CHAPTER 1

INTRODUCTION

The growing proliferation of online courses provided by numerous universities across the globe makes it a challenging process for selecting a major and a university. A myriad range of courses is offered by individual universities whose delivery mode and entry requirements all differ. Such abundant information means that students need to search, organize and use the resources that can enable them to match their individual goals, interests and current level of knowledge. This can be a time-consuming process as it involves accessing each platform, searching for available courses, carefully reading every course syllabus and then choosing the one that is most appropriate for the student.

Furthermore, studies have shown that, the student's choices are influenced by their background, personal interests and career interests. Researchers have found that three out of every four students were uncertain or tentative about their career choice at the time of college entry. The process of choosing a course can be incredibly tedious and extremely complicated. However, simply because more course information is now provided on university websites, this does not automatically mean that students possess the cognitive ability to evaluate them all. Instead, they are confronted with a problem that is termed "information overloading".

So a personalized recommendation system can be an effective way of suggesting the relevant courses to the prospective students. Artificial intelligence methods developed at the beginning of research are now being applied to information retrieval systems. Recommended systems provide a promising approach to information filtering as they help users to find the most appropriate items. Based on the needs of each user recommendation system, a series of special suggestions will be generated.

1.1 Issues

Despite the high impact of the course recommendation system and how useful it is, there are certain significant limitations, such as:

- Models based mainly on the keywords failed to address the individual user's needs in the recommendation process.
- Although models use collaborative filtering, and data mining such as association rule and decision tree, there is often a lack of historical information that makes it challenging to adopt this approach. For instance, new students who wish to use the systems do not have sufficient information about the model and therefore cannot generate any recommendations.

- The shortcoming of models that use content-based filtering is that current approaches are based only on a specific subject recommendation rather than an entire university course. Moreover, the similarity calculation in these models is based on the weighted average of features and does not take into account user interaction with the system, such as the rating value of recommendation items.
- Another shortcoming of the current models is that they do not provide comprehensive knowledge about the course that is most relevant to the student. For example, students need to know what future career the course will lead to and require information about this aspect, as well as the quality of the facilities of the educational institution itself that will be providing the course.

1.2 Problem Definition

The shortcomings listed above are overcome by the proposed system "Filtering based personalised course recommendation system "using machine learning algorithms. Through categorizing the needs of students and their areas of interest, it is possible to recommend an appropriate course. It is possible to help students to select a course by developing methods that will both integrate the data from multiple heterogeneous data sources and allow this to rapidly set valuable course-related information. By using this ontology, the user will be able to gain precise knowledge about the course. We have been able to build a relationship between the relevant information available through the internet, including the course modules, job opportunities and the users' interests. Ontology provides a vocabulary of classes and properties that can be used to both describe a domain and emphasize knowledge sharing. The use of semantic descriptions of the courses and the students' profiles allows there to be both qualitative and quantitative reasoning regarding the matching, as well as the required information about the courses and the student's interests which is necessary in order to refine the process of deciding which course to select. A novel hybrid filtering is proposed in this study, based on both the CBF and CF methods and using ontology as a way by which to overcome the problem of information overloading which has been a key challenge when consideration is given to building an effective recommendation system. This problem is related to the scarcity of information that is available (i.e. for users and items) in the recommendation filtering algorithms. The proposed approach uses ontology for data extraction and integration from multiple data sources. Data integration that is based on ontology is used in the ontology-based metadata. It utilizes a combination of model-based and memory-based use of ontology in CF to provide a high-quality recommendation. User

profiling that is based on ontology, item ontology, the semantic similarity between two ontologies and the proposed OKNN algorithm is used in the CF to overcome the new user problem. On the other hand, item-based ontology and semantic similarity are both applied in CBF to overcome the new item cold start problem. In order to ensure the measurement of semantic similarity is more accurate, a heuristic method is used in the CBF. This measures the ``IS-A" degree between the two nodes of item ontology, which was found to yield a more precise recommendation list for the target user.

1.3 Scope

In future, we will enrich our repository by absorbing more course and user information and heterogeneous data sources. Also, we plan to incorporate additional user contexts, e.g., available student behavior, learning style and learning interests into the recommendation process in order to make the system more comprehensive and intelligent. We plan to carry out more experiments with a variety of actual students from different departments and from various academic backgrounds for more flexibility.

CHAPTER 2

LITERATURE SURVEY

Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy and company strength. Once this things are satisfied, then next step are to determine which operating system and programming language can be used for developing the tool. Once the programmer starts building the tools the programmer need lot of external support. This support can be obtained from senior programmer, from books and from website, before building the system the above consideration are taken into account for developing the proposed system.

Some efforts have been reported in the literature on the development of the "Ontology based course recommendation system".

- W.Yang ,Z.Wang and M.You are the authors who introduced collaborative filtering method .In this paper the application of user-based algorithms are used. Such algorithms use two key process-similarly computing and prediction. Also, it is a model-based approach where Bayesian and cluster models are used. The advantage of this system is to improve the accuracy of recommendations by formulating a better recommendation system. The disadvantage of this system is that the suggested terms will not be relevant to the preferences of the target user.
- B.Sarwar,G.Karypis are the authors who introduced "Item-Based Collaborative filtering recommendation algorithms". In this paper the author has analyze different item-based recommendation generation algorithms. The advantage of this system is that, the item-item scheme provides quality of predictions than the user-user (K-nearest neighbor) scheme. The disadvantage of this system is that it has lack of historic knowledge about students.
- V. Ghai and Y. Gao are the authors who introduced this "Course Recommendation System using multiple criteria decision making method". In this system CSS concepts are used.MLP (Multilayer Perceptron) and CNN (Convolution Neural Network) algorithms are used in this system. The advantage of this system is that it provides suitable inductive biases catered to the input data type. The disadvantage of this system is that it provides explainable predictions seems to be really challenging task.

- The authors Rosta Farzan and Peter Bruisilovsky introduced this "Community-Based recommendation system that employs social navigation to tackle the problem of information overload. The advantage of this system is that it minimize the risk of information overloading. The disadvantage of this system is that it cannot access to each and everyone.
- J.Cuseo is the author who proposed the system called "Academic Advisement and Student Retention". In this system the author has decided to recommend the course for the higher education which is mainly helpful for students and teachers. The author also provides effective educational and career planning decision making recommendations. The advantage of this system is that it improves the higher education course recommendation system. The disadvantage of this system is that it need to focus on recruitment and selection of advisor and preparation and development of advisor.
- Rui Ren, Lingling Zhang, Limeng Cui, Bo Deng ,Yong Shi Proposed the algorithm that is the " Personalized Financial News Recommendation Algorithm" this algorithm deal with the challenge of information overloading ,it helps user to find the articles that are interesting to read. To settle the ambiguity problem, a new presented OF-IDF method is employed to represent the unstructured text data in the form of key concepts, synonyms and sets which are all stored in the domain ontology. For users, the recommendation algorithm build the profiles based on their behaviors to detect the genuine interests and predict current interests automatically and in real time by applying the thinking of relevance feedback ,which aims to deliver the interesting news articles to users not only to help individuals save time and energy but also help enterprises improve user loyalty. This algorithm has the advantage to deliver the news as per the interest of user and has the disadvantage in lack of prediction.
- Hazra Imran, Mohammad Belghis-Zadeh, Ting-Wen Chang, Kinshuk, Sabine Graf proposed the algorithm that is "a personalized learning object recommender system". This algorithm helps in the innovation of information & communication technologies plays supports learners by providing them recommendations about which learning objects within the course are more useful for them, considering the learning object they are visiting as well as the learning objects visited by other learners with similar profiles. This kind of personalization can help in improving the overall quality of learning by providing recommendations of learning objects that are

useful but were overlooked or intentionally skipped by learners. The system increases learners' performance and satisfaction during the course. It has the advantage of providing learning objects within a course .the disadvantage in this algorithm is that it Only focuses on academic related topic.

- H. Zhang and H. Yang has proposed an algorithm that is "personalized course recommendation system based on DBN in MOOC environment" This paper aims to predict the user dropout rate in MOOC learning based on the features extracted from user learning behaviors some learning behavior features were extracted from the data of MOOC platforms. The two machine learning algorithms, respectively based on support vector machine (SVM) and the artificial neural network (ANN), were introduced to predict the dropout rate of MOOC course. The two algorithms were contrasted with some commonly used prediction methods. Typical features were extracted from the click stream file and forum posts data in a MOOC platform, and were subject to post processing. The post processing could greatly improve the prediction accuracy of the machine learning algorithms, and enable the instructor to take measures against dropout from the course. This paper aims to enhance the prediction performance of the predictors, and create faster and more cost-effective predictors the author seek to improve the prediction accuracy in the initial period of the course.
- CHUNG-YI HUANG, RUNG-CHING CHEN LONG-SHENG CHEN has proposed the algorithm that is "COURSE-RECOMMENDATION SYSTEM BASED ON ONTOLOGY" To train students to learn a skill and increase the competitiveness for jobs, the tertiary institutions design special curriculum program to have a goal as learning to meet the demand of professional skills for students. This study used ontology technology to implement the Course- recommendation System, to provide students adaptive learning recommendations, and let students to learn a complete knowledge required to enter the workplace in the future. The curriculum program is designed as a package for a professional field, for this viewpoint and for the workplace. So the curriculum program data is built by adding the concept of job titles which provides reference to the students. The disadvantage in this system is that it lacks in combining the course scores and the student interest to improve the satisfaction of the useful system.

- Rel Guzman Apaza, Elizabeth Vera Cervantes, Laura Cruz Quispe, Jose Ochoa Luna has proposed the algorithm that is “Online Courses Recommendation based on LDA”. This system proposes a course recommendation system based on historical grades of students in college. The goal of our proposal is to recommend courses for students who have received low grades in college therefore, we are using grades as ratings. Probabilistic topic models are used as follows. On one hand, Latent Dirichlet Allocation (LDA) topic model infers topics from content given in a college course syllabus. On the other hand, topics are also extracted from a massive online open course (MOOC) syllabus. These two sets of topics and grading information are matched using a content based recommendation system so as to recommend relevant online courses to students. The disadvantage of this algorithm is that it has scalability issues.

CHAPTER 3

PROBLEM STATEMENT

3.1 Existing System

A significant number of recommender systems have been proposed in the education domain, as well as in teaching and academic advising. In the education domain, the target users are students, teachers or academic advisors, and the recommendable items are educational materials, universities or information, such as courses, topics, student performance and the field of study.

Sandvug and Burke presented Academic Advisor Course Recommendation Engine (AACORN) that used a case-based reasoning approach, which utilized knowledge that had been acquired from previous cases in order to solve new problems. Their system used both the course histories and experience of past students as the basis of assisting students in course decision making. At the same time, it was noticed that the future career of students is an essential factor which can influence their decision to choose a particular course.

Farzan and Brusilovsky proved this by using a reported course recommendation system that was based on an adaptive community. They employed a social navigation approach to analyses the students' assessment of their career goal in order to provide recommendations for courses.

The primary idea of this approach was to obtain the students' explicit feedback implicitly, as part of their natural interaction with the system. In this respect, Artificial Intelligence techniques could develop and improve the decision making and reasoning process of humans to minimize the amount of uncertainty there is in active learning to ensure a lifelong learning mechanism. The challenge for recommender systems, therefore, is to better understand the student's interest and the purpose of the domain. An association mining based recommender has been developed for recommending tasks that are related to learning, and are most suitable for learners based on the performance of the targeted student and other students who are similar to them.

3.2 Drawbacks of Existing System

The drawbacks of the existing system are as follows:

- The model fails to integrate data from a large number of heterogeneous resources because of information overloading.
- Another snag of the current models is that they do not provide comprehensive knowledge about the course that is most relevant to the student.
- The biggest hindrance is the cold start problem, which occurs when user information from the past is insufficient.
- Another shortcoming is the initial lack of ratings for the new users and hence it becomes impossible to make reliable recommendations.
- Consequently, this also affects the accuracy of the recommendation proposed.

3.3 Proposed System

A hybrid recommender method based on ontology has been proposed.

- The method firstly aims to extract and integrate information from multiple sources based ontology. The information sources are classified into three primary sources; course information sources, student information sources and career information sources. Integrating information using ontology will obtain an optimal result.
- Moreover, the second objective is to build dynamic ontology mapping between the user profiles and the item profiles that will help to reduce information overloading.
- In order to recommend an appropriate recommendation to the users, we have combined two main filtering approaches, CBF and CF, and thus the result is a combination of memory-based and model-based methods. In the CF, several techniques, such as user profiling that is based on ontology, item ontology and k-NN, are used to overcome the information overload problem and improve scalability and accuracy.
- On the other hand, item-based ontology and semantic similarity are applied in the content-based filtering to solve the new user issue and to also improve accuracy.
- The final objective is to put forward a list of recommendations and ask the user to assign a rating to each recommendation. The user then gives their feedback on the recommendation list and carries out a re-ranking. User feedback has been used to

evaluate the system and improve its accuracy, as is shown in greater detail in the evaluation section.

This work aims to increase the accuracy and performance of the recommender system by combining the hybrid method (CBF and CF) with enhanced ontology.

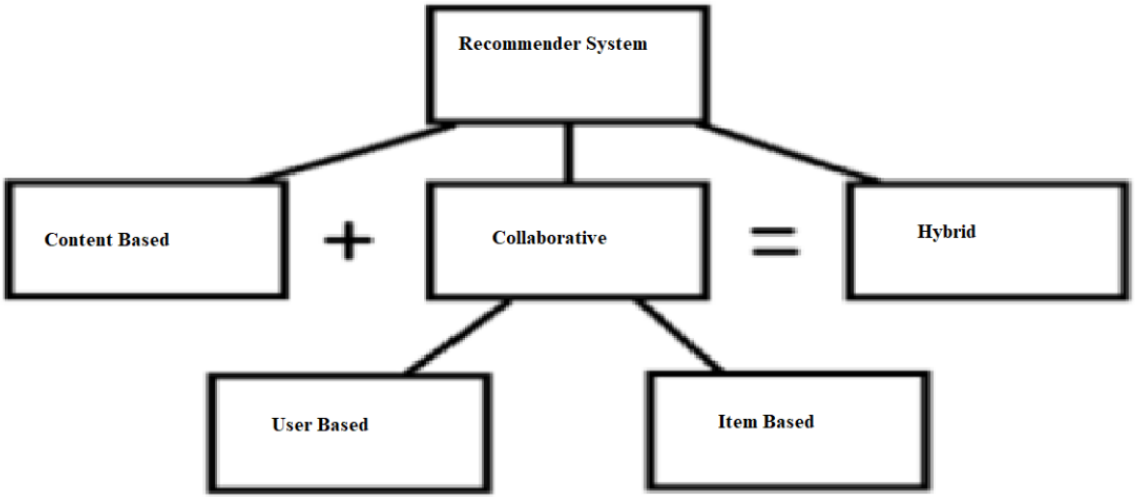
3.4 Objectives of Proposed System

The main objectives of the proposed system are:

- Develop a comprehensive framework that combines CBF and CF with an ontology technique in order to overcome the overloading information problem. This is achieved by using a similar hierarchal ontology to map the profiles of the courses with the user profile.
- Develop a new approach to extract and integrate data from multiple sources and then map them. This ontology mapping of the different data improves the ability to obtain a comprehensive knowledge of the recommended items.
- The approach tackles the new user problem by calculating the ontology similarity there is between the users' profiles by measuring the user rates for each item. The proposed recommender system is used to work out the hierarchy ontology similarity there is between the item profiles and the users' profiles before the student enrolls on the research program and chooses courses to match his/her requirement.

