarity matrix, as shown in Table 2.2, presents a measure of similarity between each pair of users in the dataset.

Table 2.2 An example of user similarity matrix

	User A	User B
User A	1	…
User B	…	1

Determine the number of nearest neighbors (k)

The success of collaborative filtering systems heavily relies on neighborhood selection, essentially grouping similar users based on their past interactions. The user similarity matrix measures the cosine similarity for the feature vector for each pair of users in the system. Then, selecting the number of neighbors for recommending their courses is required. A larger value of k leads to fewer recommendations, while having more neighbors may result in unnecessary recommendations for the active user. Therefore, the size of the neighbors must be selected carefully. The proposed method chooses the optimal size of neighbors by comparing different sizes of neighbors with the evaluation metrics.

Memory-based collaborative filtering model

The function operates by taking the user's registration number, combination number, and the current year as inputs to generate recommendations for the upcoming semester. It begins by identifying the nearest neighbors (k) for the user from the user-to-user similarity matrix, which encapsulates the similarity scores between users based on their interactions with each item (courseCode, combination, year). Subsequently, it retrieves the courses taken by these nearest neighbors, which serve as potential recommendations for the user. These recommendations are filtered based on the user's eligibility to enroll in each course, considering prerequisites and other eligibility criteria. Finally, the function outputs a list of supplementary courses and compulsory courses tailored to the user's degree program.

This algorithm is described as follows,

***Algorithm 1**: Memory-based Collaborative Filtering Technique for Course Recommendation*

```
Input: Student registration number, year, semester
Output: List of course recommendations

1. Begin
2. Apply pivot function on (courseCode, combination number, year) to create
   user-item matrix
3. If the user has interacted with specific categories of courseCode, combination
   number, and year,
           assign 1 to the corresponding cell
   Else
           assign 0 to the corresponding cell.
4. Compute the cosine similarity between users based on feature vectors of the
   user-item matrix
5. Get the user similarity matrix
6. Define the number of neighbors (k)
7. Find the k-nearest neighbors for the target user based on cosine similarity
8. Retrieve unique courses for the given year and semester for the k-nearest
   neighbors.
9. Check if the target user has taken the prerequisite courses for the recommended
   courses.
10. Recommend courses to the user
11. End
```

- **Predict courses for the upcoming semester**
  After determining the nearest neighbors (k), the recommendation system predicts courses for the next semester by analyzing the courses taken by similar students. By leveraging the similarities between students' interests and behaviors, the system identifies courses taken by the nearest neighbors but not yet taken by the target student. If any of the recommended courses have prerequisites, the system verifies that the target student meets the eligibility criteria before including them in the recommendation list.

- **Cold start problem**
  The proposed collaborative filtering course recommender system faces the "cold start" issue (Panteli & Boutsinas, 2023; Wei et al., 2017), particularly when new users join the system without any historical data. This lack of user information presents a significant challenge for conventional collaborative filtering methods, resulting in subpar recommendations. To address this challenge, an explicit approach is employed, integrating a combination of the new user's combination number (e.g., 1, 2, 3, …) and their stream (biological or physical) into the collaborative filtering technique. By incorporating this additional user context, the

system can tailor recommendations more effectively, alleviating the cold start problem and offering personalized suggestions even for newly onboarded users, such as students in their first year, first semester. This explicit integration of user characteristics significantly enhances the adaptability and accuracy of the recommendation system, ensuring a more satisfying user experience from the outset.

2.1.3 Model evaluation

Splitting data

When partitioning the dataset for the proposed collaborative filtering-based course recommendation system, the data is divided into training and testing sets while ensuring the integrity of student records. Since the dataset includes multiple entries for the same student across various academic years and semesters, it was imperative to maintain the continuity of student data within each set. To accomplish this, all instances related to a specific registration number (RegNo) were allocated entirely to either the training or testing set. This approach was crucial for preserving the underlying patterns of user preferences, which are essential for collaborative filtering. By adopting a 75:25 split ratio for training and testing, respectively, the aim was to strike a balance between effectively training the model and reliably evaluating its performance.

Evaluation metrics

In order to evaluate the quality of the course recommendation system, precision, recall, and F1-score are used on the test set.

Precision measures the capability of the system to recommend as many relevant courses as possible to the user. It is calculated as the ratio of the number of correctly recommended courses to the total number of courses recommended. A high precision indicates that a large proportion of the recommended courses are relevant to the user's preferences.

$$Precision = \frac{\sum Actual\_courses \ \cap \ \sum Recommended\_courses}{\sum Recommended\_courses}$$

Recall, on the other hand, measures the ability of the system to retrieve all relevant courses for the user. It is calculated as the ratio of the number of correctly recommended courses to the total number of actual courses taken by the user.

$$Recall = \frac{\sum Actual\_courses \ \cap \ \sum Recommended\_courses}{\sum Actual\_courses}$$

F1-score, which is the harmonic mean of precision and recall, provides a balanced measure of the system's performance, considering both precision and recall simultaneously

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Finally, the macro average precision, recall, and the F1-score are calculated to obtain the final metrics.

