**A PROJECT REPORT ON**

**A Hybrid Knowledge-Based Recommender System For**

**E-Learning Based**

**On Ontology And Sequential Pattern Mining**

Submitted to the

Savitribai Phule Pune University

In partial fulfilment for the award of the Degree of

**Bachelor of Engineering**

**In**

**Information Technology**

**By**

**JAYA SINGH (B120228520)**

**KILLADA LAVANYA (B120228523)**

**SHIMITA RUDRA (B120228548)**

**TANYA SAROHA (B120228556)**

Under the guidance of

**PROF GEETA PATIL**



**Department Of Information Technology**

**ARMY INSTITUTE OF TECHNOLOGY**

**(2017-2018)**



**ARMY INSTITUTE OF TECHNOLOGY**

**Department Of Information Technology**

**CERTIFICATE**

This is to certify that the project report entitled

**“A hybrid knowledge-based recommender system for e-learning based on ontology and sequential pattern mining”**

submitted by

**Jaya Singh (B120228520),**

**Killada Lavanya (B120228523),**

**Shimita Rudra (B120228548),**

**Tanya Saroha (B120228556)**

is a record of bonafide work carried out bythem under the supervision and guidance of **Prof. Geeta Patil** in partial fulfilment of therequirement of **Bachelor of Engineering (Information Technology)** course of Savitribai Phule PuneUniversity, Pune in the academic year 2017-2018.

Prof Geeta Patil Dr (Mrs.) Sangeeta Jadhav

(Project Guide)  **(**Head of the Department) Information Technology Information Technology

External Examiner Dr B P Patil (Principal) Army Institute of Technology

Date:

Place:

**ACKNOWLEDGEMENT**

It is our proud privilege to acknowledge the kind of help and guidance received from several people in preparation of this project report. This would not have been possible without the kind support and valuable help of many individuals. We would like to extend our sincere thanks to all of them.

We are highly indebted to our project guide **Prof Geeta Patil** for her guidance and constant supervision as well as for providing necessary information regarding the project at every step as we proceed. It was her kind co-operation and encouragement which helped us indeed.

Our sincere thanks to **Dr.(Mrs.) Sangeeta Jadhav**, the Head of Department, Information Technology, for her valuable suggestions and guidance throughout the period of this project ideation.

I also wish to acknowledge John K. Tarus, Zhendong Niua, Abdallah Yousif; the authors of the base paper of the project. I would like to express my sincere gratitude and humble respect to them for coming up with a research paper.

Jaya Singh (B120228520)

Killada Lavanya (B120228523)

Shimita Rudra (B120228548)

Tanya Saroha (B120228556)

-i-

**ABSTRACT**

In recent years, there has been an advancement in the use of online resources by consumers, namely platform sharing, support software, educational resource or even shopping online. However, due to information overload, many consumers are experiencing difficulties in retrieving useful and relevant resources that meet their needs. Although existing recommender systems have recorded significant success in e-commerce domain, they still experience drawbacks in making accurate recommendations of learning resources in e-learning domain due to differences in learner characteristics such as learning style, knowledge level as well as learners’ sequential learning patterns. Most of the existing online learning recommendation techniques do not consider differences in learner characteristics. This problem can be eased by incorporating additional information about the learner into the recommendation process. Moreover, many recommendation techniques experience cold-start and rating sparsity problems. We propose a hybrid knowledge-based recommender system based on collaborative filtering, ontology and matrix factorization (MF) for recommendation of e-learning resources to learners. In the approach, ontology is used to model and represent the domain knowledge about the learner and learning resources for discovering the learners’ sequential learning approach. Our approach involves four steps: (1) creating ontology to represent knowledge about the learner and learning resources, (2) computing ratings similarity based on ontology domain knowledge and making predictions for the target learner, (3) application of matrix factorization to remove sparsity problem, and (4) generation of top N learning items by the collaborative filtering recommendation engine giving us the final recommendations for the target learner. The proposed hybrid approach can reduce both the cold-start and data sparsity problems by making use of ontological domain knowledge and matrix factorization respectively before the initial data to work on is available in the recommender system.

-ii-

**LIST OF FIGURES**

**Sr.No** **Figure Name Page No**

1. Recommendation process of in CF 7
2. Structure of the top-level ontology of e-learning 8
3. Hybrid recommendation model 14
4. General Layout of Learner model ontology 15
5. General Layout of Learning Resource ontology 16

-iii-

**CONTENTS**

Acknowledgement -i-

Abstract -ii-

List of Figures -iii-

**Sr.No Chapter Page No**

1. Introduction
   1. Introduction to Problem Statement 1
   2. Aim 3
   3. Objective 3
   4. Brief Overview 4
2. Background and Literature Survey
   1. History 5
   2. Recommender System for E-Learning 5
   3. Recommendation Techniques 7
3. Requirements and Analysis
   1. Creating Learner and Learning Resources Ontology 10
   2. Computing Similarities and Predictions 11
   3. Generation of Top N List of Recommended Learning Resources 12
   4. Generation of Final Recommendations Based on GSP Algorithm 12
4. Design
   1. Architecture 14
   2. Learner Ontology Layout 15
   3. Learning Resource Ontology 16
   4. Algorithm for Searching Top-N Recommendations 17
5. Implementation
6. Conclusion 18