3.5 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. Such outputs or findings can then be used in such a way as to assist the user in making decisions.



3.6 Content-Based System

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.

In the context of course selection systems implemented in universities and 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 in 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 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.

3.7 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. 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. 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. 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.

3.8 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. 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. An effective method when concatenating results would be to set different weights for each set of results based on the current environment of data. 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.

CHAPTER 4

SYSTEM REQUIREMENTS

4.1 Hardware Requirements

- Processor: i3
- RAM: 512MB
- Hard Disk: 30GB (approx.)
- Display: VGA Color Monitor

4.2 Software Requirements

- Operating System: Ubuntu 18.04 LTS
- Language: Python
- Framework: HTML, CSS, JavaScript, MySQL

CHAPTER 5

SYSTEM ARCHITECTURE

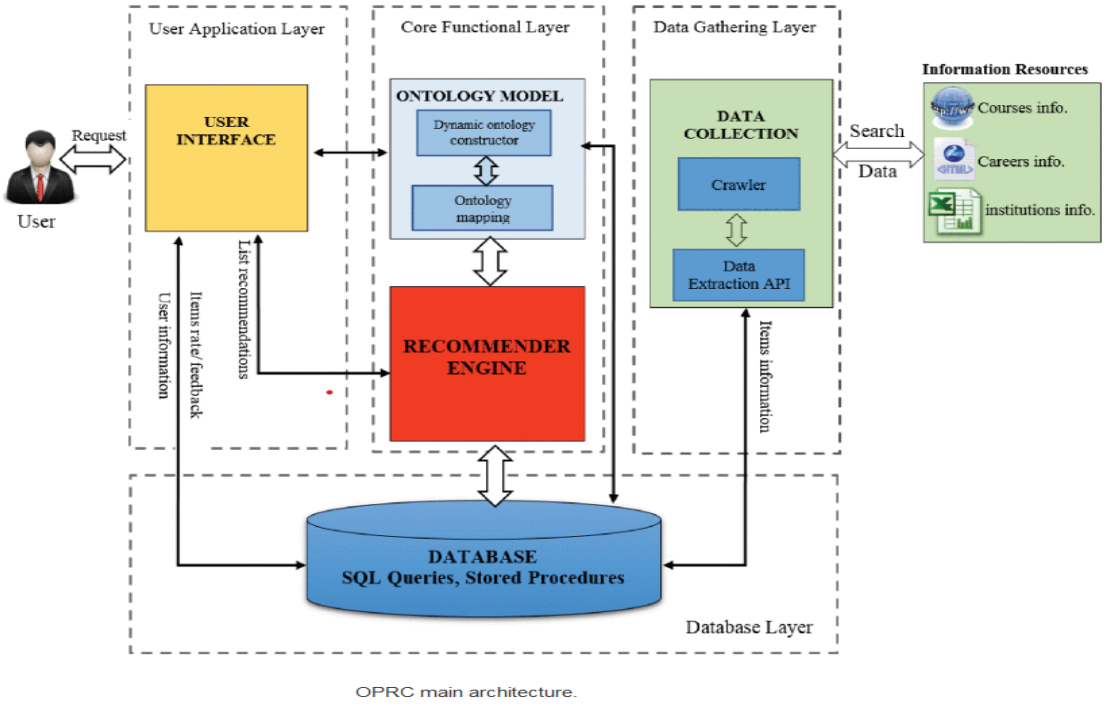


Fig. 5.1: System Architecture

The proposed ontology-based personalized course recommendation framework (OPCR) is focused on recommending courses to students by utilizing a hybrid filtering approach that combines both content-based filtering and collaborative based filtering with ontology support.

Our framework comprises the following steps:

(1) Extract all the useful information for the system from multiple sources.
(2) Build the courses' profiles by extracting all the useful information regarding course features and sorting that information in the system database. Consideration is given to the ontology hierarchy of the course features.
(3) Build the student profile by obtaining student information via both explicit and implicit approaches. We have identified different user attributes which can be used to profile the student into our system as well as the user ratings of the recommended courses.
(4) Build dynamic ontology mapping in order to link the user profile and item profile.

(5) Analyze user queries and calculate the similarity between the user profile and the course profile by employing ontology matching and cosine similarity.
(6) Use a collaborative filtering technique in order to obtain top N users that are similar to the current user by using an ontology-based k nearest neighbor (OKNN) algorithm.
(7)The final step suggests the recommended list of courses to the user and obtains user feedback.

The purpose of each of these components is explained in the following sections:

5.1 Data Gathering

As it was decided that a content-based recommender system technique should be the primary approach for the provision of recommendations, there are different formats of information that need to be gathered to support this system. Fortunately, all of these are available through information sources which are publicly available, either through websites in HTML format, such as the universities' websites for course information and recruitment websites for career information, or Microsoft Excel documents that have been uploaded to the internet, such as statistical information regarding the reputation of educational institutions, for example the NSS score for universities. The data from both the student and course ontology is prepared and pre-processed into the correct format for the recommendation engine by the pre-processing data component. It was a time-consuming task to obtain information about each course from all the universities' websites as each university publishes its course information in different formats. Extracting precise information from various websites is always a challenging task in the domain of information engineering so we customized a web crawler that browses the web page automatically. It scrapes information from a web page and then sorts this into the system database. The reformulated queries are allocated to web crawlers and APIs that search for specific course information and jobs. The web crawler analyses the web page based on a definition of the features of each course, and then extracts feature values. Each extracted feature value belongs to one of the features that we have used in this paper. Five features of the courses are marked in this study: course title, course major subject, course fee, university location and the language of the course. On the other hand, the feature that has been constructed in the user ontology is based on the feature in item ontology. The implicit information, such as the user, feedback and the rates of the recommendations, have been collected and added to the user profile for later use, when it is then utilized to locate a top-rated neighbor that is similar to the target user.

5.2 Database

A database management system (DBMS) is system software for creating and managing databases. The DBMS provides users and programmers with a systematic way to create, retrieve, update and manage data. A DBMS makes it possible for end users to create, read, update and delete data in a database. The DBMS essentially serves as the interface between the database and the user application layer. The DBMS manages three important things. The data, the database engine that allows data to be accessed, locked and modified, lastly the database schema, which defines the database's logical structure.

CHAPTER 6

DESIGN

6.1 Modules

Here, we have divided the operations into three modules which are as follows.

1. The first module is the ontology models, which includes construction of dynamic ontologies for the user and the items that map these ontologies in order to gain a comprehensive knowledge of the recommendations. It also contains course ontology model, job ontology model, student ontology model
2. The second module is the recommender engine model which takes as input the ontology models and gives the recommendation.
3. The final module is a graphical user interface that consists of the user interface model, which is responsible for user interaction with the framework, for searching items and for giving feedback on the recommendation list. 4

Every layer and model in the framework both links and interacts with the others, based on the input and output of each one.

6.1.1 Ontological Models

This section is the most important part of the framework. The ontology model, which includes construction of dynamic ontologies for the user and the items that map these ontologies in order to gain a comprehensive knowledge of the recommendations. After building the ontologies and mapping them, this will be used as an input in the recommender engine.

The present work has constructed three ontologies. Firstly, the course ontology; secondly the student ontology; and thirdly, the job ontology. The protégé tool has been used to evaluate the ontologies with hierarchical mapping between the ontology classes that are used to compute the similarity between them. Knowledge, represented by the ontologies, has been combined into one single ontology. The ontology model created significantly helps to reduce information overloading.

6.1.1.1 Dynamic Ontology Construction

A dynamic ontology-based model is proposed to classify the extracted terms and to build a knowledge base for a specific domain. It is a challenge to obtain a well classified corpus. Even if a corpus is available, it may be classified improperly due to fewer terms being

classified because of the limited and static nature of the classifiers. To overcome this, we propose using an ontology-based model in order to classify the terms and prepare the knowledge base. Ontology is a data model that characterizes knowledge about a set of classes or concepts and the relationships between them.

The classes define the types of attributes or properties that are common to individual objects within the class. The following modules explain our proposed dynamic ontology model: Document Analysis, Ontology Construction

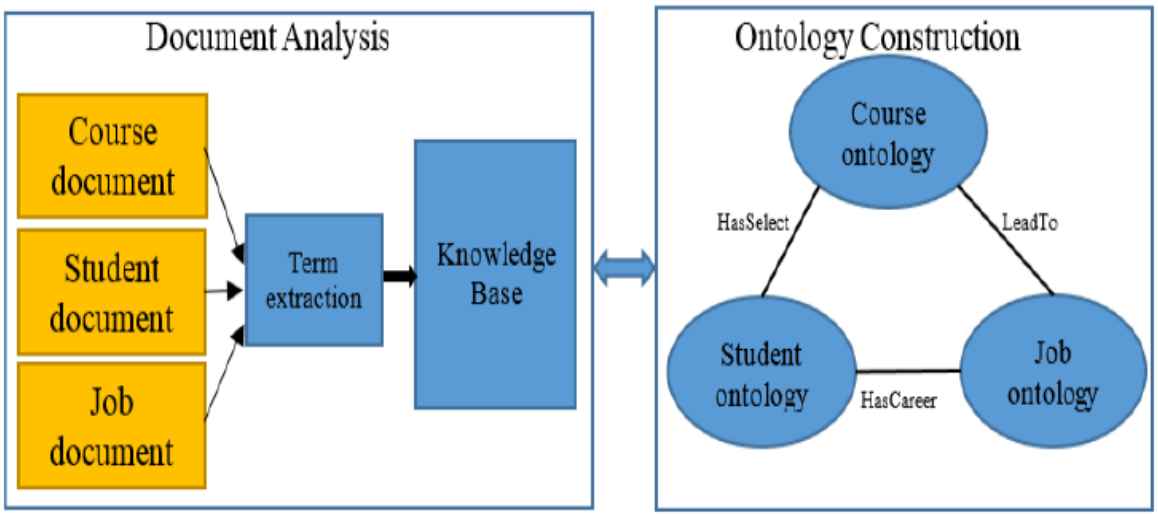


Fig. 5.2: Dynamic Ontology Construction

In order to construct this ontology, the following steps have been considered:

1. Determine the domain and scope of the ontology

In this proposed work, higher education has been determined as the domain and master's courses in Computing and Business Management have been determined as the scope of the ontology.

2. Take into account reusing existing ontology

In education, many ontologies were found that model this aspect of the domain. However, no ontology was found that could be reused to serve our intended purpose. Despite this, current ontologies have been used as a guideline to model the common concepts of the new ontology.

3. Enumerate the domain terms

The ontology is defined as a taxonomy that helps to describe different aspects of the domain, such as the student, course and career. Some concepts are further divided into subclasses that would improve the classification of the instances of these classes.

4. Determine the classes and the class hierarchy

The classes are defined as a group of individuals or instances that represent a class where all of the members share the same concepts. When the classes are ordered hierarchically, this is termed taxonomy. Inference engines use hierarchies to denote inheritance relationships. Classes are defined by following the combination development process, which is a combination of both bottom-to-top and top-to-bottom approaches. When this approach is followed, the important terms are first defined and then generalization and specialization takes place.

5. Define the relationships between classes

The relationship that exists between class members in ontology is termed the properties. There are two types of properties: object and data properties. Object properties represent the binary relations that exist between members of the classes, such as the relationship between a student and the courses. Here, we define a property called *HasSelected*, which is used to represent this relationship. Data properties link an individual to a data literal, such a student's ID.

6.1.1.2 Course Ontology Model

Identifying different attributes is necessary for course profiling. In order to construct course ontology, we need to identify factors that most influence a student when they make a decision in choosing a university course. These factors then become the main classes of the ontology.

A course program’s title, fees, location and prominence where all factors that appeared to be the most important when the students determined their choice of university for higher education study.

Among the elements included in the program factors, both the field of study and the details regarding course information appear to exert the most considerable influence on the students’ choice of university course program. The factor that was uppermost in the students' decision making frameworks was the issue of fees, which had the greatest impact on university choice and the type of career that could be achieved following completion of the course. It was found that issues of institutional prominence maintain a fairly high profile in students' decision-making. The overall reputation of the institution and the National Student Survey score (NSS) of teaching students are both significant. The course attributes are considered when extracting the course profile, including the essential information, course information, as well as information regarding fees and university rankings and the university's NSS score.