Macro average precision is calculated by summing up all precision values for all the instances (unique combination of registration number, year, semester) in the test set and then dividing by the number of instances.

$$Precision_{macro-averge} = \frac{Precision_1 + Precision_2 + \cdots + Precision_N}{N}$$

Macro average recall is calculated by summing up all recall values for all the instances in the test set and then dividing by the number of instances.

$$Recall_{macro-averge} = \frac{Recall_1 + Recall_2 + \cdots + Recall_N}{N}$$

where N is the total number of instances, and Precision1, Precision2, ..., PrecisionN and Recall1, Recall2, ..., RecallN are the precision and recall values for each instance.

Metrics beyond accuracy

Beyond accuracy, recommender systems also prioritize coverage metrics such as catalog coverage and user coverage, ensuring a diverse range of recommended items and personalized suggestions for a larger user base (Ge et al., 2010). Coverage is important as it provides valuable insights about courses that are less popular among students or no longer followed by them, prompting faculty to assess their relevance and potentially revise or remove them from offerings

Coverage

Coverage associated with two concepts. (I) Catalog coverage and (II) user coverage are two fundamental metrics in recommender systems.(Herlocker et al., 2004).

i. Catalog coverage (CC)

Catalogue coverage quantifies the percentage of items within the entire catalog for which the system can generate recommendations.

$$CC = \frac{Number\ of\ unique\ items\ recommended}{Total\ number\ of\ items\ in\ the\ system} \times 100\%$$

Number of unique items recommended: The count of distinct items recommended by the system.
Total number of items in the catalog: The total count of items available in the catalog.

ii. User coverage

User coverage evaluates the percentage of users for whom the system can effectively provide at least one recommendation from the available items

$$UC = \frac{Number\ of\ users\ with\ atleast\ one\ recommendation}{Total\ number\ of\ users\ in\ the\ system} \times 100\%$$

Number of users with at least one recommendation: The count of users who receive at least one recommendation from the system.

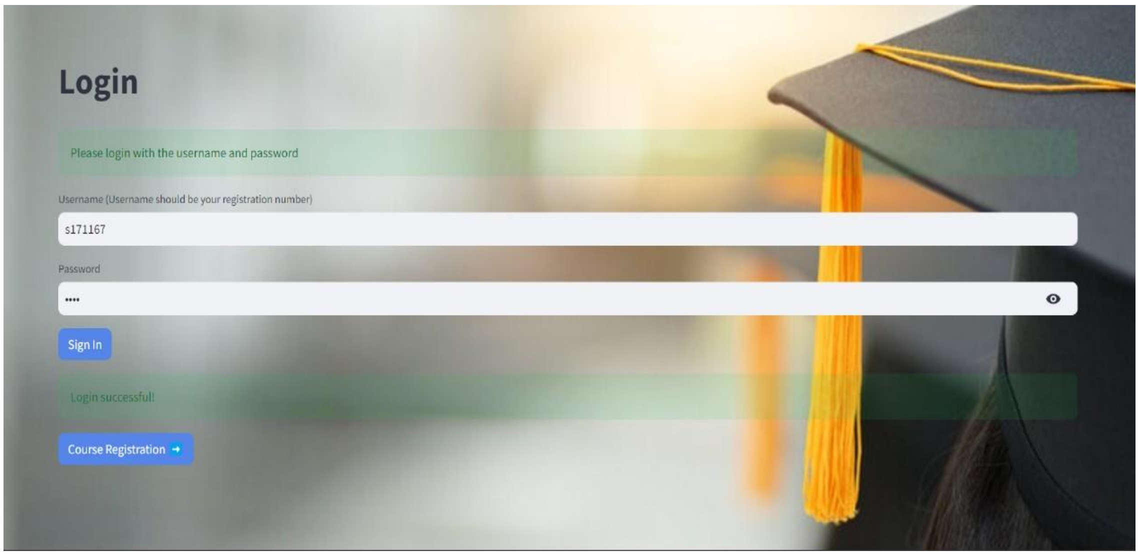
Total number of users in the system: The total count of users present in the system.

2.1.4 Developing the web application

We develop a web application for our course recommendation system to effectively showcase the outcomes of the project. Streamlit is a Python tool that we selected for developing it.

MySQL is utilized for seamless data management and retrieval within our web application, enhancing its reliability, scalability, and performance. This relational database management system ensures efficient storage and retrieval of user and course data.

The current user of the web application can log in by entering their username and password.