Annexure A. References 19

Annexure B. Plagiarism Report

**CHAPTER 1**

**INTRODUCTION**

The explosive growth of available digital data in the past decades has created a reservoir of information while the number of visitors on the Internet have escalated as well. This has led to information overload, which hinders access to items of interest on the Internet. This has increased the demand for recommender systems more than ever before. Recommender systems are information filtering systems that seek to filter vital information fragments out of large amount of dynamically generated information according to user’s preferences, interest, or observed behaviour about an item. Recommender systems have become increasingly popular, and are utilized in a variety of areas including music, news, movies, research articles, etc. Most of the existing approaches to recommender systems focus on recommending the most relevant content to users using contextual information and do not take into account the risk of disturbing the user in specific situation. Typically, research on recommender systems is concerned about finding the most accurate recommendation algorithms. The major reason for choosing E-learning was because of its importance in the present-day scenario. How books and theories are now being turned into electronic form making it more comprehensive and its convenience of use worldwide. E-learning describes the cognitive principles of effective multimedia learning using electronic educational technology. Cognitive research and theory suggest that the selection of appropriate concurrent multimedia modalities may enhance learning, as may application of several other principles.

If a recommender system is built for an E-Learning system, it can be used to overcome the information overload problem by filtering out irrelevant learning resources and automatically recommending relevant resources to the learners according to their personalized preferences. In recent years, the Internet has witnessed an exponential growth in the amount of learning resources available online. This explosion of learning resources on the World Wide Web has been precipitated by increasing demand for online learning resources by learners in e-learning environments. However, with this increase in volumes of online learning resources, learners are experiencing difficulties in choosing learning resources that are useful and relevant to their learning needs due to information overload.

* 1. **Introduction to Problem Statement**

On the Internet, where the number of choices is overwhelming, there is need to filter, prioritize and efficiently deliver relevant information in order to alleviate the problem of information overload, which has created a potential problem to many Internet users. Recommender systems solve this problem by searching through large volume of dynamically generated information to provide users with personalized content and services.

The main idea of e-learning recommender systems is to predict the preference or rating of a target learner on a learning object for generating recommendations. Traditional recommender systems such as collaborative filtering (CF) and content-based (CB) have been used in different domains. Recommendation for books in Amazon and movies in Netflix are examples of application areas of recommender systems. Recommender systems typically produce a list of recommendations in one of two ways – through collaborative filtering or through content-based filtering (also known as the personality-based approach). Collaborative filtering approaches build a model from a user's past behaviour (items previously purchased or selected, and/or numerical ratings given to those items) as well as similar decisions made by other users. This model is then used to predict items (or ratings for items) that the user may have an interest in. Content-based filtering approaches utilize a series of discrete characteristics of an item in order to recommend additional items with similar properties. These approaches are often combined to give rise to a Hybrid system.

In the context of e-learning, CF recommend to the target learner learning resources that other similar learners liked in the past. While Content-based recommender systems on the other hand recommend learning resources that are similar in terms of content to the ones that the learner liked in the past. However, previous studies have shown that traditional recommenders suffer from cold-start and rating sparsity problems which limit their performance. The cold-start problem occurs in the recommender system due to an initial lack of ratings for new users who have not rated any item or new items which have not been rated by any user, hence it becomes impossible to make reliable recommendations. On the other hand, sparsity problem occurs in the event the number of users who have rated items is too small compared to the number of items, hence the recommender system cannot generate any recommendations if there is no overlap in ratings with the target user. For this matrix factorization plays an important role in handling the problem related to sparsity. Furthermore, CF and CB on their own are not suitable for e-learning domain since they do not consider the differences in learner characteristics such as learning style, knowledge level as well as learning patterns, resulting in inaccurate recommendations. Although two learners may have similar ratings for a learning resource, their characteristics and historical learning patterns may differ, and this has an impact on the learners’ preference of a learning resource. For accuracy of recommendations, additional learner information needs to be incorporated into the recommendation process alongside the ratings. To address this problem, ontologies and MF can be used to integrate the additional information about the learner into the recommendation process with a view to personalizing the learner profile and recommendations. These additional learner characteristics need to be incorporated into the recommendation process. Recommender systems are a useful alternative to search algorithms since they help users discover items they might not have found otherwise. Of note, recommender systems are often implemented using search engines indexing non-traditional data.

The project goes through phases including requirement analysis, designing a blueprint structure/architecture for the knowledge of the flow of work then moving onto implementation using the necessary frameworks and languages on the specific hardware having pre-requisite installations as per requirements.