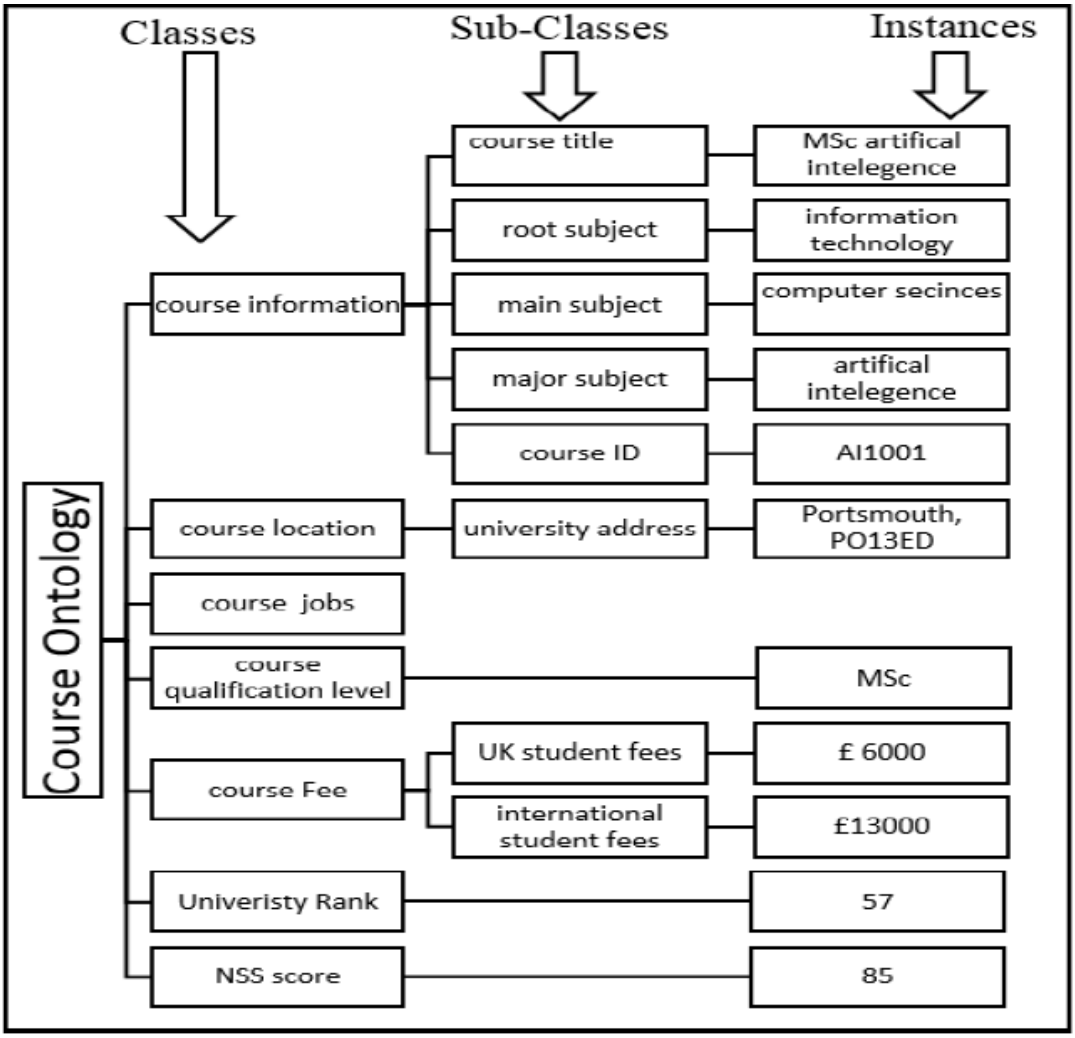


Fig.5.3: Course Ontology Structure

The course profile attributes will match the user profile feature through the ontology mapping. Each class of the course profile will be a map to the equivalent class in the user profile. Ontology reference is used to identify the equivalent classes in both the course profile and the user profile. The protégé tool was used for the construction and evaluation of the ontology model.

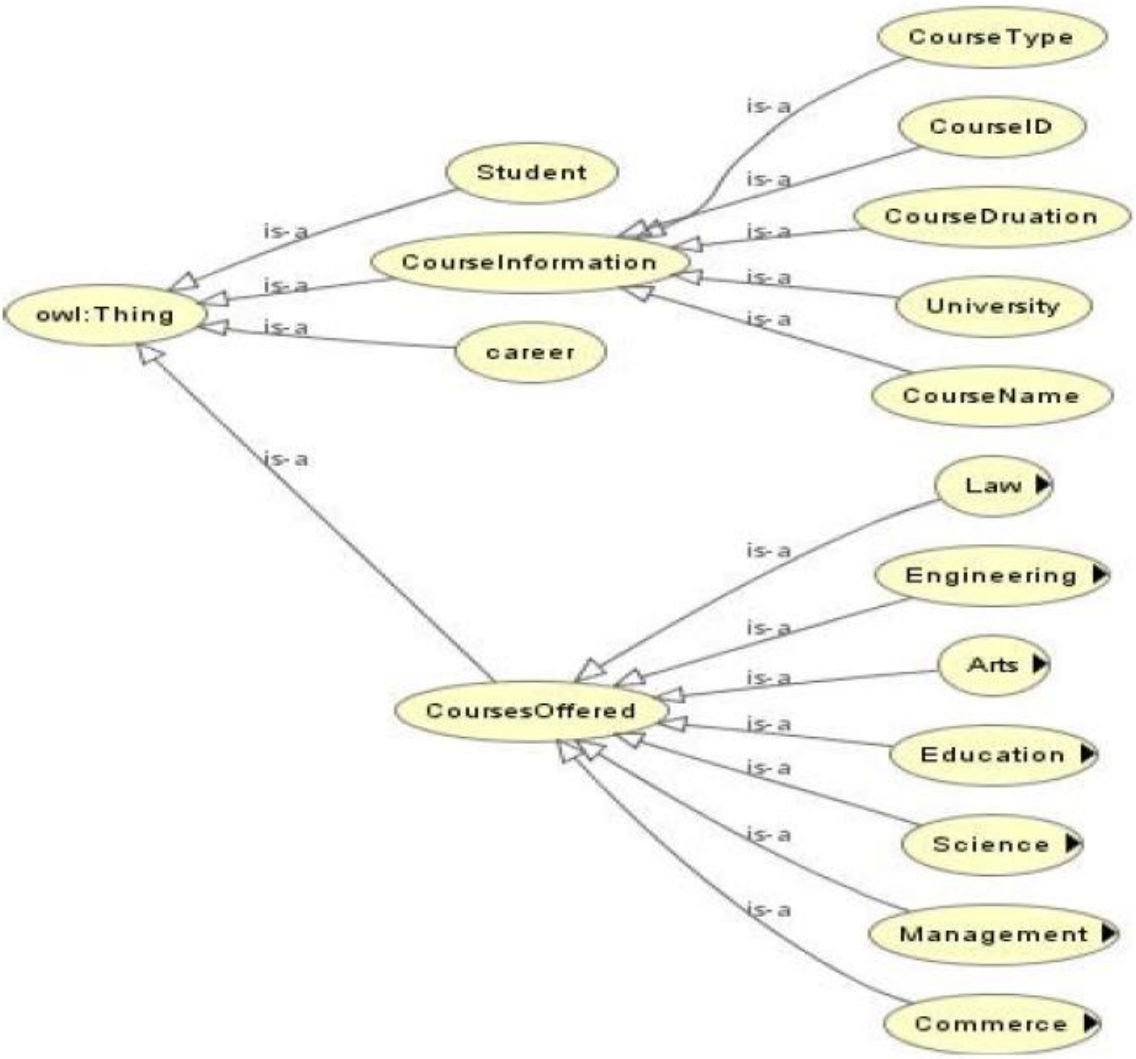


Fig. 5.4 Use Case Diagram of the Course Ontology

6.1.1.3 Student Ontology

Firstly, we need to model the student profile before recommending the appropriate course. The user profile consists of two main parts. The first part is the personal attributes and education attributes of the user and the second is the user's rating of the previously recommended course. The personal attributes include the user's individual personal information, as well as education and background information, such as their hometown, gender, the field of study, main subject, major subject, interest area, technical and non-technical skills.

Therefore, in this paper, a student profile can be formally defined as Formula (1) and

Formula (2):

U = {a1,a2,……an}

where U is the user/student, ai represents the users and *i*th attributes.

If a student has obtained an offer from the system in the past and rated the courses, we can further define that student as:

Ur={u,r}={a1,a2,……..an,r}

Here, Ur is the user that received a recommendation for the courses from the system and has rated the courses. Furthermore, in order to make a satisfactory recommendation, it is important to ensure that the characteristics of the recommended activities match the user's interests. The course ontology is created for all the courses that are to be recommended to the user/student. The system recommends several courses in the faculties of arts, information technology, science, social science, management, commerce, engineering, education and law. The student obtains a recommendation for any course depending on their eligibility, i.e. if the student has a graduation degree, the system can recommend any postgraduate course and if the student has a postgraduate degree, either a research course or a PhD can be selected, depending on the faculty.

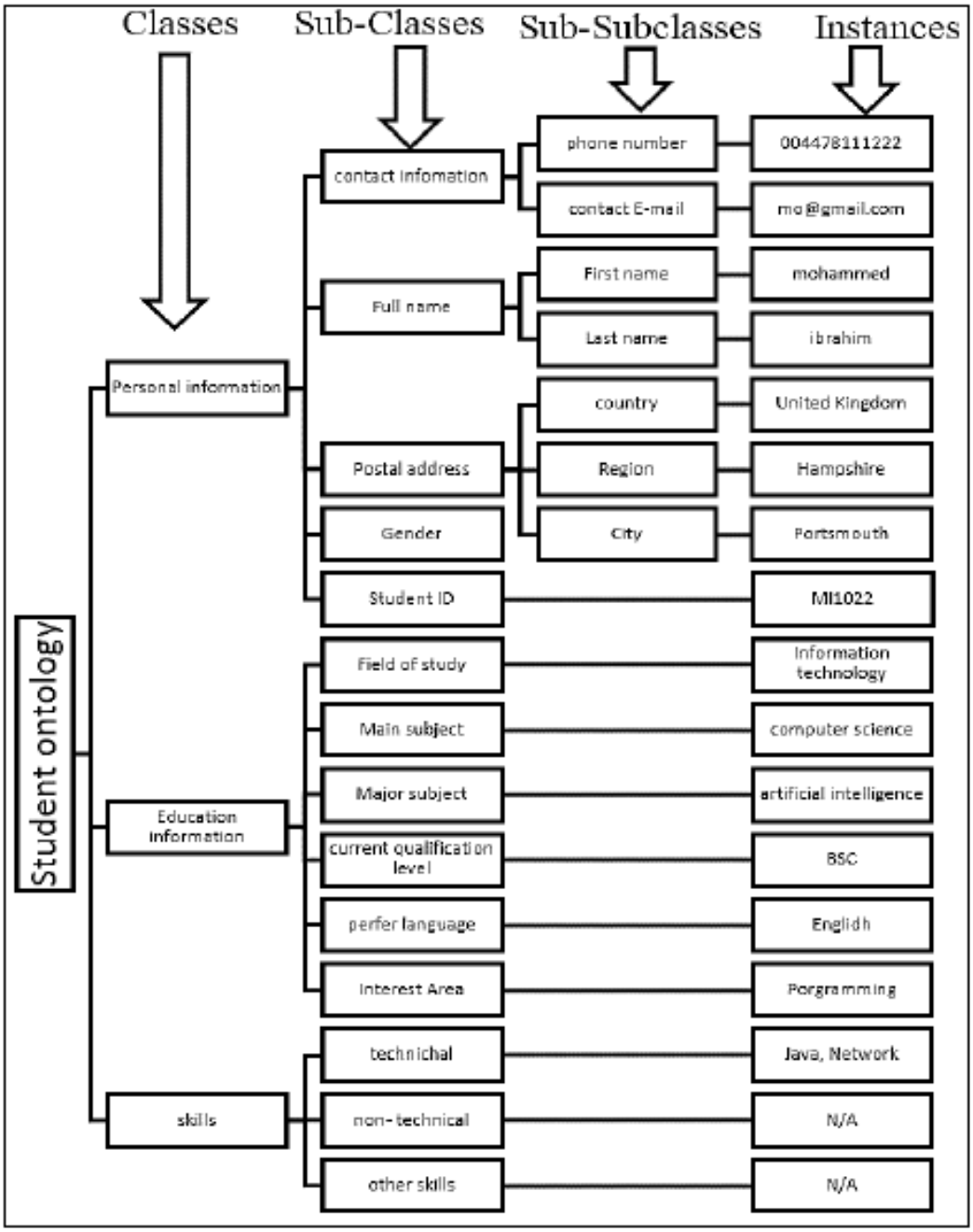


Fig. 5.5: Student Ontology Structure

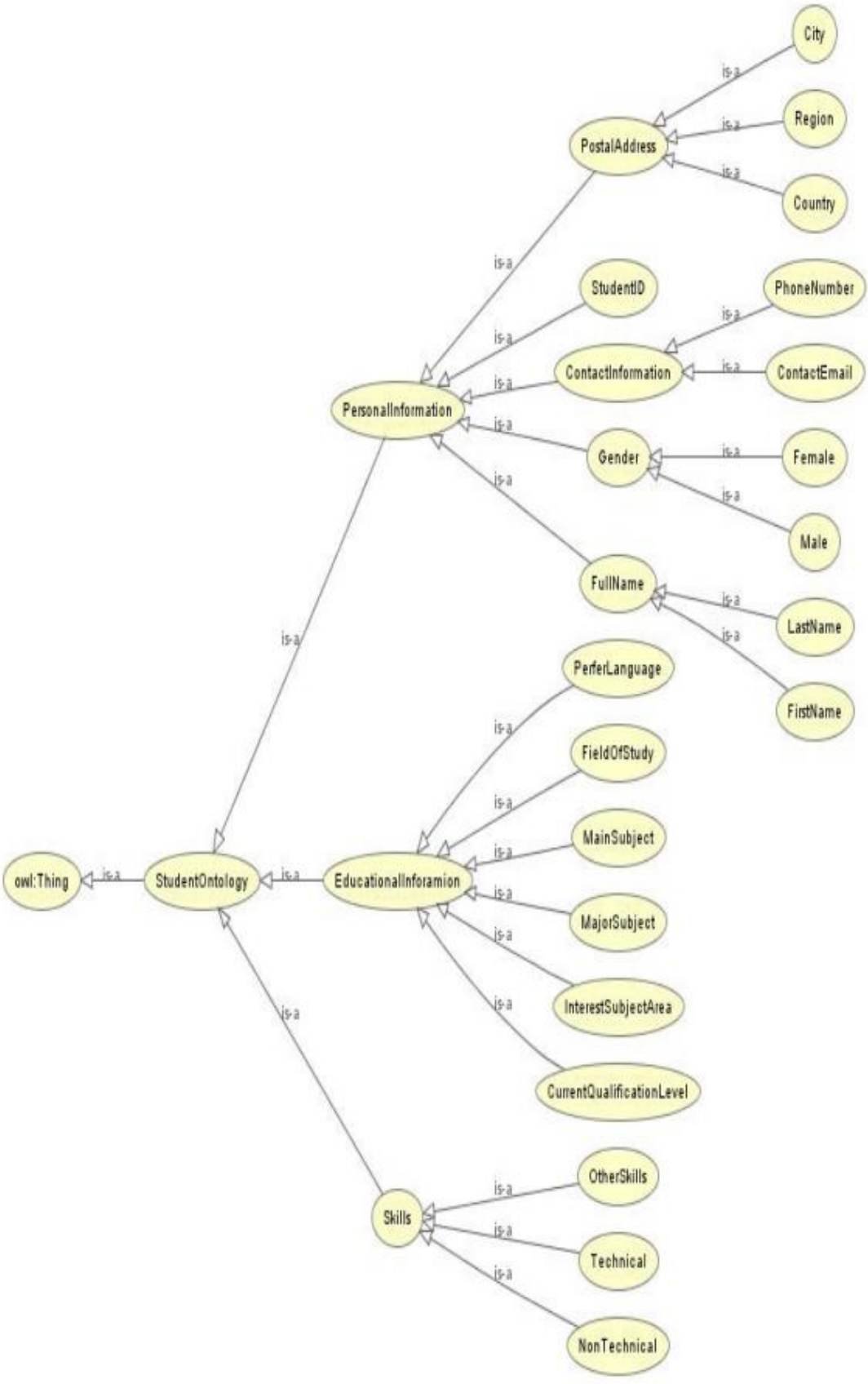


Fig. 5.6: Use Case Diagram of Student Ontology

6.1.1.4 Job Ontology

A student's future career is an essential factor that can influence their decision making when they are selecting a university course. Constructing job ontology is vital if a student is to understand the attributes of the job. This is extracted from a recruitment website. Job attributes include information such as job title, job description, job salary, job location and the required education qualification.