Login

Please login with the username and password

Username (Username should be your registration number)

s171167

Password

••••

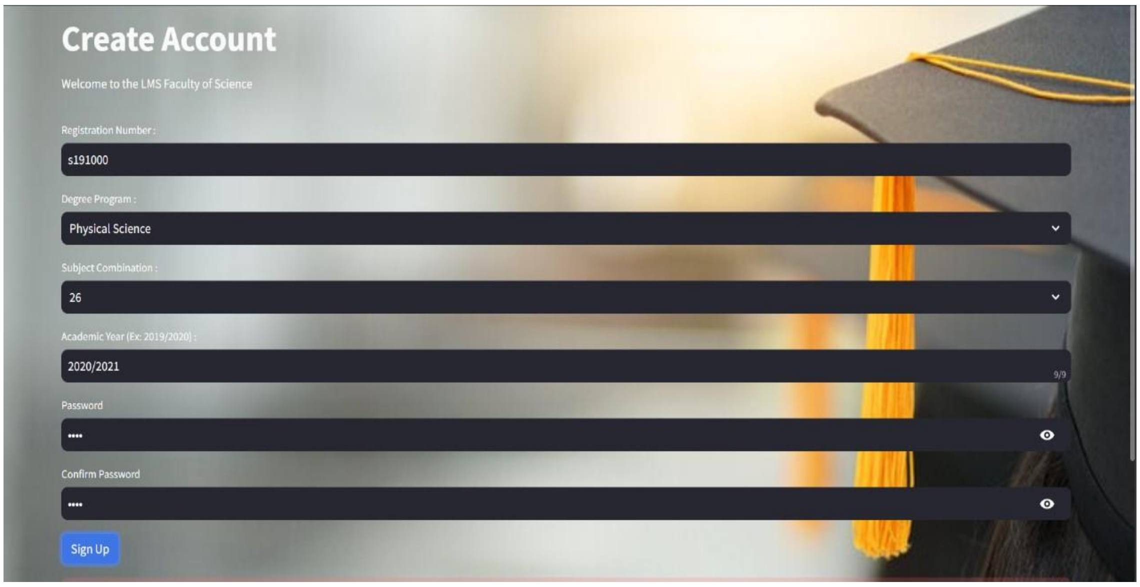
Sign In

Login successful!

Course Registration

Figure 2.4 Web application - user login page

If the user is new to the web application, they need to sign up first before being able to log in.



Create Account

Welcome to the LMS Faculty of Science

Registration Number : s191000

Degree Program : Physical Science

Subject Combination : 26

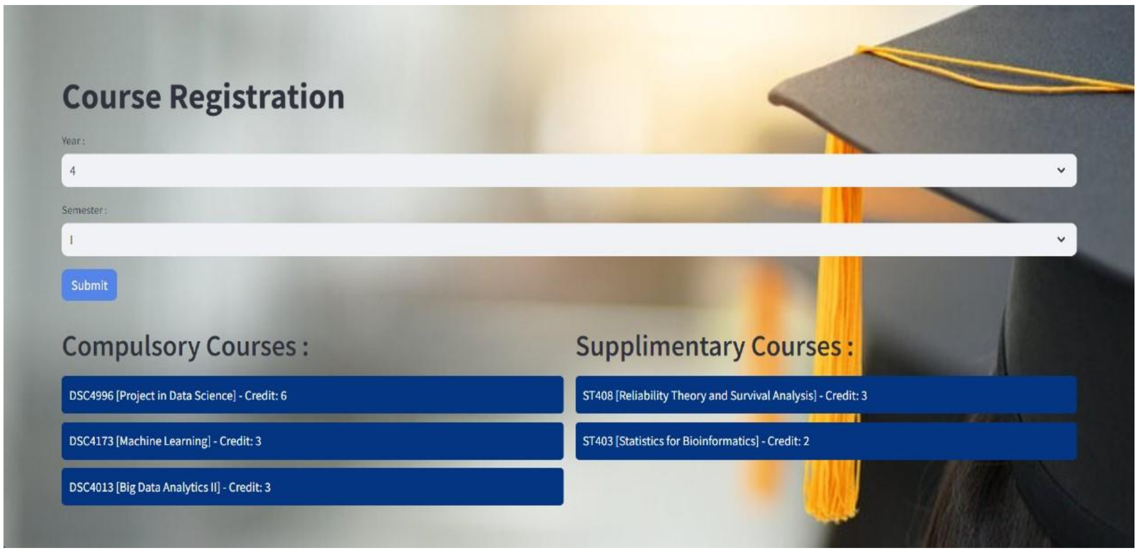
Academic Year (Ex: 2019/2020) : 2020/2021

Password ••••

Confirm Password ••••

Figure 2.5 Web application - user sign up page

Once signed up and logged in successfully, the user gains access to the course registration feature. In the course registration section, the user is prompted to input the academic year and semester they intend to register for. Based on this input, our recommendation system generates suggestions for compulsory and supplementary courses that the user may consider registering for.



Course Registration

Year : 4

Semester : I

Compulsory Courses :

DSC4996 [Project in Data Science] - Credit: 6

DSC4173 [Machine Learning] - Credit: 3

DSC4013 [Big Data Analytics II] - Credit: 3

Supplimentary Courses :

ST408 [Reliability Theory and Survival Analysis] - Credit: 3

ST403 [Statistics for Bioinformatics] - Credit: 2

Figure 2.6 Web application – display recommendations for a user

CHAPTER 03

3.1 Results

3.1.1 Descriptive Analysis

Descriptive analysis is crucial for understanding data trends. In our study, we employ it to explore enrollment patterns, student performance, and course offerings at the Faculty of Science, University of Peradeniya. Through visualizations like bar charts, we uncover key insights to inform educational planning.