Thus, we propose a hybrid knowledge-based recommendation approach based on collaborative filtering, ontology and matrix factorization for recommending learning resources to the learners. MF will be applied to get rid of the sparsity in the Learner vs Learning Resource matrix while ontology will be used to remove the cold start problem and represent knowledge about the learner and learning resources which will be applied to discover the learner’s patterns. Collaborative filtering will be used to compute similarities of ratings and make predictions for the target learner. The main benefit of hybridization of techniques is to take advantage of the strength of each technique while overcoming limitations of individual techniques. Though some of the previous studies have used various techniques in their recommender systems, the novelty of our work is in integrating CF, ontology and MF as a hybrid recommender system.

* 1. **Need of the project**

With the advent of Machine Learning, a field that remains relatively obscure despite having a huge potential in the management and utilization of E-Learning as a resource. This is chiefly due to the inadequate exploration in this field. The unscathed learning resources from the source of abundant provider of unlimited resources, i.e. the Internet need to have a way of finding themselves to a user. The system includes various machine learning algorithms to increase the probability of correct recommendation for a particular user. It suggests the users pertaining the relevance of their ontology. The aspect of knowledge based is regarding the ontology of both the learner and the study material as it adds weight to the object which is important in a personalized recommendation scenario. It often happens that learners suffer from endless list of resources and links over the internet which leaves the person quite baffles. The recommendation system is meant to ease a person from disorderly stacked resources, to quit worrying about searching the right resource rather, letting a resource make itself discoverable by the learner.

By proper implementation, learners will be at a big advantage by getting good recommendations as well as relevancy to their style and distinction. Gone will be the days when learners have to go through endless information and resources before choosing the correct one that suits their learning interest.

Hence, this project becomes important to all learners to use for the technological advancements can be used to largely benefit the community. The community which includes a large student body.

* 1. **Aim of the project**

AIM: The proposed hybrid knowledge-based recommendation approach will be used for recommending learning resources to the learners taking into account the ontology domain knowledge about the learner and learning resources. To achieve this, we must use the appropriate algorithms that will be best suited according to our requirements. The development of a recommendation engine having hybrid properties that provides learners with capabilities such that it gives them ease from search. Search which is an inevitable waste of time can be easily avoided.

Aggregation of ontology for domain knowledge representation and MF to avoid sparsity in matrix of the target learner in the recommendation process. Using ontology domain knowledge and is useful in the personalization of the learner profile and preferences, hence resulting in generation of more accurate recommendations.

**1.4 Objective of the project**

The purpose of recommender systems is to provide recommendations based on recorded information on the users' preferences. These systems use information filtering techniques to process information and provide the user with potentially more relevant items. Our objective is to make the algorithm better by hybridisation of various techniques so as to overcome the drawbacks of using individual techniques.

OBJECTIVES: To achieve the above-mentioned aim we have to perform various tasks: -

1. To understand the importance of Recommendation System.
2. Various techniques used in Recommender Systems.
3. Understanding the existing schemes and their drawbacks.
4. Literature Survey of the various algorithms and learning the importance of each.
5. Understanding the major techniques required in the proposed Hybrid System.
6. Selecting the right techniques and the appropriate features.
7. Presenting the proposed system in detail.
8. Doing the precise requirement analysis for the proposed system.
9. Applying the Recommendation System over an accurate Data set.
10. Analysis and efficiency analysis of the Hybrid Recommender System.
11. To present the difference between the various Recommender Systems individually vs the Hybrid System.

**1.5 Brief Overview**

The next chapters of the preliminary project report consist of valuable information about the progress, working and the idea behind the project.

* The background and literature survey have been explained in detail in Chapter 2, briefly pointing out the history of Recommendation Systems and the techniques used widely.
* Chapter 3 is a breif analysis of the requirements in hardware, software and technologies as well as dataset requirements have been discussed.
* Chapter 4 gives an outline of the architectural design of the proposed model. The recommendation model and hybrid approach has been explained in Chapter 5.
* Chapter 6 presents the experimental results on a real world dataset.
* The codes and Screenshots of dataset and UI constiture Chapter 7.
* In Chapter 8 Conclusion and suggestions for future work are outlined.

**CHAPTER 2**

**BACKGROUND AND LITERATURE SURVEY**

**2.1 History**

We often make choices without sufficient personal experience, and we rely on other people’s recommendations in our everyday life (Resnick P. et al 1997). The earliest systems were traditional information - filtering and information - retrieval systems which could not really recommend rather than giving certain results accordingly to the requests.