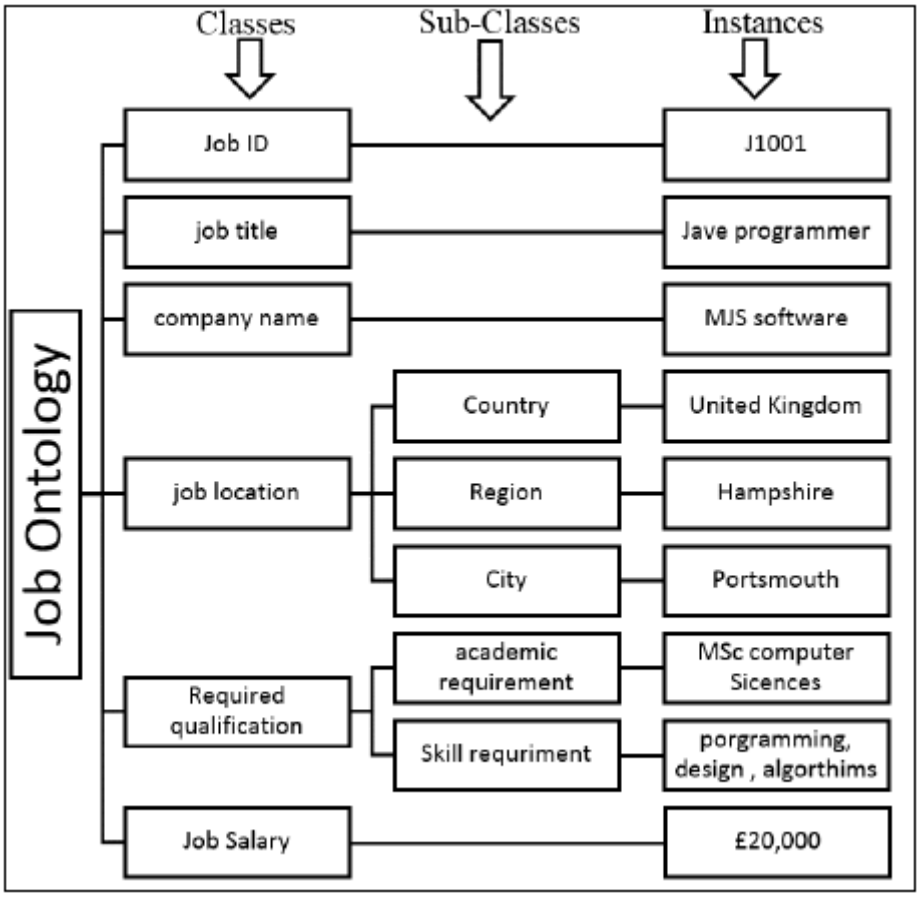


Fig. 5.7: Job Ontology Structure

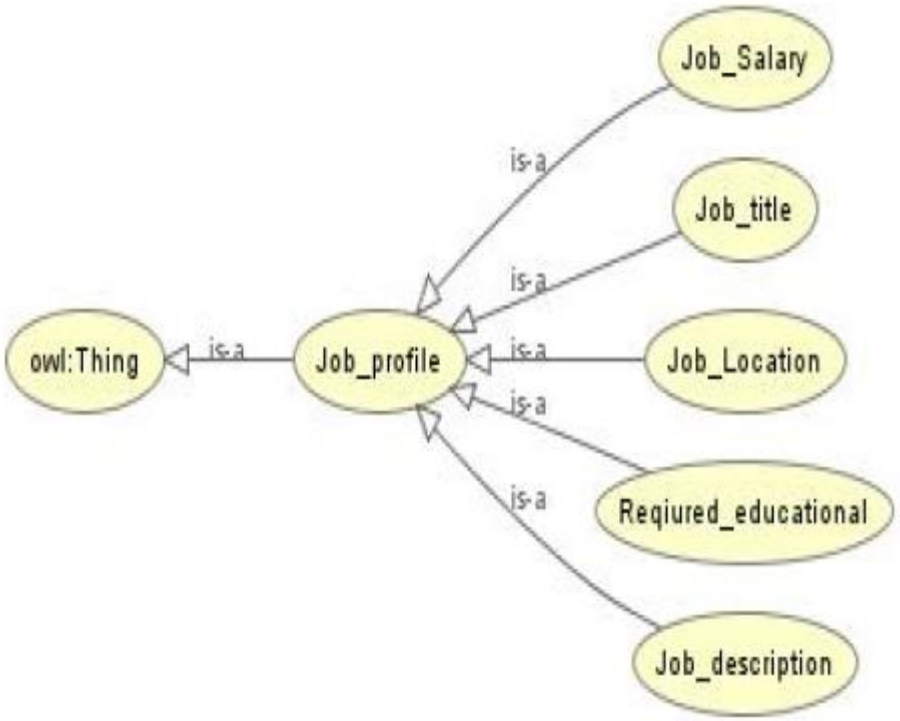


Fig. 5.8: Use Case Diagram of the Job Ontology

6.2 Recommender Engine

We used a hybrid method which combined the CBF and CF filtering approaches with supporting ontology model mapping, and this is the core component of the framework. In the following sections, we explain in detail how each element of the hybrid approach works.

A) CBF Method

We have different types of similarity in the CBF approach, such as cosine similarity, matching similarity and normalization similarity, depending on the nature of the feature. This is as follows:

- Used cosine similarity to calculate the course title and major course subject, according to the formula:

$$Sim\left(If_a, If_b\right) = \frac{\vec{If}_a . \vec{If}_b}{||\vec{If}_a|| \times ||\vec{If}_b||}$$

- **Course fee similarity calculation**: The similarity between the university course fees and the user preferred fees has been calculated by using the following formula:

$$FS\,(U, C) = \frac{F_{umax} - F_c}{F_{max} - (F_{min} - 1)} \qquad (4)$$

where:
$FS\ (U,\ C)$ = the course fee similarity between the user preferred fee and the course fee for each university
F_{umax} = the maximum university course fee that is expected from the user
F_{min} = the minimum university course fee in the database
F_c = the university course fee

- **Location similarity calculation**: The matching similarity has been used to compute the similarity between the user location and the location of the university providing the courses.

- **University ranking similarity calculation**: we calculated the ranking attribute in the user query and course profile, according to the formula:

$$RS\,(U, C) = \frac{R_{max} - R_c}{R_{max} - R_{min}} \qquad (5)$$

where:
$RS\,(U, C)$ = the university ranking similarity between the user preferred ranking and university ranking
R_{max} = the maximum university ranking in the database
R_{min} = the minimum university ranking in the database
R_c = the ranking of the university which provides the course

- **NSS score similarity calculation:** to find the similarity between the NSS score of the course and the NSS score that the user is satisfied with, the following formula has been used:

$$NS\,(U, C) = \frac{U_N - (N_{min} - 1)}{N_{max} - (N_{min} - 1)} \qquad (6)$$

where:
$NS\,(U,\ C)$ = the NSS score similarity between the user and the course
U_N = the user preferred NSS score
N_{min} = the minimum NSS score in the database
N_{max} = the maximum NSS score in the database

B) CF METHOD

The most important aspect of the CF is how to measure the similarity between the active user and the other users in the database. In addition, a new algorithm has been produced in order to enhance the KNN algorithm by using the ontology similarity called (OKNN). In the following sub-sections, each part will be presented in detail.

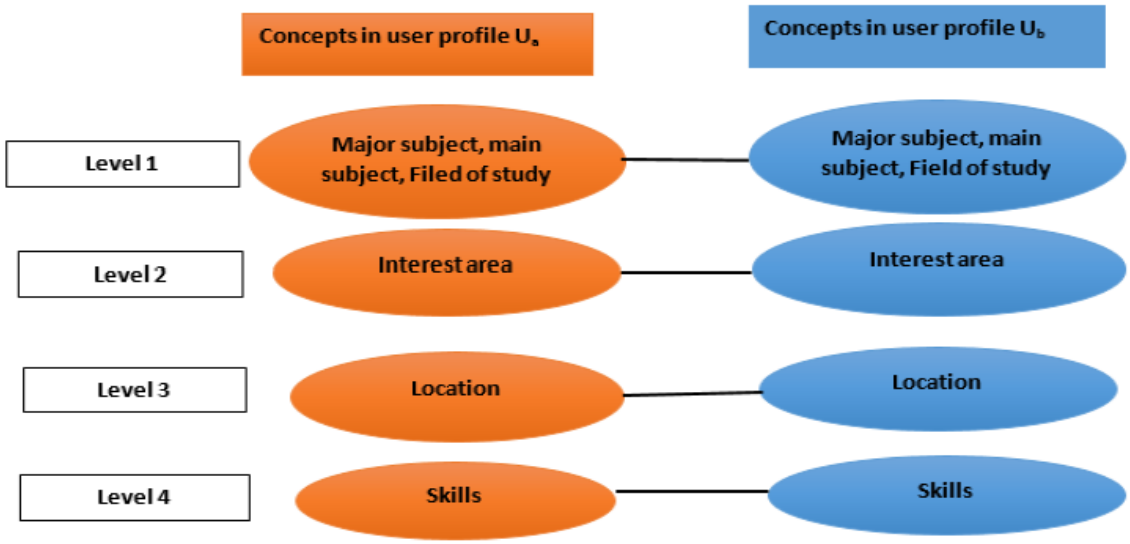


Fig. 5.9: Hierarchical matching and matching parameters

(a) User similarity calculation

The user profile vector consists of two parts; the first part is the user attributes, such as personal and academic information. The second part is the ratings that the user gives the item in the CBF case. The user similarity value range will be between (0, 1) and the weight for each part 50%.

US (Ua ,Un) = ontology similarity +recommendation history similarity

Where: (US) is a similarity between the target user Ua and the users in the system Un. The system considers the levels of the ontology concepts in the user profile by classifying the ontology similarity to four levels. Moreover, the given weight for each level is based on its importance, as follows:

Level 1 (major subject, main subject, the field of study)

Level2 (interest area)

Level3 (user location)

Level 4 (user skills)

The ontology similarity calculation between Ua, Ub will be based:

Moreover, after computing the ontology similarity it will be necessary to obtain the recommendation history similarity between Ua, Ub. In the proposed work, the recommendation history includes all the courses that have been rated by the user in the CBF case. Many algorithms have been applied to compute the similarity between the user recommendation histories. Cosine similarity is one of the algorithms that is most widely used in this area. The similarity between the users' recommendation histories has been computed accordingly to

$$Sim\,(U_a, U_b) = \frac{\vec{U}_a.\vec{U}_b}{||\vec{U}_a|| \times ||\vec{U}_b||} \quad (9)$$

$$Sim\,(U_a, U_b) = \frac{\sum_{p\in P} U_a.U_b}{\sqrt{\sum_{p\in P}(Ua)^2}\sqrt{\sum_{p\in P}(Ub)^2}}. \quad (10)$$

where:

$Sim\,(U_a, U_b)$ = cosine similarity of two vectors

P = the set of courses that have been rated by user U_a and U_a

6.3 Graphical User Interface

The user interface (UI), in the industrial design field of human-computer interaction, is the space where interactions between humans and software occur. The goal of this interaction is to allow effective operation and control of the machine from the human end, whilst the machine simultaneously feeds back information that aids the operators' decision-making process.

Generally, the goal of user interface design is to produce a user interface which makes it easy, efficient, and enjoyable (user-friendly) to operate a machine in the way which produces the desired result. This generally means that the operator needs to provide minimal input to achieve the desired output, and also that the machine minimizes undesired outputs to the human.

6.3.1 HTML

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web. HTML elements are the basic building blocks of the HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to

create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML can embed programs written in a scripting language such as JavaScript, which affects the behaviour and content of web pages. The World Wide Web Consortium (W3C) maintains the standards for both HTML and CSS.

6.3.2 CSS

Cascading Style Sheets (CSS) is a simple mechanism for adding style (e.g., fonts, colors, and spacing) to Web Documents. CSS is designed to enable the separation of presentation and content including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

6.3.3 JAVASCRIPT

JavaScript often abbreviated as JS is a high-level, interpreted programming language. It is a language which is also characterized as dynamic, weakly-typed, prototype-based and metaparadigm. It supports event-driven, and imperative (including object-oriented and prototype based) programming styles. It has an API for working with text, arrays, dates, regular expression, and basic manipulation of the DOM.

6.3.4 PHP

Hypertext Pre-processor (or simply PHP) is a general-purpose programming language originally designed for web development. Hypertext Mark-up Language, a standardized system for tagging text files to achieve font, colour, graphic, and hyperlink effects on World Wide Web pages. MySQL is an Oracle-backed open source relational database management system (RDBMS) based on Structured Query Language (SQL). Database management system (DBMS) is system software for creating and managing databases. ADBMS makes it possible for end users to create, read, update and delete data in a database.