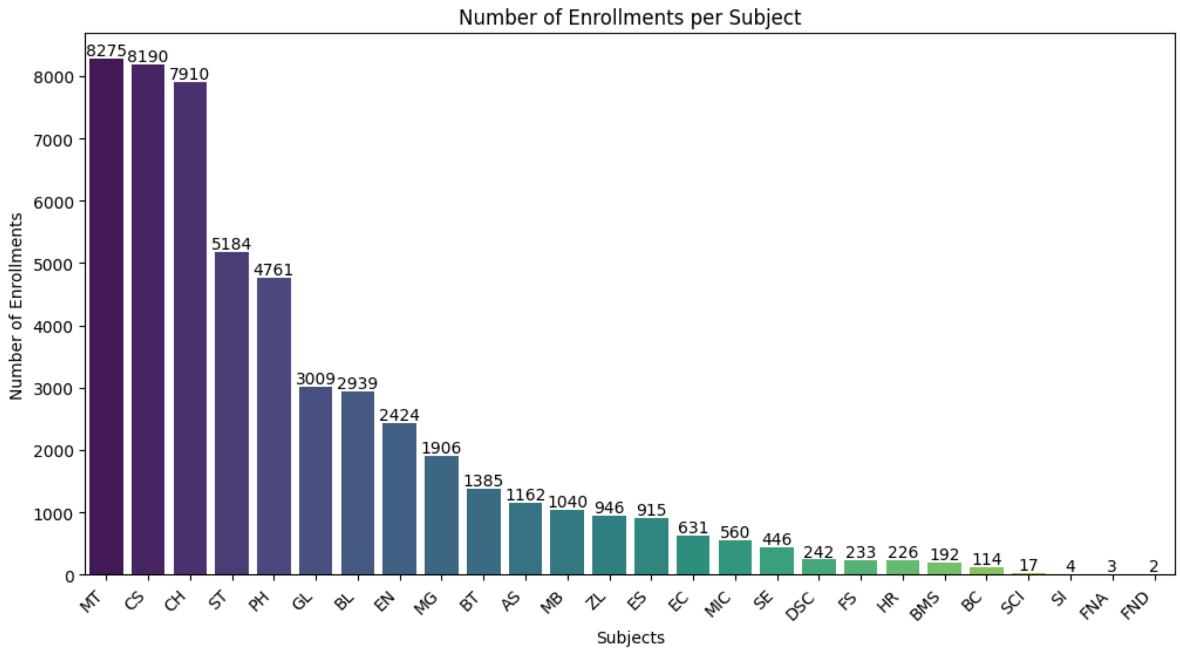


Figure 3.1 Number of enrollments per subject

The bar Figure 3.1 illustrates the enrollment distribution across different subjects. Each bar represents a subject, with its height indicating the number of enrollments. Subjects with taller bars have higher enrollments, while those with shorter bars have fewer enrollments. This visualization provides insights into the popularity of subjects among students and helps identify areas of academic interest within the dataset.