Tapestry experimental mail filtering system was introduced by Goldberg D. et al (1992) developed at Xerox Palo Alto Research Centre. Tapestry was designed to deal with both content based and collaborative filtering. The collaborative part then entailed people helping each other by recording their reactions or annotations to documents they had read explicitly. This was the real first recommender system. Recommender systems assist and augment the natural process of decision making. (Resnick P. et al 1997) Recommender systems are widely used in E - commerce, entertainment, content - consumption, and service industry, nowadays on internet services for helping the users finding the items they want and boost commercial benefits for the merchants or service provider. (Ricci, F. et al. 2011) Amazon.com, YouTube, Facebook, E bay, Netflix, iTunes, IMDB and Yelp are some of the big names you may think of when mentioning about recommender systems. In fact, recommender systems play such an important role that Netflix had started a competition for increasing the accuracy of the recommendation their current system could yield. In 2009, they awarded the winning team Belkor’s pragmatic chaos a million - dollar prize over the 10% increase of accuracy the team had offered with their recommender system, which was really a combination of numerous types of algorithms that compensates each other’s disadvantages with their advantages.

* 1. **Recommender system for E-learning**

Application of recommender systems in e-learning domain has become an important research field [1].. The notion of combining recommendation techniques to improve performance has been a growing trend in this field.

“For instance, propose a hybrid recommender system which combines both ontology and genetic algorithm (GA) in making recommendations. Their results show that combining ontology and GA improves the quality of recommendations.

Similarly, Zheng et al. [2]. propose a hybrid trust-based recommendation approach to mitigate learning issues in on-line communities of practice. Their experimental results showed that the hybrid algorithm can provide more accurate recommendations than other related algorithms.

Other studies such as Chen et al. [3]. also proposed a hybrid recommender system to recommend learning materials in an e-learning environment and their results demonstrated significant improvement in performance.

On the other hand, Pukkhem [4]. used ontology in their learning object recommender (LORecommendNet) which can be used to enable machines to interpret and process learning objects in a recommendation system. In their system, ontology was used to map the learner to personalized learning object.

Furthermore, Mota, deCarvalho, and Reis [5]. propose a knowledge-based recommender system supported by an ontological modelling approach to assist educators in designing of teaching and learning activities.

Moreover, Cobos et al. [6]. present a system that allows lecturers to define their best teaching strategies for use in the context of a specific class. Their system namely ‘‘Recommendation System of Pedagogical Patterns’’ (RSPP) is a hybrid system which combines both CB and CF for recommendation and uses ontology for representation of pedagogical patterns.

Takano and Li [7]. on the other hand propose a recommender system for e-learning by utilizing a hybrid feedback method that extracts a user’s preference and Web-browsing behaviour. A hybrid technique was used to monitor explicit feedback for recommended contents and implicit feedback for Web browsing actions associated with user’s preferred contents.

Other related works such as Shaikh and Khoja [8]. used social semantic web and recommender system technologies to integrate teachers’ personal learning environment (PLE-based), learning competencies and learners’ social web and interaction history data to generate personalized recommendations for each learner.

Alimam et al. [9]. propose an ontology-based recommender system for career recommendation in e-learning context while [10]. present a semantic ontology-based recommender system framework to provide personalized e-learning recommendations to assist learners in finding and selecting the relevant learning objects to their field of interest. Their recommendation algorithm is based on the semantic relations and reasoning means in the domain ontology.

Likewise, a hybrid recommendation approach based on the different Matrix Factorization models such as Singular Value Decomposition (SVD), Principal Component Analysis (PCA) and Probabilistic Matrix Factorization (PMF). Their survey shows some good performance of the hybrid algorithm. [11].

Literature review on this area of study has revealed that although many studies on recommendation of e-learning resources have been carried out using various techniques, a more accurate recommendation approach is yet to be realized. Moreover, the literature shows that researchers have attempted to hybridize different techniques with a view to addressing the limitations of traditional recommendation techniques. Our approach is different from previous studies since we aggregate ontology for domain knowledge representation and to capture the learning patterns of the target learner in the recommendation process. Using both ontology domain knowledge and CF is useful in the personalization of the learner profile and preferences, hence resulting in generation of more accurate recommendations.”

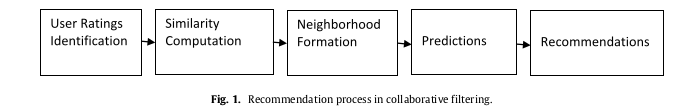
Thus, the above literature survey provides a lot of insight into the recommendation techniques and an attempt to further fuel the development phase of our project.

* 1. **Recommendation Techniques**

**2.3.1 Collaborative filtering**

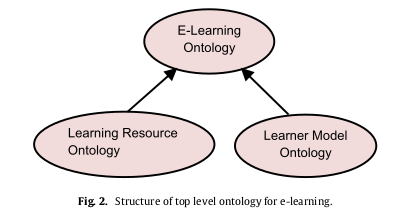
Basic idea behind this algorithm is to provide item recommendations based on the opinions of other like-minded users. The underlying assumption is that if users had similar tastes in the past they will have similar tastes in the future. The similarity in taste of two users is calculated based on the similarity in the rating history of the users [10].. Computation of similarity between users or items is therefore the principle behind CF. It entails searching for other users having similar ratings with the current target user and using those ratings to compute the predictions of the target user; or looking for similar items liked by other users and using the ratings from those similar items to calculate predictions of the target user [13].. The most widely used algorithm for CF is the k Nearest Neighbours (kNN). In our project we have used the Pearson Correlation to find similarity between users.