CHAPTER 7

ALGORITHMS

7.1 Ontology based k-nearest neighbor algorithm

The k-nearest neighbor users of the active user (target user) must be determined in order to make a recommendations list by CF, to achieve this result, we proposed a new algorithm, OKNN algorithm that combines the ontology similarity of the user profile attribute and the item rate when the recommendation history is applied. The k-nearest neighbor users to the target user are found by searching only those who exist among the same group, rather than all the users. For instance, if the target user has a main subject of Computer Sciences and their major is Computer Programming, the nearest neighbor will search for all the users who have Computer Sciences as a main subject in their profiles. A common rate problem we faced for the top k-nearest neighbor was that the same item had been rated by different values, respectively. In order to solve this problem, the following formula has been proposed:

$$\text{Average weight score} = \left(\left(\frac{\text{ARWC} * (\text{KNNW} - \text{Omax} * \text{K})}{\text{KNNW}}\right) + Oc * K\right)/100 \quad (11)$$

where:

KNNW = KNN weight in the final scoring function

ARW c = average weight of the rate for the current course ∗100%

Omax = the maximum occurrence of the rate in the recommendation history of all the top N users

K = constant (e.g. 2)

Oc = the number of occurrences of the current course has been rated

The algorithm is as follows:

1: For user Uc Get user profile and create vector

2: while there are users to compare U do

3: Create vector for U

4: Calculate the Similarity between U and Uc by using Formula 7

5: Sort the nearest neighbour list

6: Get the top 5 nearest neighbour

7: for each user in top 5 list do

8: for each course in the user's recommendation history

9: If C rate >= 3 then

10: add the C to the KNN list

11: end If

12: end for

13: end for

14: for each C in KNN list do

15: Calculate the C rate using Equation 11

16: Update the KNN list using new score

17: end for

18:end while

7.2 Final Scoring Algorithm

The proposed approach to filtering combines CBF and CF with ontology to recommend courses to the user. For the new user, the system will recommend courses based on his/her profile. The recommendation process will begin based on the OPCR algorithm by creating a vector of users and courses. The final recommendation list is produced by using the final scoring function (FSF). FSF combines the similarity score of a content-based filtering list and a collaborative filtering list. Moreover, the other factor will be added to the final score as well, such as the university ranking and the NSS score as shown in the Eq. (12). The value of the final score function similarity should be between the ranges (0-1). The weight percentage for each part in FCF (CBF, CF, university rank, NSS score) is 50%, 30%, 10%, 10%, respectively.

Final Scoring Function (FSF) = (CBF∗ (50%)) +CF ∗ (30%)) + (university rank∗ (10%)) + (NSS score8 (10%))

The algorithm is as follows:

1: Calculate course score based on the user's profiles and query term

2: Calculate course NSS score similarity by using formula

3: Calculate university rank similarity by using formula

4:Calculate course fees similarity by using formula

5:Rank and get the top 10 courses which have the highest similarity score.

6:Recommend the user by top 10 score.

7:If the user chooses any of the recommended courses then

8:Use OKNN algorithm to determine the five nearest neighbours of the current user Uc. . . Calculate course scores.

9:add top 10 courses to the recommendation list

10:end if

11:Return the refined recommendations to the user.

CHAPTER 8

IMPLEMENTATION

8.1 Back-end Code

```
Y = load('course-ratings.txt');
R = logical(Y);
Avg = sum(Y,1)./sum(Y~=0,1);
course_index = [1:1:size(Y,2)];
figure('name','Average ratings for courses');
bar(course_index,Avg);
ylim([0 10]);
ylabel('Average Rating');
xlabel('Course Index');
fid = fopen('course-list.txt');
n = 15;
courseList = cell(n, 1);
for i = 1:n

    courseList{i} = strtrim(fgets(fid));
end
fclose(fid);
s=cell(1,15);
[s{1} s{2} s{3} s{4} s{5} s{6} s{7} s{8} s{9} s{10} s{11} s{12} s{13} s{14}
s{15}]=f2(courseList{1},courseList{2},courseList{3},courseList{4},courseList{5
},courseList{6},courseList{7},courseList{8},courseList{9},courseList{10},cours
eList{11},courseList{12},courseList{13},courseList{14},courseList{15});
student_rating = cell2mat(s);
rated_index = [];
for i=1:15
    if student_rating(i) ~=0
        rated_index = [rated_index i];
    end
end



fprintf('\nTraining collaborative filtering model\n');
student_rating=student_rating';
Y = [student_rating Y'];
R = [(student_rating ~= 0) R'];
Courses = size(Y,1);
Students = size(Y,2);
Features = 9;
X = randn(Courses, Features);
Theta = randn(Students, Features);
initial_parameters = [X(:); Theta(:)];
options = optimset('GradObj', 'on', 'MaxIter', 100);
%  Setting Regularization Parameter lambda
lambda = 3.2;
theta = fmincg (@(t)(CostFunction(t, Y, R, Students, Courses, Features,
```

```
    lambda)), initial_parameters, options);
    X = reshape(theta(1:Courses*Features), Courses);
    Theta = reshape(theta(Courses*Features+1:end), Students, Features);
    fprintf('Recommender system learning completed.\n');
    p = X * Theta';
    my_predictions = p(:,1);
    for i = 1:length(rated_index)
        course = rated_index(i);
        my_predictions(course) = 0;
    end
    [r, index] = sort(my_predictions, 'descend');
    fprintf('\nYour ratings :-\n');
    for i = 1:length(student_rating)
        if student_rating(i) > 0
            fprintf('Rated %d for %s\n', student_rating(i), courseList{i});
        end
    end
    fprintf('\nCourses you may like :-\n');
    for i=1:3
        j = index(i);
        fprintf('Predicted rating %.1f : %s\n', my_predictions(j), courseList{j});
    end
    GUI3(my_predictions(index(1)), courseList(index(1)),my_predictions(index(2)),
    courseList(index(2)),my_predictions(index(3)), courseList(index(3)));
```

8.2 Front-End Code

8.2.1 Admin Login

```
<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

    <meta charset="utf-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1, maximum-
scale=1" />

    <meta name="description" content="" />

    <meta name="author" content="" />


    <title>Admin Login</title>

    <link href="assets/css/bootstrap.css" rel="stylesheet" />

    <link href="assets/css/font-awesome.css" rel="stylesheet" />
```

```
    <link href="assets/css/style.css" rel="stylesheet" />
</head>
<body>

    <div class="content-wrapper">
        <div class="container">
            <div class="row">
                <div class="col-md-12">
                    <h4 class="page-head-line">Please Login To Enter </h4>

                </div>

            </div>
             <span style="color:red;" ></span>
            <form name="admin" method="post">
            <div class="row">
                <div class="col-md-6">
                    <label>Enter Username : </label>
                      <input type="text" name="username" class="form-control" required />
                      <label>Enter Password :  </label>
                      <input type="password" name="password" class="form-control" required />
                      <hr />
                      <button type="submit" name="submit" class="btn btn-info"><span
class="glyphicon glyphicon-user"></span>  Log Me In </button> 
                </div>
                </form>

                            </div>

            </div>
```

```
        </div>
    </div>

    <script src="assets/js/jquery-1.11.1.js"></script>

    <script src="assets/js/bootstrap.js"></script>
</body>
</html>
```

8.2.2 Change Password

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1, maximum-
scale=1" />
    <meta name="description" content="" />
    <meta name="author" content="" />
    <title>Admin | Student Password</title>
    <link href="assets/css/bootstrap.css" rel="stylesheet" />
    <link href="assets/css/font-awesome.css" rel="stylesheet" />
    <link href="assets/css/style.css" rel="stylesheet" />
</head>
<script type="text/javascript">
function valid()
{
if(document.chngpwd.cpass.value=="")
{
alert("Current Password Filed is Empty !!");
document.chngpwd.cpass.focus();
return false;
}
else if(document.chngpwd.newpass.value=="")
{
alert("New Password Filed is Empty !!");
document.chngpwd.newpass.focus();
return false;
}
else if(document.chngpwd.cnfpass.value=="")
{
alert("Confirm Password Filed is Empty !!");
document.chngpwd.cnfpass.focus();
return false;
```

```
}
else if(document.chngpwd.newpass.value!= document.chngpwd.cnfpass.value)
{
alert("Password and Confirm Password Field do not match  !!");
document.chngpwd.cnfpass.focus();
return false;
}
return true;
}
</script>
<body>


   <div class="content-wrapper">
      <div class="container">
          <div class="row">
              <div class="col-md-12">
                 <h1 class="page-head-line">Student Change Password </h1>
              </div>
           </div>
           <div class="row" >
             <div class="col-md-3"></div>
              <div class="col-md-6">
                 <div class="panel panel-default">
                 <div class="panel-heading">
                   Change Password
                 </div>
<font color="green" align="center"></font>


                  <div class="panel-body">
                 <form name="chngpwd" method="post" onSubmit="return valid();">
  <div class="form-group">
   <label for="exampleInputPassword1">Current Password</label>
   <input type="password" class="form-control" id="exampleInputPassword1"
name="cpass" placeholder="Password" />
 </div>
  <div class="form-group">
   <label for="exampleInputPassword1">New Password</label>
   <input type="password" class="form-control" id="exampleInputPassword2"
name="newpass" placeholder="Password" />
 </div>
 <div class="form-group">
   <label for="exampleInputPassword1">Confirm Password</label>
   <input type="password" class="form-control" id="exampleInputPassword3"
name="cnfpass" placeholder="Password" />
 </div>

 <button type="submit" name="submit" class="btn btn-default">Submit</button>
                    <hr />
```

```
</form>
                    </div>
                    </div>
                </div>

            </div>
        </div>
    </div>

    <script src="assets/js/jquery-1.11.1.js"></script>

    <script src="assets/js/bootstrap.js"></script>
</body>
</html>
```

8.2.3 Course Registration

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1, maximum-
scale=1" />
    <meta name="description" content="" />
    <meta name="author" content="" />
    <title>Admin | Course</title>
    <link href="assets/css/bootstrap.css" rel="stylesheet" />
    <link href="assets/css/font-awesome.css" rel="stylesheet" />
    <link href="assets/css/style.css" rel="stylesheet" />
</head>

<body>

    <div class="content-wrapper">
        <div class="container">
            <div class="row">
                <div class="col-md-12">
                    <h1 class="page-head-line">Course  </h1>
                </div>
            </div>
            <div class="row" >
              <div class="col-md-3"></div>
                <div class="col-md-6">
                    <div class="panel panel-default">
                    <div class="panel-heading">
```

```
                    Course
                  </div>
<font color="green" align="center"></font>


                   <div class="panel-body">
                  <form name="dept" method="post">
   <div class="form-group">
    <label for="coursecode">Course Code  </label>
    <input type="text" class="form-control" id="coursecode" name="coursecode"
placeholder="Course Code" required />
  </div>

 <div class="form-group">
    <label for="coursename">Course Name  </label>
    <input type="text" class="form-control" id="coursename" name="coursename"
placeholder="Course Name" required />
  </div>

<div class="form-group">
    <label for="courseunit">Course unit  </label>
    <input type="text" class="form-control" id="courseunit" name="courseunit"
placeholder="Course Unit" required />
  </div>


 <button type="submit" name="submit" class="btn btn-default">Submit</button>
</form>
                    </div>
                    </div>
                </div>

            </div>
            <font color="red" align="center"></font>
            <div class="col-md-12">

                <div class="panel panel-default">
                    <div class="panel-heading">
                        Manage Course
                    </div>

                    <div class="panel-body">
                        <div class="table-responsive table-bordered">
                            <table class="table">
                                <thead>
                                    <tr>
                                        <th>Serial no.</th>
                                        <th>Course Code</th>
                                        <th>Course Name </th>
                                        <th>Course Unit</th>
```

```
                                <

                              </tr>
                            </thead>
                            <tbody>



                              <tr>


<button class="btn btn-primary"><i class="fa fa-edit "></i> Edit</button> </a>
  <a href="course.php?id=<?php echo $row['id']?>&del=delete" onClick="return
confirm('Are you sure you want to delete?')">
                                  <button class="btn btn-danger">Delete</button>
</a>
                                  </td>
                              </tr>



                            </tbody>
                          </table>
                        </div>
                      </div>
                    </div>

                  </div>
                </div>




        </div>
    </div>

    <script src="assets/js/jquery-1.11.1.js"></script>

    <script src="assets/js/bootstrap.js"></script>
</body>
</html>
```

8.2.4 Course Selection

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <meta charset="utf-8" />
```

```
   <meta name="viewport" content="width=device-width, initial-scale=1, maximum-
scale=1" />
   <meta name="description" content="" />
   <meta name="author" content="" />
   <title>Admin | Course</title>
   <link href="assets/css/bootstrap.css" rel="stylesheet" />
   <link href="assets/css/font-awesome.css" rel="stylesheet" />
   <link href="assets/css/style.css" rel="stylesheet" />
</head>

<body>

   <div class="content-wrapper">
      <div class="container">
          <div class="row">
              <div class="col-md-12">
                 <h1 class="page-head-line">Course  </h1>
              </div>
           </div>
           <div class="row" >
            <div class="col-md-3"></div>
              <div class="col-md-6">
                 <div class="panel panel-default">
                 <div class="panel-heading">
                   Course
                 </div>
<font color="green" align="center"></font>


                  <div class="panel-body">
                 <form name="dept" method="post">

<p><b>Last Updated at</b> </p>
  <div class="form-group">
   <label for="coursecode">Course Code  </label>
   <input type="text" class="form-control" id="coursecode" name="coursecode"
placeholder="Course Code" value=" " required />
  </div>

 <div class="form-group">
   <label for="coursename">Course Name  </label>
   <input type="text" class="form-control" id="coursename" name="coursename"
placeholder="Course Name" value=" " required />
  </div>

<div class="form-group">
   <label for="courseunit">Course unit  </label>
   <input type="text" class="form-control" id="courseunit" name="courseunit"
placeholder="Course Unit" value="" required />
  </div>
```

```
 <button type="submit" name="submit" class="btn btn-default"><i class=" fa fa-refresh
"></i> Update</button>
</form>
                        </div>
                        </div>
                    </div>