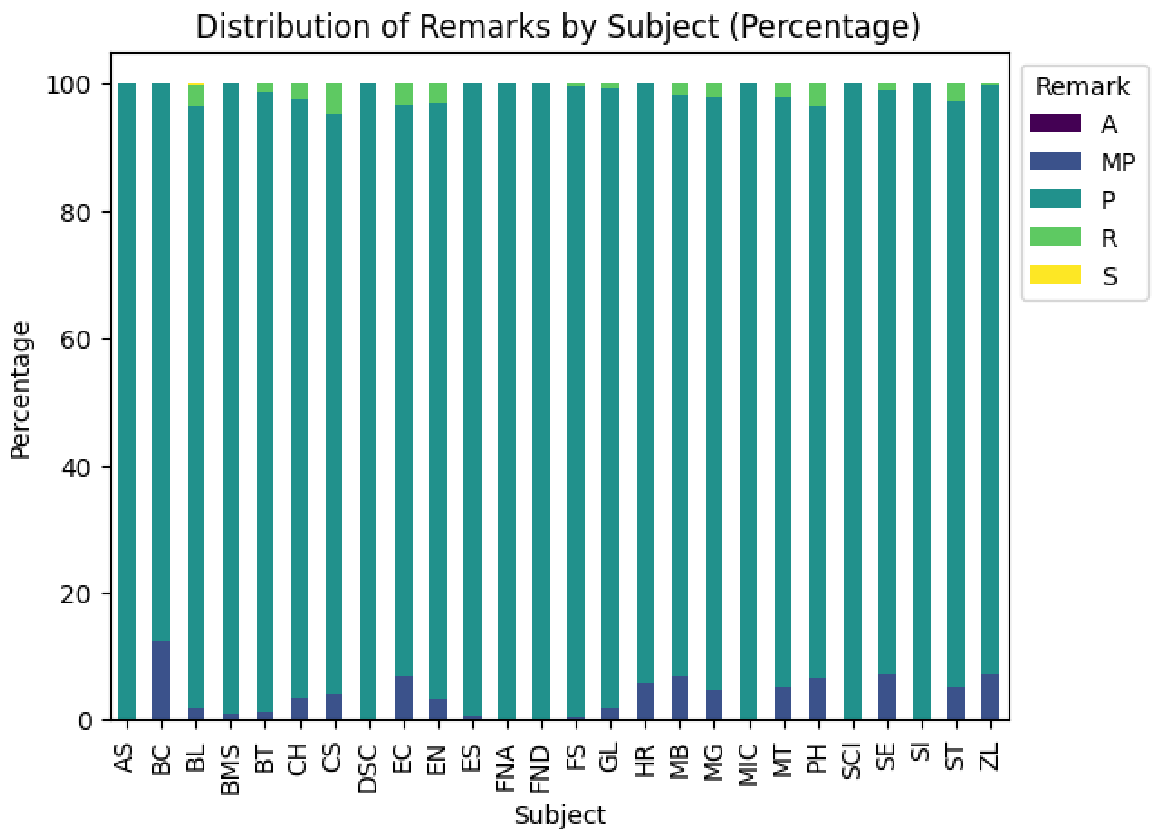


Figure 3.2 Distribution of remarks by subject

The stacked bar Figure 3.2 displays the percentage distribution of remarks across different subjects. Each bar represents a subject, segmented to show the proportion of each remark type. Mainly 'MP' indicates Medical Proper Enrollments, 'P' signifies Proper Enrollments, and 'R' represents Repeat Enrollments. The chart allows for easy comparison of remark distributions across subjects, offering insights into enrollment status and academic performance. The legend clarifies the meaning of each colored segment.

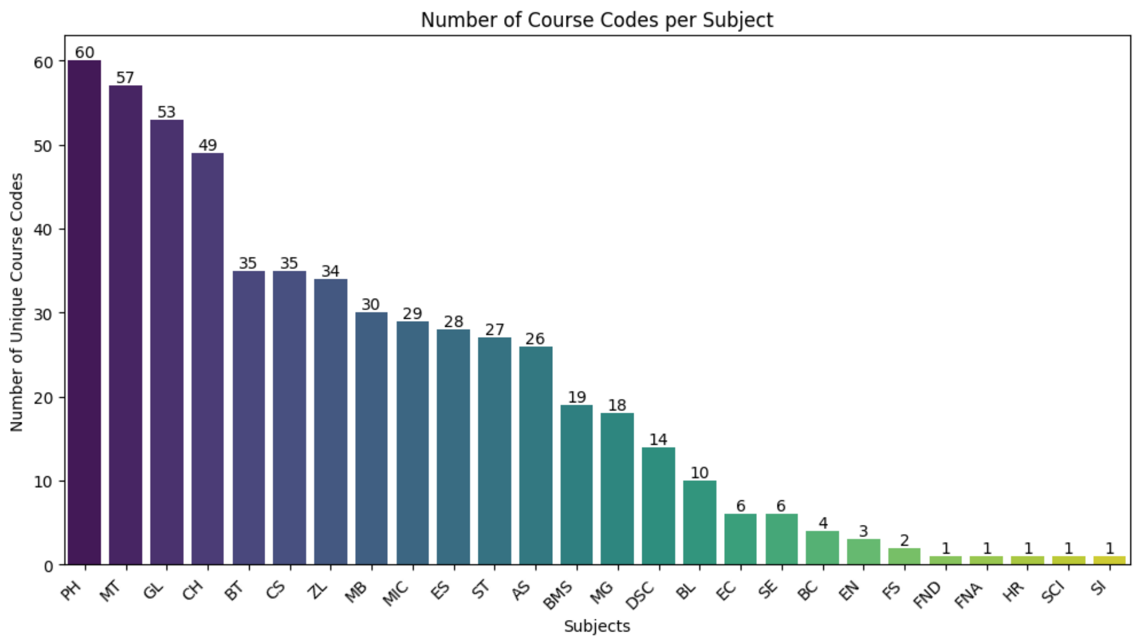


Figure 3.3 Number of different courses per subject

The bar Figure 3.3 illustrates the number of unique course codes per subject. Each bar represents a subject, with the height indicating the count of unique course codes within that subject. The chart provides insights into the diversity of course offerings across different subjects within the Faculty of Science. Subjects with taller bars have a greater variety of unique course codes, suggesting a broader range of academic offerings within those disciplines. This visualization aids in understanding the depth and variety of courses available to students within each subject area.

Evaluation of the Model

The model performance was evaluated by measuring its efficiency and effectiveness of recommending suitable courses to students.

Experiment Steps:

Our test consists of two main experiments. The first experiment focuses on comparing the evaluation metrics with different sizes of neighborhoods, while the second experiment involves comparing the coverage metrics with varying neighborhood sizes. Both experiments were conducted on the test set, which comprises 75% of the student_courses dataset.