Fig. 1 summarizes the process of recommendation in CF.



**2.3.2 Hybrid recommender systems**

Hybrid recommender systems combine two or more recommendation techniques [14].. The main purpose of hybridization is to improve the performance of the recommender system as well as overcome the limitations of individual recommendation approaches. There are many methods of hybridizing recommendation techniques. These include weighted, switching, mixed, feature combination, cascade, feature augmentation and meta-level [15].. Limitations associated with traditional recommendation techniques such as cold-start problem, data sparsity and over specialization can be alleviated through hybridization of recommendation techniques [16]..

**2.3.2 Ontology Based recommender system**

Ontology is an explicit specification of a conceptualization. It contains a set of concepts namely entities, attributes and properties related to a domain along with their definitions and relations among them. Domain ontologies can be created manually or automatically, and such ontologies can be integrated with web mining tools. Most ontologies are created using ontology representation languages such as Web Ontology language (OWL) and Resource Description Framework (RDF). Ontologies are beneficial in that it enables reuse of domain knowledge. Reusing ontologies saves time as well as promotes quality ontologies since the ontology components have been well tested previously. Moreover, ontologies can be used alongside other tools and techniques such as data mining and machine learning tools to give better results [17].. Due to the usefulness of ontology as a tool for knowledge representation, it has been widely adopted by researchers in the spheres of information retrieval and recommender systems. In the context of e-learning recommender systems, ontology is used to model the knowledge about the learner and learning resources. Like knowledge-based recommender systems, ontology-based systems do not experience most of the problems associated with traditional recommender systems such as cold-start, data sparsity and over specialization due to use of ontology domain knowledge. Fig. 2 illustrates the structure of the top-level ontology in an e-learning environment.

**2.3.4 Matrix Factorization:**

**CHAPTER 3**

**REQUIREMENTS AND ANALYSIS**

Application of recommender systems in E-learning domain has become an important research field [1].. The notion of combining recommendation techniques to improve performance has been a growing trend in this field. The previous studies have demonstrated that combining techniques in a recommendation process can improve performance of the recommender system. The requirements of the project had to be studied before the implementation. The specification analysis was done, a structured list of requirements was acquired. The requirement specifications determine specific feature expectations, resolution of conflict or ambiguity in requirements as demanded by the various users or groups of users. Requirements analysis includes those tasks that go into determining the needs or conditions to be met for a new or altered product or project, analysing, documenting, validating and managing software or system requirements. Requirements analysis is critical to the success or failure of a systems or software project. The requirements should be documented, actionable, measurable, testable, traceable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design. These features, called requirements, must be quantifiable, relevant and detailed. Requirements analysis is a team effort that demands a combination of hardware, software and human factors engineering expertise. Therefore, here is an explicit assembly of all the requirements of this project that were surveyed at the requirement analysis phase.

**3.1 Software Requirements**

**3.1.1 Frameworks and Languages**

3.1.1.1 Python

Python is a widely used high-level, general-purpose, interpreted, dynamic programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C++ or Java. The language provides constructs intended to enable clear programs on both a small and large scale. Python distribution is available for a wide variety of platforms. You need to download only the binary code applicable for your platform and install Python. If the binary code for your platform is not available, you need a C compiler to compile the source code manually. Compiling the source code offers more flexibility in terms of choice of features that you require in your installation. There are many good reasons to choose Python as your primary programming language. First of all, Python is an easy to learn, powerful programming language. Furthermore, it has efficient high-level data structures, which allow you to write complex operations in fewer statements than in C, C++ or Java. Object-oriented programming is a lot easier than in languages like Java.

Python has become one of the most popular programming languages among developers and programmers. They praise it for its clean syntax and code readability. Python is a general-purpose high-level programming language. Python is both object oriented and imperative and it can be even used in a functional style as well. Python programs are portable, i.e. they can be ported to other operating systems like Windows, Linux, Unix and Mac OS X, and they can be run on Java and .NET virtual machines. Python is very fast. The source code is compiled into bytecode, so that executing the same file will be faster, if the script will be executed again. The bytecode is an "intermediate language", which is said to run on a virtual machine that executes the machine code corresponding to each bytecode.