                </div>

            </div>




        </div>
    </div>

    <script src="assets/js/jquery-1.11.1.js"></script>

    <script src="assets/js/bootstrap.js"></script>
</body>
</html>
```

8.2.5 Course Enroll

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1, maximum-
scale=1" />
    <meta name="description" content="" />
    <meta name="author" content="" />
    <title>Course Enroll</title>
    <link href="assets/css/bootstrap.css" rel="stylesheet" />
    <link href="assets/css/font-awesome.css" rel="stylesheet" />
    <link href="assets/css/style.css" rel="stylesheet" />
</head>

<body>

    <!-- MENU SECTION END-->
    <div class="content-wrapper">
        <div class="container">
            <div class="row">
                <div class="col-md-12">
```

```
                    <h1 class="page-head-line">Course Enroll </h1>
                  </div>
                </div>
                <div class="row" >
                  <div class="col-md-3"></div>
                    <div class="col-md-6">
                      <div class="panel panel-default">
                      <div class="panel-heading">
                        Course Enroll
                      </div>
<font color="green" align="center"></font>


                      <div class="panel-body">
                     <form name="dept" method="post" enctype="multipart/form-data">
   <div class="form-group">
    <label for="studentname">Student Name  </label>
    <input type="text" class="form-control" id="studentname" name="studentname" value="
"  />
  </div>

 <div class="form-group">
    <label for="studentregno">Student Reg No   </label>
    <input type="text" class="form-control" id="studentregno" name="studentregno" value="
" />

  </div>


<div class="form-group">
    <label for="Session">Session  </label>
    <select class="form-control" name="session" required="required">
   <option value="">Select Session</option>


<option value="<?php echo htmlentities($row['id']);?>">

</option>


    </select>
  </div>



<div class="form-group">
    <label for="Department">Department  </label>
    <select class="form-control" name="department" required="required">
   <option value="">Select Depertment</option>
```

```
<option value="<?php echo htmlentities($row['id']);?>">

</option>


   </select>
  </div>


<div class="form-group">
   <label for="Level">Level  </label>
   <select class="form-control" name="level" required="required">
  <option value="">Select Level</option>


<option value="<?php echo htmlentities($row['id']);?>">

   </select>
  </div>

<div class="form-group">
   <label for="Semester">Semester  </label>
   <select class="form-control" name="sem" required="required">
  <option value="">Select Semester</option>

<option value="<?php echo htmlentities($row['id']);?>"></option>


   </select>
  </div>


<div class="form-group">
   <label for="Course">Course  </label>
   <select class="form-control" name="course" id="course" onBlur="courseAvailability()"
required="required">
  <option value="">Select Course</option>

<option value="<?php echo htmlentities($row['id']);?>">
<?php } ?>
   </select>
   <span id="course-availability-status1" style="font-size:12px;">
  </div>



 <button type="submit" name="submit" id="submit" class="btn btn-
default">Enroll</button>
```

```
</form>
                    </div>
                    </div>
                </div>

            </div>

        </div>
    </div>
</div>

<script src="assets/js/jquery-1.11.1.js"></script>
<script src="assets/js/bootstrap.js"></script>
<script>
function courseAvailability() {
$("#loaderIcon").show();
jQuery.ajax({
url: "check_availability.php",
data:'cid='+$("#course").val(),
type: "POST",
success:function(data){
$("#course-availability-status1").html(data);
$("#loaderIcon").hide();
},
error:function (){}
});
}
</script>
</body>
</html>
```

CHAPTER 9

TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover faults or defects in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product it is the process of exercising software with the intent of ensuring that the software system meets its requirements and user exceptions and does not fail in an unacceptable manner.

A good test case is the one that has a high probability of finding an as yet undiscovered error. A successful test is the one that uncovers an undiscovered error. Testing may be carried out during the implementation phase to verify that the software behaves as intended by its designers and after implementation is complete.

9.1 SOFTWARE TESTING

A primary purpose of testing is to detect software failures so that defects may be discovered and corrected. Testing cannot establish that a product functions properly under all conditions but can only establish that it does not function properly under specific conditions. The scope of software testing often includes examination of code as well as execution of that code in various environments and conditions as well as examining the aspects of code: does it do what it is supposed to do and do what it needs to do. In the current culture of software development, a testing organization may be separate from the development team. There are various roles for testing team members. Information derived from software testing may be used to correct the process by which software is developed.

9.2 SOFTWARE TESTING TYPES

Software testing life cycle is the process that explains the flow of the tests that are to be carried on each step of software testing of the product. The V-model i.e. Verification and validation model is a perfect model which is used in the improvement of the software project. This model contains software development life cycle on one side and software testing life cycle on the other hand side. A checklist for software tester sets a baseline that guides him to carry on the day-to-day activities.

9.2.1BLACK BOX TESTING

It explains the process of giving the input to the system and checking the output, without considering how the system generates the output. It is also called as Behaviour Testing.

Functional Testing

In this type of testing, the software is tested for the functional requirements. This checks whether the application is behaving according to the specification.

Performance Testing

This type of testing checks whether the system is performing properly, according to the user's requirements. Performance testing depends upon the Load and Stress Testing that is internally or externally applied to the system.

Integration Testing

Integration Testing is the phase in software testing in which individual software modules are combined and tested as a group. This mostly focuses in the design and construction of the software architecture. The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items.

Integration testing is further classified into Bottom-up Integration and Top-Down Integration testing.

Bottom-up Integration Testing

It is an approach to integrated testing where the lowest level components are tested first, then used to facilitate the testing of higher level components. The process is repeated until the component at the top of the hierarchy is tested. All the bottom or lowlevel modules, procedures or functions are integrated and then tested. After the integration testing of lower level integrated modules, the next level of modules will be formed and can be used for integration testing.

Top-Down Integration testing

It is an approach to integrated testing where the top down integrated modules are tested and the branch of the module is tested step by step until the end of the related module.

System Testing

System testing is the testing conducted on a complete, integrated system, to evaluate the system's compliance with the specified requirements. This type of software testing validates that the system meets its functional and non-functional requirements and also intended to test beyond the bounds defined in the software/hardware requirement specifications.

9.2.2 WHITE BOX TESTING

It is the process of giving the input to the system and checking, how the system processes the input, to generate the output. It is mandatory for a tester to have the knowledge of the source code.

Integration Testing

Integration Testing can proceed in a number of different ways, which can be broadly characterized as top down or bottom up. In top down integration testing the high level control routines are tested first, possibly with the middle level control structures present only as stubs. Subprogram stubs were presented in section2 as incomplete subprograms which are only present to allow the higher. Level control routines to be tested. Top down testing can proceed in a depth-first or a breadth-first manner. For depth-first integration each module is tested in increasing detail, replacing more and more levels of detail with actual code rather than stubs. Alternatively breadth first would processed by refining all the modules at the same level of control throughout the application .in practice a combination of the two techniques would be used. At the initial stages all the modules might be only partly functional, possibly being implemented only to deal with non-erroneous data. These would be tested in breadth-first manner, but over a period of time each would be replaced with successive refinements which were closer to the full functionality. This allows depth-first testing of a module to be performed simultaneously with breadth-first testing of all the modules. The other major category of integration testing is Bottom-up Integration Testing where an individual module is tested form a test harness. Once a set of individual module have been tested they are then combined into a collection of modules ,known as builds, which are then tested by a second test harness. This process can continue until the build consists of the entire application. In practice a combination of top down and bottom-up testing would be used. In a large software project being developed by a number of sub-teams, or a smaller project where different modules were built by individuals. The sub teams or individuals would conduct bottom-up testing of the modules which they were constructing before releasing them to an integration team which would assemble them together for top-down testing.

Unit Testing

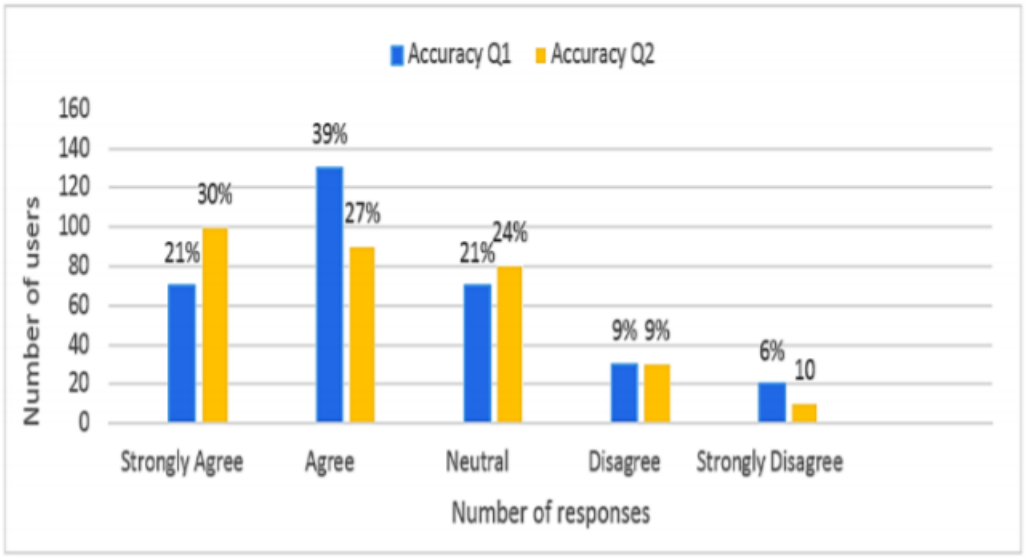
Unit testing deals with testing a unit as a whole. This would test the interaction of many functions but confine the test within one unit. The exact scope of a unit is left to interpretation. Supporting test code, sometimes called Scaffolding, may be necessary to support an individual test. This type of testing is driven by the architecture and implementation teams. This focus is also called black-box testing because only the details of the interface are visible to the test. Limits that are global to a unit are tested here. In the construction industry, scaffolding is a temporary, easy to assemble and disassemble, frame placed around a building to facilitate the construction of the building. The construction workers first build the scaffolding and then the building. Later the scaffolding is removed, exposing the completed building. Similarly, in software testing, one particular test may need some supporting software. This software establishes can a correct evaluation of the test take place. The scaffolding software may establish state and values for data structures as well as providing dummy external functions for the test. Different scaffolding software may be needed form one test to another test. Scaffolding software rarely is considered part of the system. Sometimes the scaffolding software becomes larger than the system software being tested. Usually the scaffolding software is not of the same quality as the system software and frequently is quite fragile. A small change in test may lead to much larger changes in the scaffolding. Internal and unit testing can be automated with the help of coverage tools. Analyses the source code and generated a test that will execute every alternative thread of execution. Typically, the coverage tool is used in a slightly different way. First the coverage tool is used to augment the source by placing information prints after each line of code. Then the testing suite is executed generating an audit trail. This audit trail is analysed and reports the percent of the total system code executed during the test suite. If the coverage is high and the untested source lines are of low impact to the system's overall quality, then no more additional tests are required.

CHAPTER 10

RESULT ANALYSIS

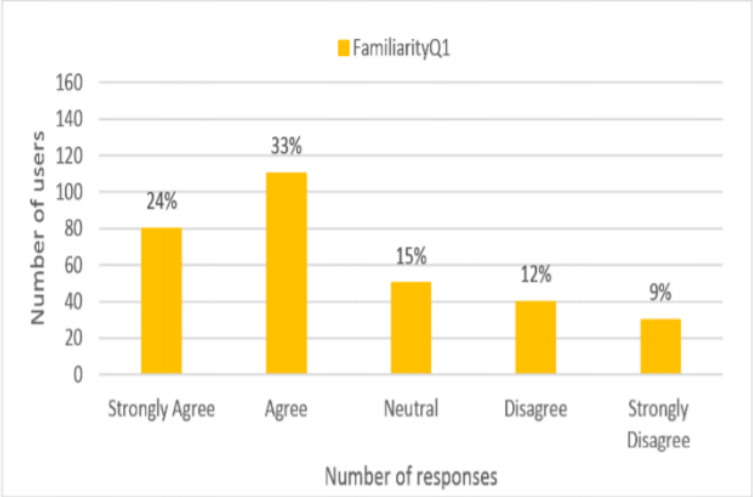
10.1 ACCURACY

1. The items recommended to me matched my interests.
2. This recommender system gave me good suggestions.



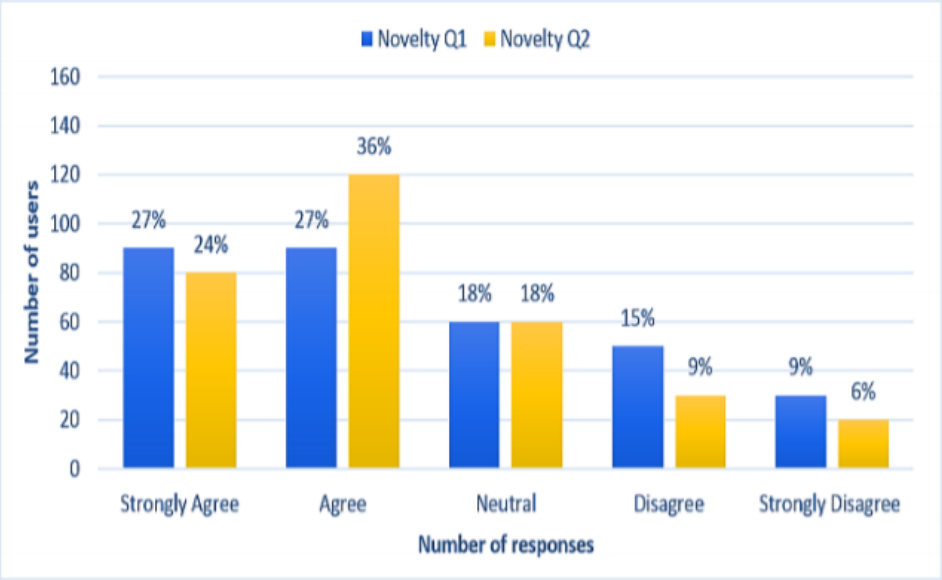
10.2 FAMILIARITY

1. Familiarity captures how well the users know some of the recommended items.
2. OPCR used an ontology-based recommendation technique to recommend the most relevant items to users.



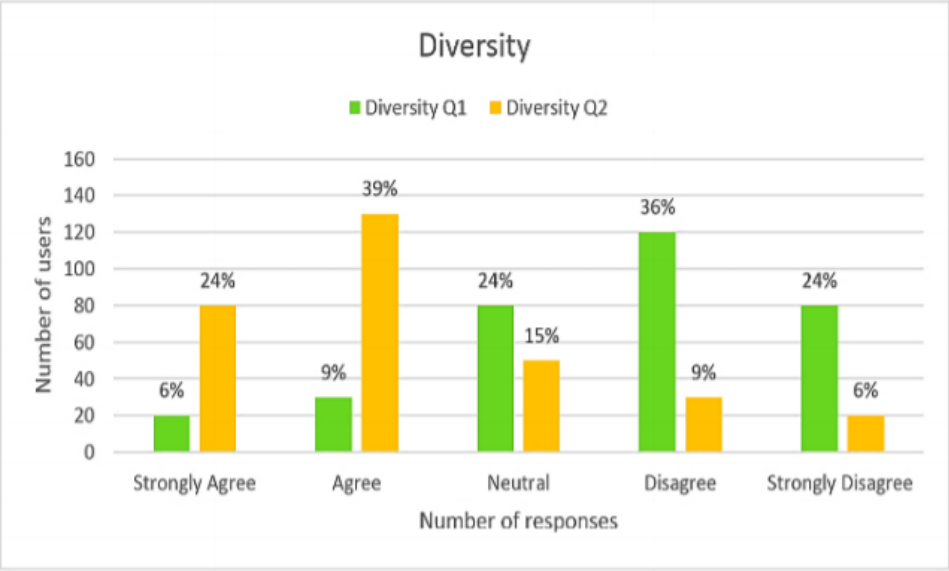
10.3 NOVELTY

1. Novelty is one of the important indicators of user satisfaction as it helps users in the decision-making process.
2. OPCR provided the users with recommendations that included novel items which were not expected because ontology mapping is able to link all of the attributes in the course profiles and user profiles.
3. Recommendations were included for novel items and also helped the user to discover new items, according to the results of the user's responses.