Experiment 1. Determine the number of nearest neighbors for tuning

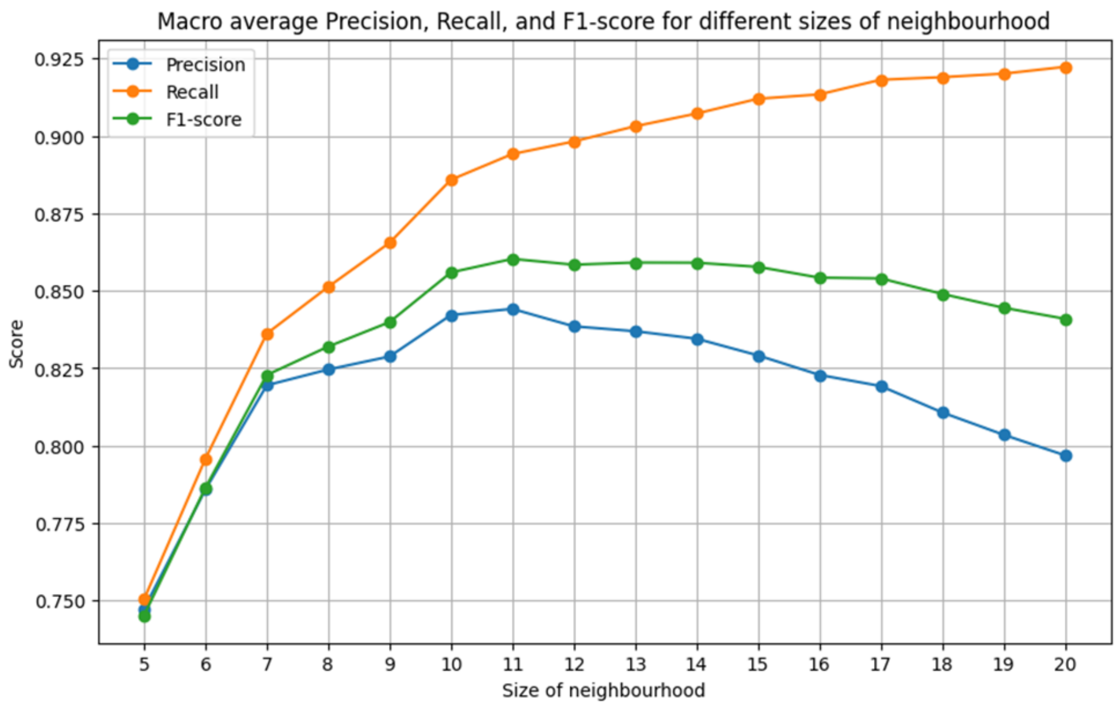


Figure 3.4 Macro average Precision, Recall, and F1 score for different size of neighborhood

The

Figure 3.4 illustrates the experimental results of the macro average precision, recall, and F1 score with different sizes of neighborhoods (k). The experiment analyzes the metric evaluation scores across a range of neighborhood sizes from 5 to 20. As the number of neighbors increases from 5 to 11, precision, recall, and F1 scores also increase. Between 10 and 14 neighbors, the graph indicates stable scores for precision and F1, while recall continues to rise. Beyond a neighborhood size of 14, precision and F1 scores tend to decrease, although recall increases. This suggests that the recommendation system provides fewer successful recommendations but more courses. The highest precision, recall, and F1 scores indicate good recommendation quality. At k = 11, the graph shows the highest scores for precision (0.844), recall (0.895), and F1 (0.861), implying that the recommendation engine provides more accurate recommendations for the active user when the size of the neighborhood is 11. Consequently, the nearest neighbor user selected is 11 for subsequent parameter adjustments.

The table provides Table 3.1 values representing the performance metrics for different sizes of neighborhoods ranging from 8 to 15, aiding in the assessment of the system's precision, recall, and F1-score across varying neighborhood sizes.

Table 3.1 Performance comparisons for different sizes of neighborhood

Size of neighborhood (K)	Macro-average Precision	Macro-average Recall	Macro-average F1-score
8	0.8234	0.85	0.831
9	0.827	0.865	0.839
10	0.842	0.887	0.856
11	0.844	0.895	0.861
12	0.839	0.899	0.859
13	0.836	0.904	0.859
14	0.836	0.91	0.861
15	0.829	0.912	0.857