Python is easy to embed and also better the Python interpreter into C programs. Vice versa, it's possible to extend the Python interpreter by adding a module written in C. One reason to do this is if a C library exists that does something which Python doesn't. Another good reason is if you need something to run faster than you can manage in Python. The Python Standard Library contains an enormous number of useful modules and is part of every standard Python installation. After having learned the essentials of Python, it is necessary to become familiar with the Python Standard Library because many problems can be solved quickly and easily if you are acquainted with the possibilities that these libraries offer.

In our project, we have used the 2.7 version of Python. Since Python 2.7 has abundant libraries and easy to use and easy to install in Linux/UNIX/Windows. Before getting started, we had to find out which IDEs and text editors are tailored to make Python editing easy. Among all Python distributions Anaconda is popular and reliable, having various packages like numpy, matplotlib, pytest, etc, those that were under our requirement specifications.

Details of some of the vital Python packages used in our project, namely-

1. Numpy: is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. It incorporates features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.
2. Matplotlib: is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+. Several toolkits are available which extend matplotlib functionality. Some are separate downloads, others ship with the matplotlib source code but have external dependencies.
3. Sklearn: is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy. It provides in-built methods for testing the accuracy and errors of the models.

3.1.1.2 Anaconda

Anaconda is a free and open source distribution of the Python and R programming languages for data science and machine learning related applications (large-scale data processing, predictive analytics, scientific computing), that aims to simplify package management and deployment. Package versions are managed by the package management system conda, which makes it quite simple to install, run, and update complex data science and machine learning software libraries like Scikit-learn, TensorFlow, and SciPy. Anaconda Distribution is used by over 6 million users, and it includes more than 250 popular data science packages suitable for Windows, Linux, and MacOS. Together with a list of Python packages, tools like editors, Python distributions include the Python interpreter. Anaconda is one of several Python distributions. Anaconda is a new distribution of the Python and R data science package. It was formerly known as Continuum Analytics. Anaconda has more than 100 new packages. This work environment, Anaconda is used for scientific computing, data science, statistical analysis, and machine learning. This package manager is also an environment manager, a Python distribution, and a collection of open source packages and contains more than 1000 R and Python Data Science Packages.

Once you install Anaconda package manager you have access to Anaconda Navigator. Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows launching of applications and easily manageable conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository. It is available for Windows, macOS and Linux. Spyder, Jupyter Notebook, etc are available by default on the Navigator which make coding in Python much easier. Navigator is an easy, point-and-click way to work with packages and environments without needing to type conda commands in a terminal window. You can use it to find the packages you want, install them in an environment, run the packages and update them, all inside Navigator.

Spyder is a powerful interactive development environment for the Python language with advanced editing, interactive testing, debugging and introspection features. We have mainly used Spyder for Python 2.7 because of its ability to execute snippets of code from the editor in the console and continuous parsing of files in editor, and provision of visual warnings about potential errors.

3.1.1.3 Flask

Flask is a micro web framework written in Python and based on the Werkzeug toolkit and Jinja2 template engine. It is BSD licensed. The latest stable version of Flask is 0.12.2 as of May 2017. Applications that use the Flask framework include Pinterest, LinkedIn, and the community web page for Flask itself. “Micro” does not mean that your whole web application has to fit into a single Python file, nor does it mean that Flask is lacking in functionality. The “micro” in microframework means Flask aims to keep the core simple but extensible. Flask has many configuration values, with sensible defaults, and a few conventions when getting started.

Flask is called a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more regularly than the core Flask program.

Python 2.6 or higher is usually required for installation of Flask. Although Flask and its dependencies work well with Python 3 (Python 3.3 onwards), many Flask extensions do not support it properly. Hence, it is recommended that Flask should be installed on Python 2.7.

3.1.1.4 Bootstrap and Ajax

Bootstrap is a free and open-source front-end library for designing websites and web applications. It contains HTML- and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. Unlike many web frameworks, it concerns itself with front-end development only. It has become the most popular front end framework in the recent time. It is sleek, intuitive, and powerful mobile first front-end framework for faster and easier web development. The fast switch over, from the usage of desktops and laptops to tablets and mobile phones that is happening in the digital world has introduced the need for swift adaptation to powerful technological developments.

In this array, Bootstrap is the recent innovation in the field of web design, paving ways for new creations to the web designers. It is the front-end framework that can be developed with JavaScript, HTML and CSS. It is responsive, mobile-first and powerful. It has the option of switching off responsiveness feature to accommodate fixed, non-flexible functionality. Twitter introduced this framework and used it for over a year before Bootstrap became an open source application.

AJAX stands for Asynchronous JavaScript and XML. Ajax is not a single technology, but rather a group of technologies. HTML and CSS can be used in combination to mark up and style information. The built-in XMLHttpRequest object within JavaScript is commonly used to execute Ajax on webpages allowing websites to load content onto the screen without refreshing the page. Ajax is not a new technology, or different language, just existing technologies used in new ways. AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS, and Java Script. Ajax is used to communicate with web pages and web servers. Ajax is used to communicate with web pages and web servers. AJAX is the most viable Rich Internet Application (RIA) technology so far. It is getting tremendous industry momentum and several tool kit and frameworks are emerging. But at the same time, AJAX has browser incompatibility and it is supported by JavaScript, which is hard to maintain and debug. The term Ajax has come to represent a broad group of Web technologies that can be used to implement a Web application that communicates with a server in the background, without interfering with the current state of the page.