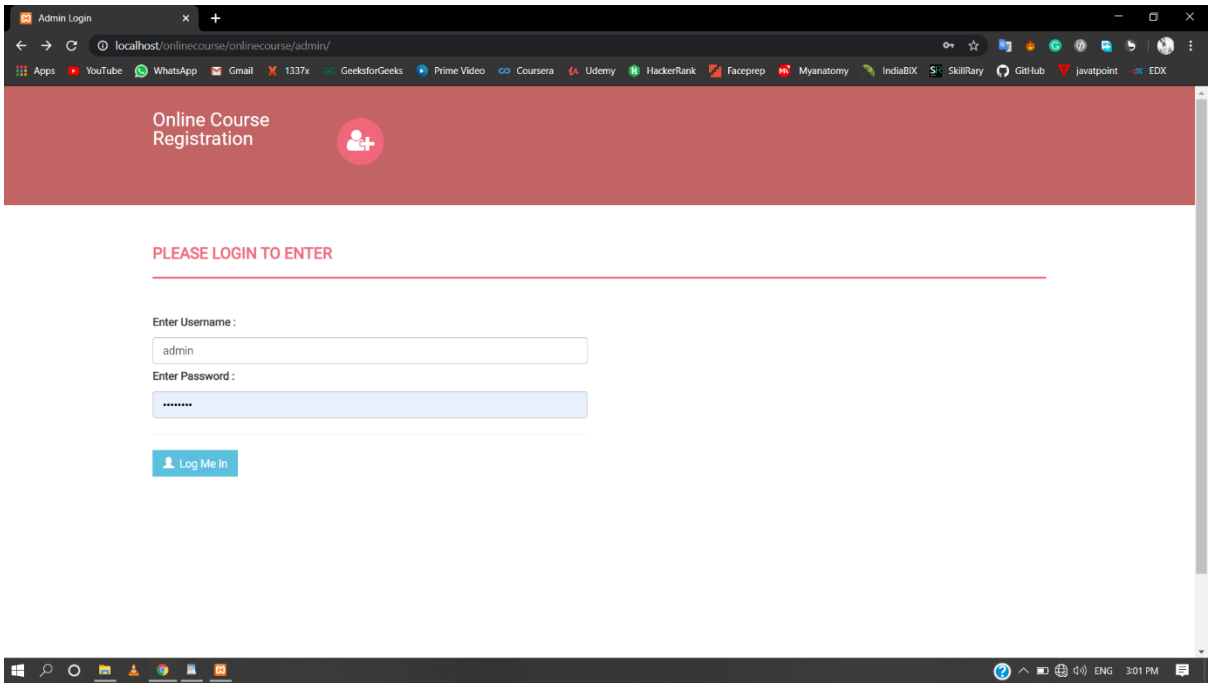
10.4 DIVERSITY

1. OPCR mainly recommended courses based on content-based filtering, which measures the similarity between the user profile and the item.
2. The recommendations are similar to each other because the ontology mapping technique will not allow irrelative items to appear with the recommendation items.

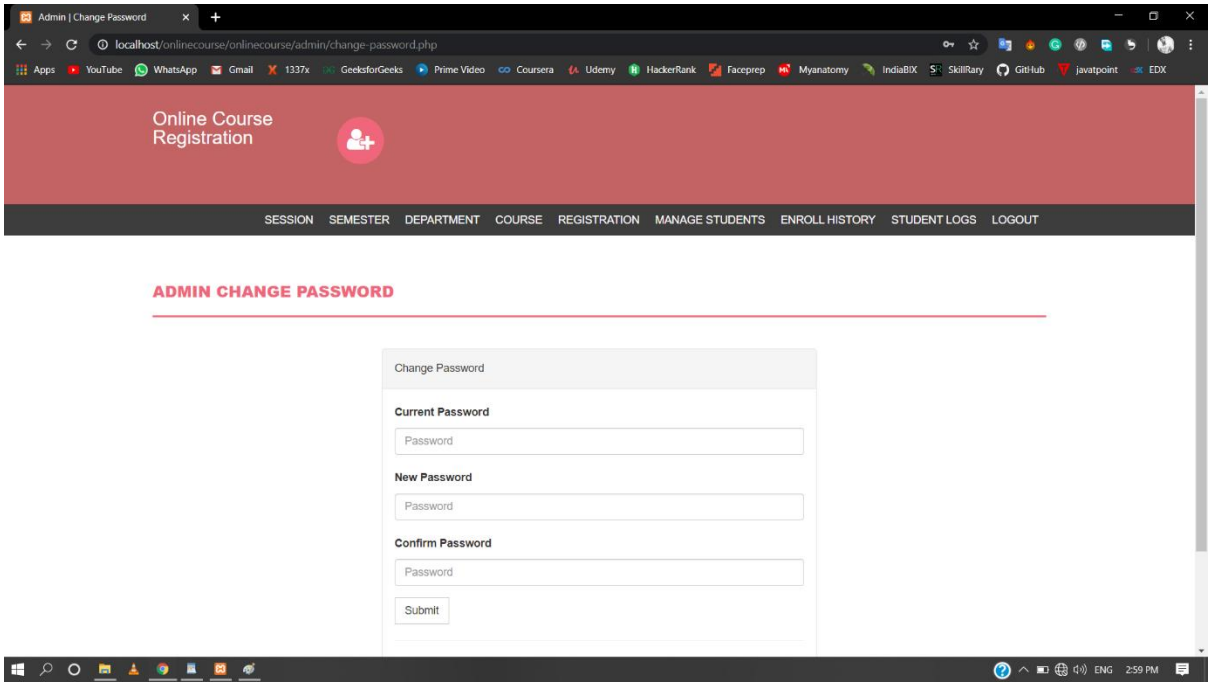

Diversity
Diversity Q1
Diversity Q2
Number of users
160
140
120
100
80
60
40
20
0
6%
24%
9%
39%
24%
15%
36%
9%
24%
6%
Strongly Agree
Agree
Neutral
Disagree
Strongly Disagree
Number of responses

CHAPTER 11

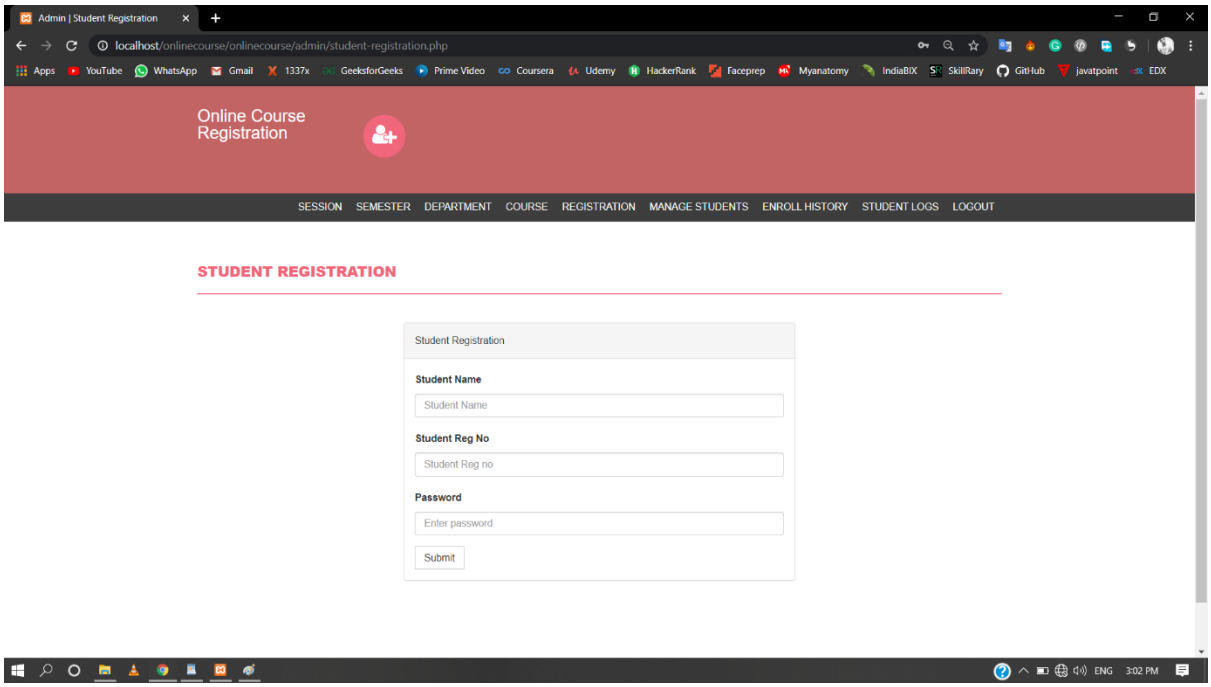
SNAPSHOTS



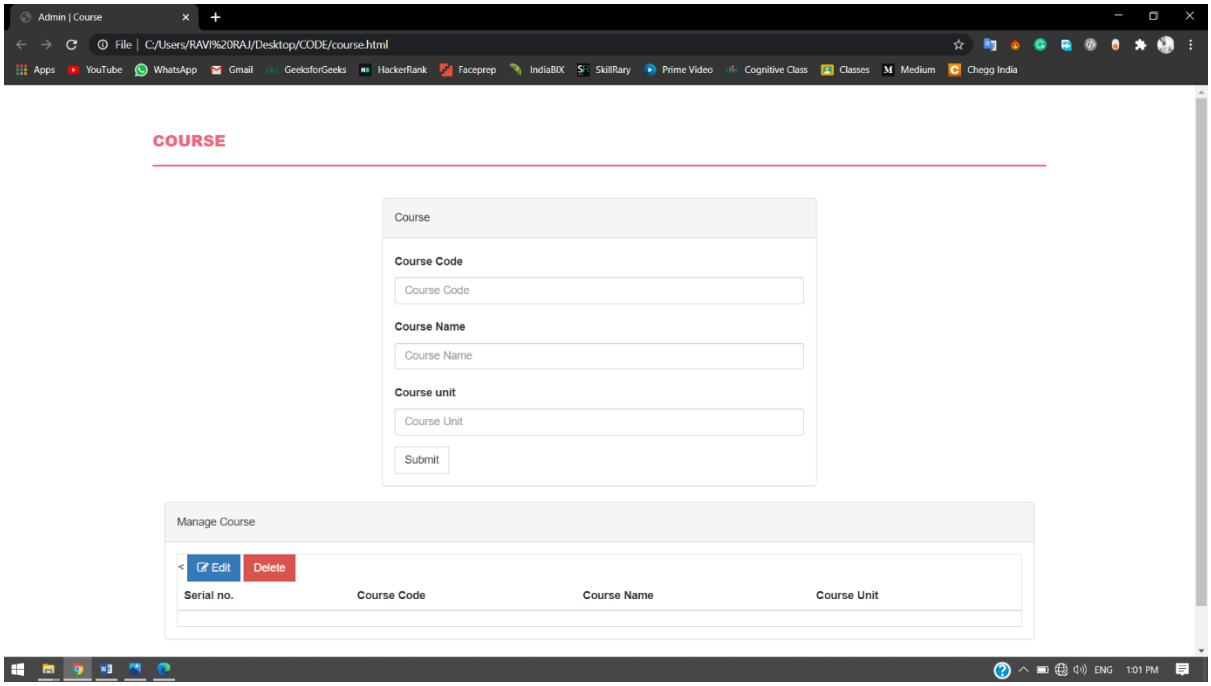
Snapshot 11.1 Login Page



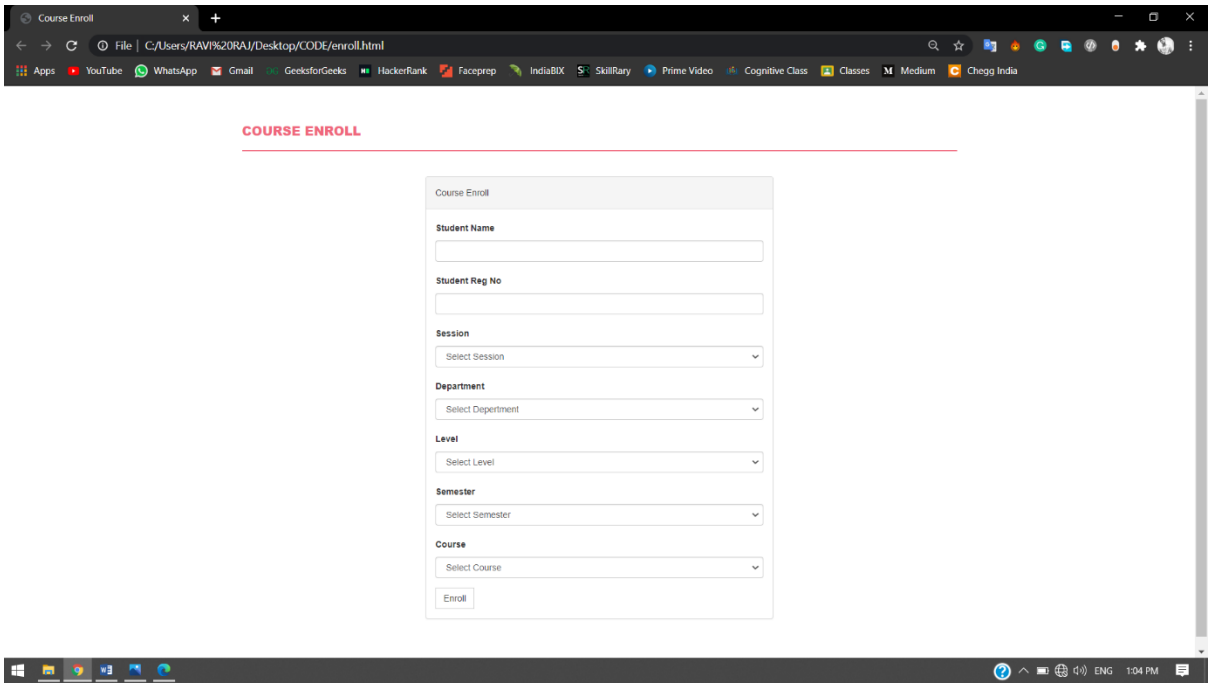
Snapshot 11.2 Admin Change Password Page