Experiment 2. Coverage metric for different size of neighbors

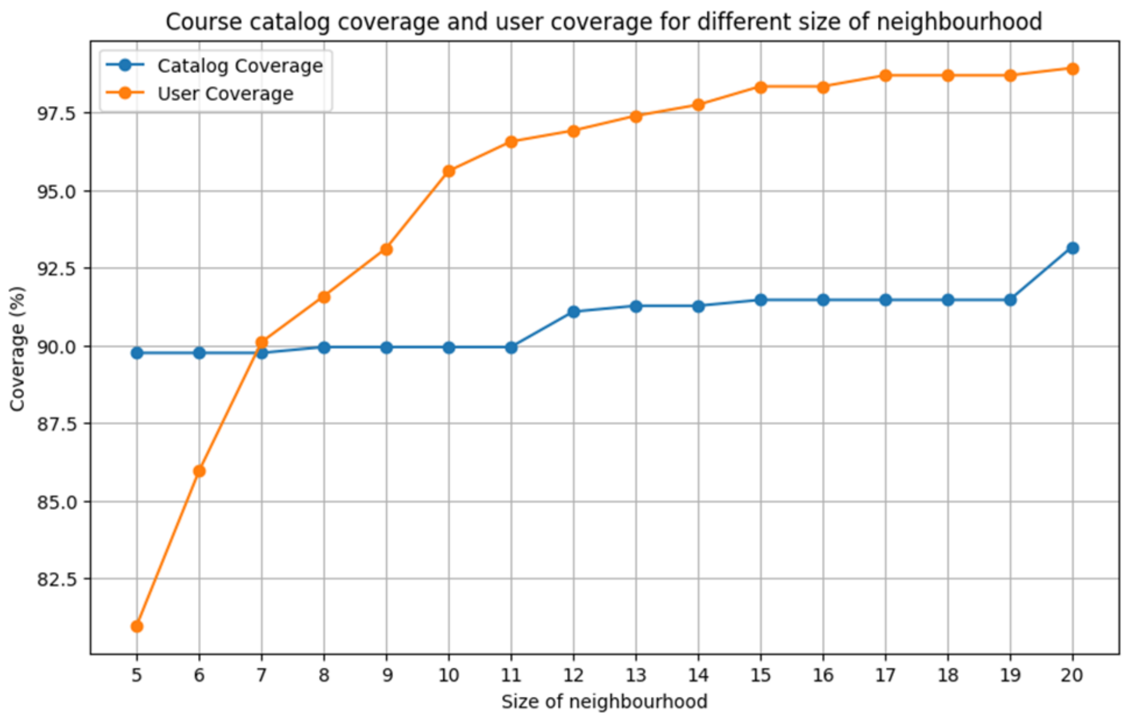


Figure 3.5 Coverage for different size of neighborhood

The Figure 3.5 illustrates the experimental results of how the size of neighborhoods (size of neighbors (k) values ranging from 5 to 20) affects two aspects: user coverage and catalog coverage. As the size of neighbors increases, both user coverage and catalog coverage also increase. This indicates that considering a larger number of neighbors helps cover a large number of users, recommending at least one course, and enables the system to provide a wider range of different courses available in the system.

However, when adding more neighbors, the catalog coverage doesn't improve significantly (increased by only 3%). On the other hand, the user coverage changes significantly from 84% to 98% when changing the size of the neighborhood from 5 to 20. This indicates that the system is able to provide recommendations for users with at least one recommendation when adding more neighbors. However, coverage doesn't provide information about the correctness of the recommendations. Instead, it indicates how many items are ever recommended to users out of the available items and how many users receive recommendations for at least one course.

The values presented in the Table 3.2 offer a comprehensive evaluation of coverage metrics across a spectrum of neighborhood sizes, providing valuable insights into how different neighborhood sizes affect coverage in the collaborative filtering system.

Table 3.2 Coverage metric comparison for different size of neighborhood

Size of neighborhood (K)	Catalog Coverage(%)	User Coverage(%)
8	89.94	91.58
9	89.94	93.12
10	89.94	95.61
11	89.94	96.56
12	91.08	96.92
13	91.27	97.39
14	91.27	97.75
15	91.46	98.34

Table 3.3 Course catalog coverage and user coverage (k = 11)

Course Catalog Coverage	89.94%
User Coverage	96.56%

When the neighborhood size is set to 11, the course catalog coverage of 89.94% indicates that nearly 90% of all unique courses available in the catalog are being recommended to users by the system. This suggests that the recommendation system effectively covers a wide range of available courses, ensuring users have access to a diverse set of options.

On the other hand, the user coverage of 96.56% signifies that almost 97% of all users received at least one recommended course for each semester. This implies that the system successfully provides recommendations to the majority of users across each semester.

3.2 Discussion

In this study, the proposed method presents a collaborative filtering recommendation system that utilizes the cosine similarity algorithm to recommend courses to students at the Faculty of Science, University of Peradeniya. By incorporating the preferences of students with similar interactions, our aim was to better predict courses for the upcoming semester in their university journey. Our experiments were conducted on the training dataset, which comprises 75% of the student course data obtained from the Science faculty LMS at the University of Peradeniya.

The experimental results demonstrate the impact of different sizes of neighborhoods (k) on the performance and coverage of the recommendation system. In Experiment 1, we observed that increasing the size of the neighborhood generally leads to improvements in macro-average precision, recall, and F1 score. Specifically, precision, recall, and F1 score increase as the number of neighbors grows from 5 to 11, indicating better recommendation quality. However, beyond a neighborhood size of 11, precision and F1 scores start to decline, although recall continues to rise. This suggests a trade-off between precision and recall as the neighborhood size increases, with the system recommending more courses with fewer successes, achieving an optimal balance at k = 11.

In Experiment 2, we evaluated the coverage metrics for different neighborhood sizes. The results show that both user coverage and catalog coverage increase as the size of neighbors increases. This indicates that considering a larger number of neighbors helps cover a larger proportion of users and a wider range of courses available in the system. However, while user coverage significantly improves with more neighbors, catalog coverage shows only marginal improvement. This suggests that the system cannot significantly improve course catalog coverage by further increasing the size of the neighborhood.

Overall, the findings suggest that a neighborhood size of 11 strikes a balance between recommendation quality and coverage. At this size, the recommendation system achieves high precision, recall, and F1 score, while also providing recommendations to a large proportion of users and covering a diverse set of courses.