Bootstrap makes it easy to design the feature for GUI applications on various devices, and Ajax and jQuery make it fast and interactive.

* + 1. **Operating System**

1. Microsoft Windows 7 or later
2. Fedora Core 10.0 later
3. Ubuntu 14.4 or later
4. MacOS 10.4 or later
   * 1. **Data Set Requirements**

For the accomplishment of this project, the requirement of a data set was summoned. The analysis of the particularity of the data set, the vital features and the extraction from the right data base was important. Much after discussions and analyzations the requirement of this project were listed which includes data set for the learning resources, learner information and the ratings of the learners. The learners’ information and learning resource data was collected into a database. Database usage of each is explained below in detail, the dataset that were fetched out the main database.

Manual collection of the learning resource will be done to accomplish approximately 180 number of resources from PDF notes & audios to visually provisioned video lectures & PPTs. Each resource having a format {text, audio, video, ppt} and level indicating the resource is meant for a beginner, intermediate or advanced level student.

Ontology description of Learning Resource Data set includes the following attributes:

* Learning Resource ID
* Resource Title
* Author
* Type
* Level

Learners data base will be collected through the registration form. The data set will include personal details of the learners, knowledge level {beginner, intermediate, advanced} and the learning style preferred by a learner. To indicate the style of any learner, we have come across a reliable online questionnaire ‘‘Index of Learning Styles Questionnaire’’ [18]. which brings forth the kind of learning done by a student. It includes factors such as active, reflective, sensing, intuitive, visual, verbal, sequential and global. These are powerful indicators of one’s learning styles and capabilities. At the end of the questionnaire the result gets calculated and the learner will have his or her degree of active/reflective, sensing/intuitive, visual/verbal, and sequential/global.

Ontology description of Learner Data set includes the following attributes:

* Learner ID
* Name
* Gender
* Email ID
* Class
* Knowledge Level
* Active/Reflective
* Sensing/intuitive
* Visual/verbal
* Sequential/global

For generating CF recommendations calculation, the learner rating of the resources will be taken. Users will go through resources and rate the resources according to personal relevance. These ratings will be stored in a tabular form as dataset. So, this will comprise of Learner Rating Dataset for the project.

* + 1. **Other Requirements**

1. Mozilla Firefox 3.5, Google Chrome 45.0 or later, Internet Explorer 9 set as your default browser.
2. Java Runtime Environment (JRE) version 5 or later
3. Java SE Development Kit (JDK) version 8 or later
4. TCP/IP port for Python service

**3.2 Hardware Requirements**

* + 1. CPU Speed :

Minimum Requirement - 1GHz;

Pentium or equivalent.

* + 1. RAM

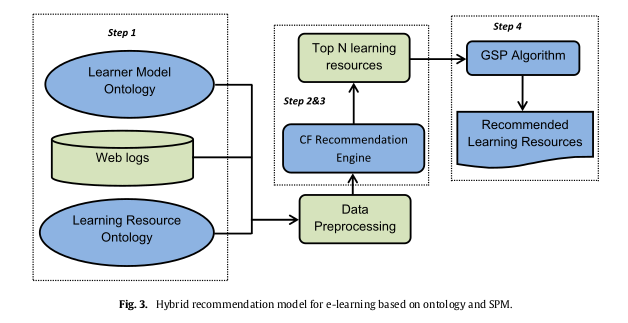
Minimum Requirement – 1 GB (2 GB recommended);

* + 1. Free Disk

Minimum Requirement – 1 GB.

**CHAPTER 4**

**DESIGN**

**4.1 Architecture **

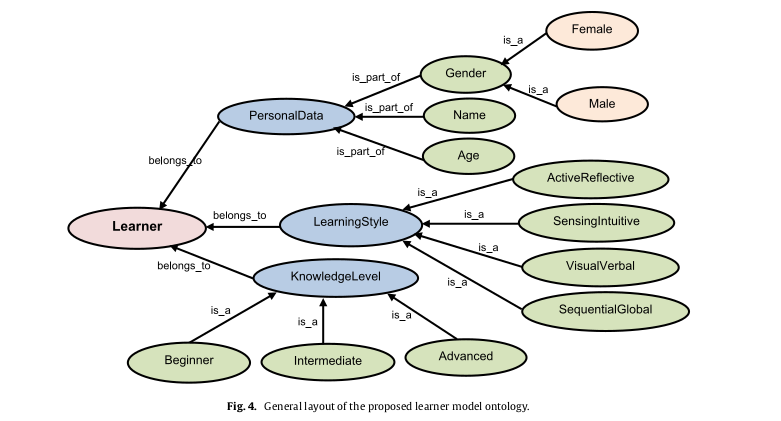
The model contains 5 main components namely the learner model ontology, the learning resource ontology, the recommendation engine, the GSP algorithm and the final recommendations component.