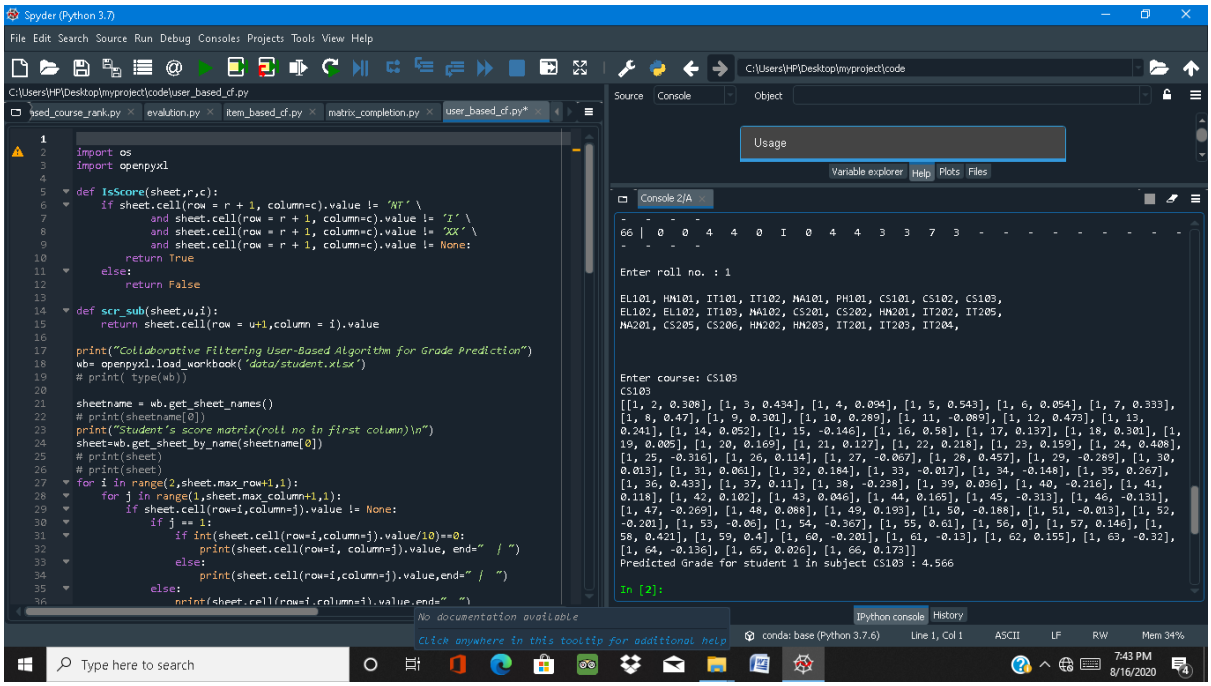
Snapshot 11.3 Student Registration Page



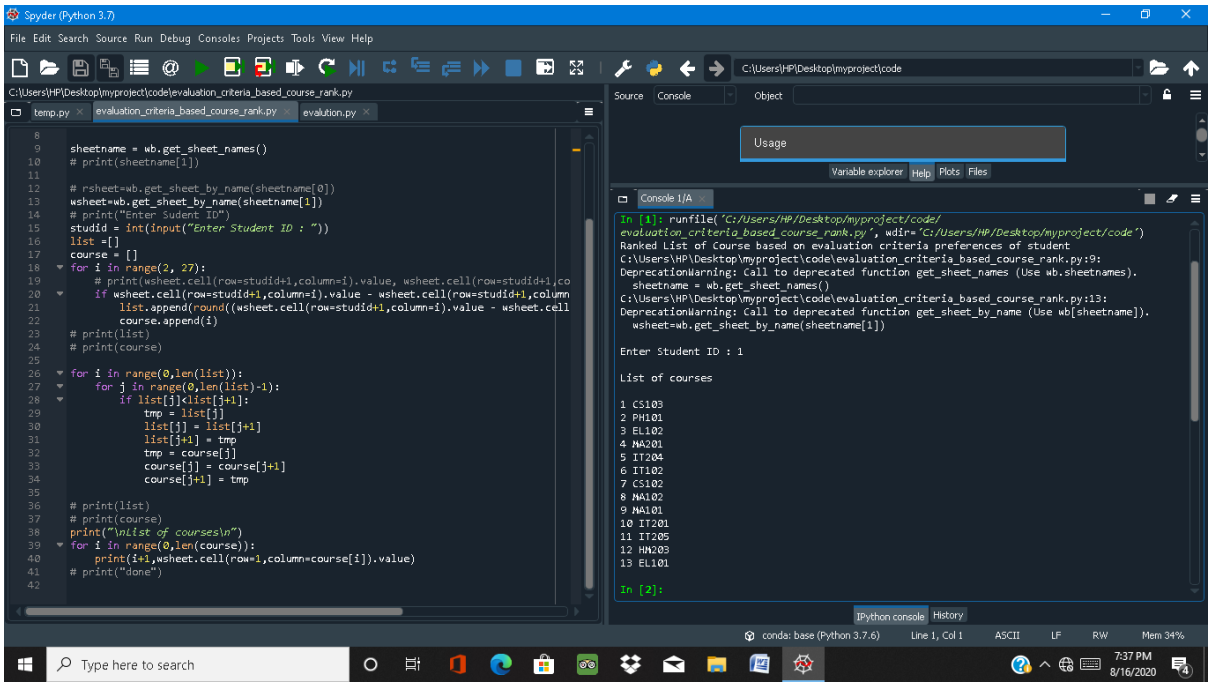
Snapshot 11.4 Course Selection



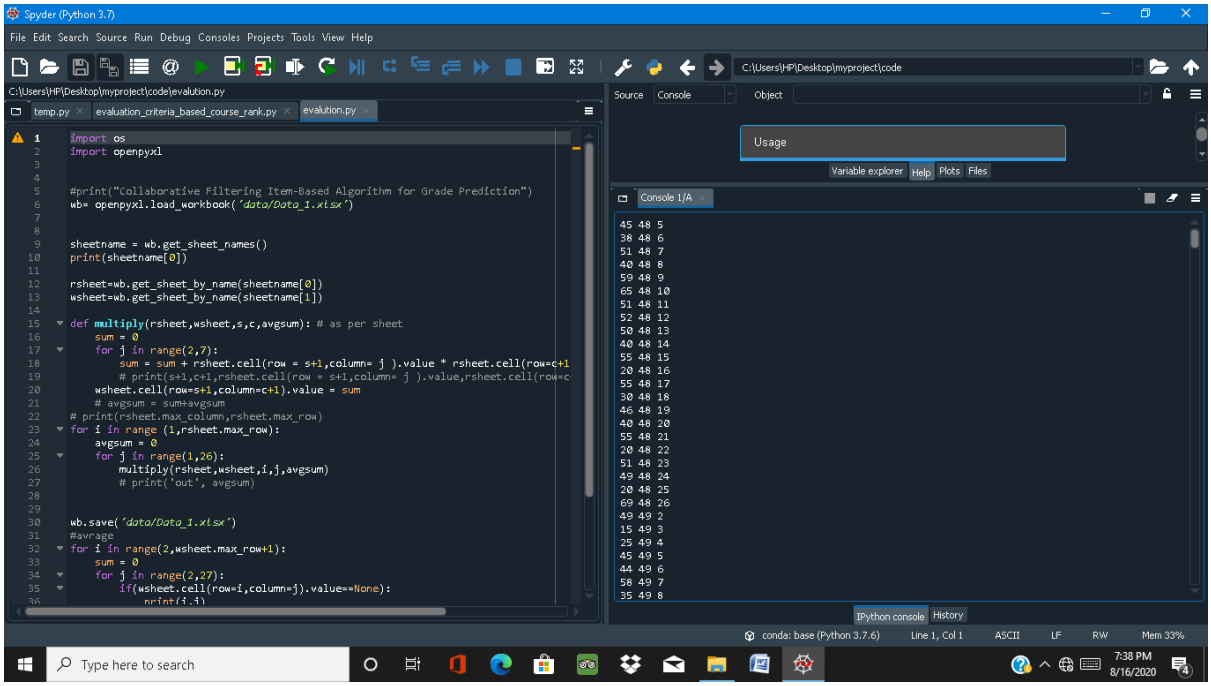
Snapshot 11.5 Course Enroll



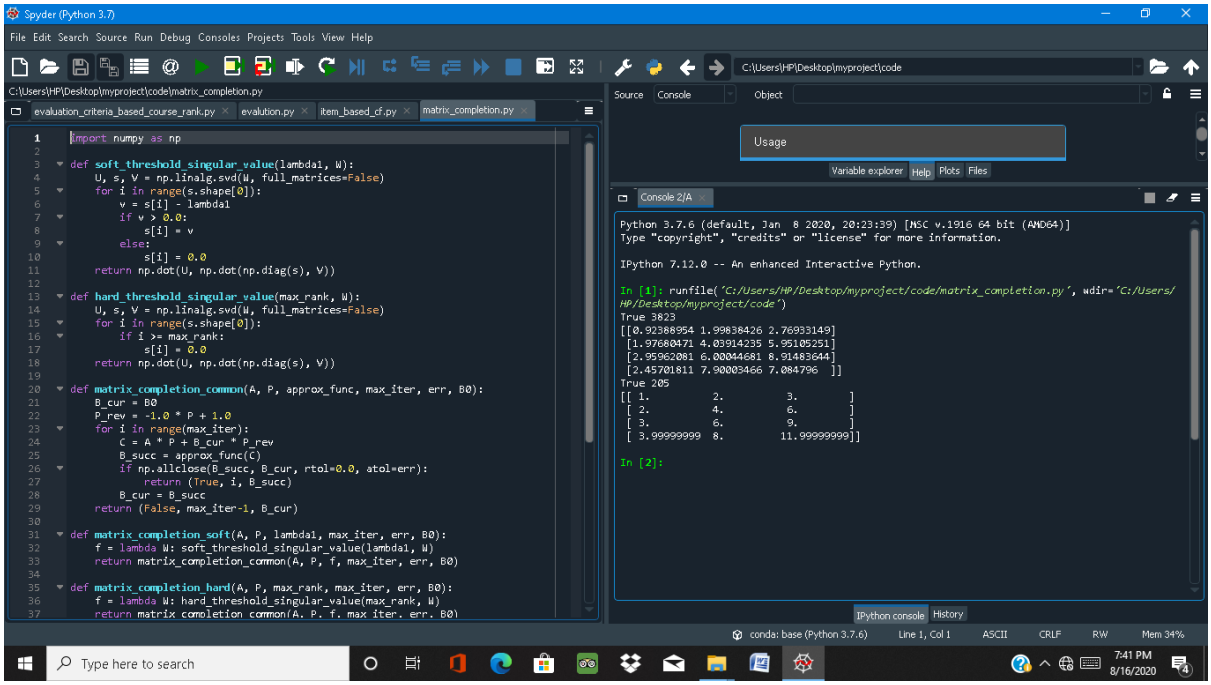
Snapshot 11.6 User Based



Snapshot 11.7 Course Rank



Snapshot 11.8 Evaluation based Course Outcomes



Snapshot 11.9 Matrix Completion

CONCLUSION

Searching and finding an item that is relevant to the user is a huge challenge. Choosing a higher education course at university is a massive decision for students. The recommendation system in education plays a vital role in overcoming the problem of information overloading, and helps the students to find relevant and useful courses from a large number of online course resources that are available on the internet. The current approaches to filtering have many limitations. To generate a comprehensive knowledge of the recommended items, information from multiple heterogenic sources needs to be mapped and linked. This paper proposes a novel approach to the recommendation system, which combines CBF and CF, supported by ontology similarity. OPCR algorithms are used to recommend university courses to a target student based on the user's interest and the choices made by similar students. The experiments showed that ontology matching is a desirable tool for making a recommendation to a target student, and it can be seen that the proposed approach can obtain better results, including greater user satisfaction and accuracy, than other approaches. Furthermore, the proposed approach can help to combat information overloading and the problems faced by new users by using ontology similarity between the users' profiles. Furthermore, using the ontology-based integration approach to integrate data from multiple heterogeneous sources will help the system to provide a comprehensive recommendation to users. Throughout the experiments, we noticed that building a new hybrid recommendation system which combines CF and CBF utilizing ontology improves the information overloading problem. Moreover, the ontology mapping and recommendation filter algorithms were incorporated to improve the accuracy of the recommendation and increase the user stratification of the recommendation. In addition, this approach improves the new user problem in the CF by incorporating ontology similarity into the proposed method. It was found that using dynamic ontology mapping to link the course profiles and student profile with the job profile helped to provide comprehensive knowledge about the course that was not only more relevant to the student oncology-based course recommender system, but also successfully brought a new dimension of ontology domain knowledge about the student and the course resources into the recommendation process.

FUTURE ENHACEMENT

In future, we will enrich our repository by absorbing more course and user information and heterogeneous data sources. In addition, we plan to incorporate additional user contexts, e.g., available student behavior, learning style and learning interests into the recommendation process in order to make the system more comprehensive and intelligent. We may employ more feedback information from students for effective courses and improve the student model based on students' feedback and consider more aspects and techniques related to recommender systems. We plan to carry out more experiments with a variety of actual students from different departments and from various academic backgrounds in order to prove the flexibility of our proposal.

REFERENCES

1. B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, ‘‘Item-based collaborative filtering recommendation algorithms”.
2. W. Yang, Z. Wang, and M. You, ‘‘An improved collaborative filtering method for recommendations’ generation”.
3. F. Le Roux, E. Ranjeet, V. Ghai, Y. Gao, and J. Lu, ‘‘A course recommender system using multiple criteria decision making method”.
4. R. Farzan and P. Brusilovsky, ‘‘Social navigation support in a course recommendation system,’’ in Adaptive Hypermedia and Adaptive WebBased Systems”.
5. J. Cuseo, ‘‘Academic advisement and student retention: Empirical connections & systematic interventions
6. R. Ren, L. Zhang, L. Cui, B. Deng, and Y. Shi, ‘‘Personalized financial news recommendation algorithm based on ontology”.
7. H. Imran, M. Belghis-Zadeh, T.-W. Chang, and S. Graf, ‘‘PLORS: A personalized learning object recommender system”.
8. H. Zhang, H. Yang, T. Huang, and G. Zhan, ‘‘DBNCF: Personalized courses recommendation system based on DBN in MOOC environment”.
9. R. G. Apaza, E. V. Cervantes, L. C. Quispe, and J. O. Luna, ‘‘Online courses recommendation based on LDA”.
10. C.-Y. Huang, R.-C. Chen, and L.-S. Chen, ‘‘Course-recommendation system based on ontology”.