CHAPTER 04

4.1 Conclusion

In the field of education, recommendation systems have been utilized to propose suitable courses to students based on their individual performance, personalities, and the ratings provided by other students. The main focus of the project was to incorporate data from students who had similar profiles in order to enhance the accuracy of course recommendations for students. To achieve this, a memory-based collaborative filtering method was employed to develop the course recommendation model. The performance of the model was then evaluated to assess its efficiency and effectiveness in providing course recommendations. By leveraging data from similar students, the course recommendation system is able to generate a tailored list of courses for students to consider during their course registration process each semester.

4.2 Future Works

- Integrating the recommendation engine with the existing Learning Management System (LMS) of the Faculty of Science at the University of Peradeniya. This integration will streamline access to our recommendation engine, providing students with personalized course suggestions directly within the LMS platform.

- Utilizing more advanced approaches, such as coupling collaborative filtering with deep learning neural networks, to tackle the cold start problem. This approach leverages the strengths of both techniques to provide more accurate and personalized recommendations, particularly for new users or new courses with limited historical data.

- To improve the recommendation system, incorporating students' grades to assess similarity is planned. However, the dataset lacks adequate grade information for this analysis. By incorporating grades alongside other data, more personalized recommendations can be offered based on academic performance and preferences. This enhancement would improve the accuracy and relevance of recommendations.

REFERENCES

Al-Badarenah, A., & Alsakran, J. (2016). An Automated Recommender System for Course Selection. *International Journal of Advanced Computer Science and Applications*, *7*. https://doi.org/10.14569/ijacsa.2016.070323

Estrela, D., Batista, S., Martinho, D., & Marreiros, G. (2017). A Recommendation System for Online Courses. *Advances in Intelligent Systems and Computing*, 195–204. https://doi.org/10.1007/978-3-319-56535-4_20

Fkih, F. (2021). Similarity measures for Collaborative Filtering-based Recommender Systems: Review and experimental comparison. *Journal of King Saud University -Computer and Information Sciences*. https://doi.org/10.1016/j.jksuci.2021.09.014

Ge, M., Delgado, C., & Jannach, D. (2010). Beyond accuracy: Evaluating recommender systems by coverage and serendipity. In *RecSys'10—Proceedings of the 4th ACM Conference on Recommender Systems* (p. 260). https://doi.org/10.1145/1864708.1864761

Herlocker, J., Konstan, J., Terveen, L., Lui, J. C. s, & Riedl, T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems, 22*, 5–53. https://doi.org/10.1145/963770.963772

Itmazi, J. A., & Megías, M. G. (2008). Using Recommendation Systems in Course Management Systems to recommend Learning Objects. *The International Arab Journal of Information Technology*, *5,* 234–240.

Jomsri, P. (2018). FUCL mining technique for book recommender system in library service. *Procedia Manufacturing, 22,* 550–557. https://doi.org/10.1016/j.promfg.2018.03.081

Lahoud, C., Moussa, S., Obeid, C., Khoury, H. E., & Champin, P.-A. (2022). A comparative analysis of different recommender systems for university major and career domain guidance. *Education and Information Technologies*. https://doi.org/10.1007/s10639-022-11541-3

Panteli, A., & Boutsinas, B. (2023). Addressing the Cold-Start Problem in Recommender Systems Based on Frequent Patterns. *Algorithms*, *16*, 182. https://doi.org/10.3390/a16040182

Rahman, M. M., Islam, M. S., Richi, R. R., & Chakraborty, A. (2022). Course Recommendation System for Students Using K-Means and Association Rule Mining. *2022 International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*. https://doi.org/10.1109/ismsit56059.2022.9932747

Ray, S., & Sharma, A. (2013). A Collaborative Filtering Based Approach for Recommending Elective Courses. *arXiv (Cornell University)*.

Roy, D., & Dutta, M. (2022). A systematic review and research perspective on recommender systems. *Journal of Big Data*, *9*. https://doi.org/10.1186/s40537-022-00592-5

Sarwar, B., Karypis, G., Konstan, J., & Reidl, J. (2001). Item-based collaborative filtering recommendation algorithms. *Proceedings of the Tenth International Conference on World Wide Web - WWW '01*. https://doi.org/10.1145/371920.372071

Vizine Pereira, A. L., & Hruschka, E. R. (2015). Simultaneous co-clustering and learning to address the cold start problem in recommender systems. *Knowledge-Based Systems*, *82*, 11–19. https://doi.org/10.1016/j.knosys.2015.02.016

Wei, J., He, J., Chen, K., Zhou, Y., & Tang, Z. (2017). Collaborative filtering and deep learning based recommendation system for cold start items. *Expert Systems with Applications*, *69*, 29–39. https://doi.org/10.1016/j.eswa.2016.09.040

Zhi-Dan Zhao, & Ming-Sheng Shang. (2010). User-Based Collaborative-Filtering Recommendation Algorithms on Hadoop. *2010 Third International Conference* *on Knowledge Discovery and Data Mining*. https://doi.org/10.1109/wkdd.2010.54