Step1:- Creating an ontology to represent learner and learning resources domain knowledge;

Step 2:- Computing similarities and predictions of ratings for the learner based on the ontology domain knowledge;

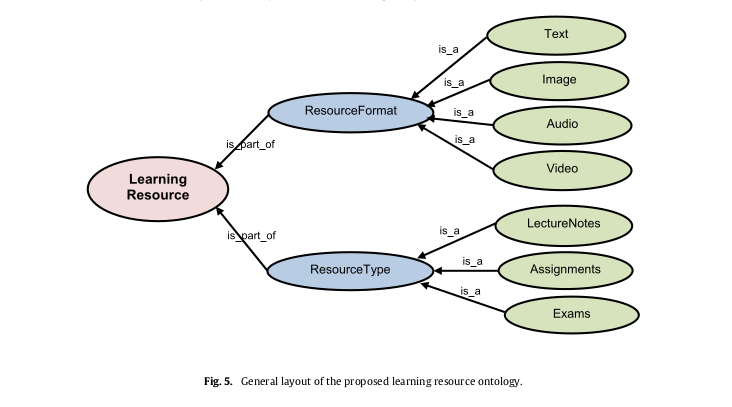
Step 3:- Generation of top N learning resources by the CF recommendation engine; and

Step 4:- Application of GSP algorithm to top N learning resources to generate the final recommendations for the target learner.

**4.2 Learner Ontology Layout**

The learner model ontology represents information about the learner such as demographic data (name, age, gender); learning style, level of education, experience, and knowledge level among others. The lower levels of the learner ontology contain more specific information about the learner. For example, the learner’s learning style would include any of the four dimensions of the Felder–Silverman Learning Style Model namely sequential/global, active/reflective, visual/verbal and sensing/intuitive [18].. Whereas more additional information can be incorporated into the recommendation process to improve personalization of learner recommendations, the downside is increase in utilization of computational resources as well as time complexity. Once the learner’s learning style and knowledge level have been obtained, the learner model ontology is automatically updated. The recommendation engine will make use of this additional information during computation of similarities and recommendations.

**4.3 Learning Resource Ontology Layout**

The learning resource ontology represents knowledge about the learning resources. Knowledge represented in this ontology include learning resource types such as lecture notes, exams and assignments as well as learning resource format which may be text, image, audio or video. In this model, ontology has been used for personalization of learner profile as well as modelling of the learning resource ontology.

**CHAPTER 5**

**IMPLEMENTATION**

The proposed approach is a hybrid knowledge-based recommender system for online learning resources based on ontology and SPM. The main components of the model namely the learner model ontology, the learning resource ontology, the recommendation engine, the GSP algorithm and the final recommendations component. To generate recommendations, our approach involves some major steps as shown in Fig. 3: (1) creating an ontology to represent learner and learning resources domain knowledge; (2) computing similarities and predictions of ratings for the learner based on the ontology domain knowledge; (3) generation of top N learning resources by the CF recommendation engine. The details of the recommendation approach are explained in the following subsections.

**5.1 Creating learner and learning resources ontology**

For this study, the collection of the ontology resources was taken up via questionnaires. Only the learning style and knowledge level were considered for collection as learner characteristics, whereas more axillary data can be incorporated into the recommendation process to refine the personalization of recommendations. Though the downside of it is the increase in computational costs as well as time complexity, since the calculations using these two characteristics were accomplishable we preferred these. To obtain the learner’s learning style, a reliable online questionnaire ‘‘Index of Learning Styles Questionnaire’’ [18]. was regulated to the learner during registration procedure. The Index of Learning Styles is an on-line survey instrument used to assess preferences on four dimensions (active/reflective, sensing/intuitive, visual/verbal, and sequential/global) of a learning style model formulated by Richard M. Felder and Linda K. Silverman. The instrument was developed and validated by Richard M. Felder and Barbara A. Soloman. Users answer 44 a-b questions and submit the survey, and their four preferences are reported back to them immediately.

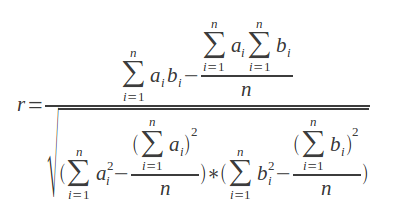
To advocate the knowledge level of the learners, self-assessing scores were administered. The learners score themselves on a scale varying from 1-5 (Beginner = 1-2, Intermediate = 3, Advanced = 4-5). For computational purposes, the different learning styles and knowledge levels are assigned the values {1, 2, 3, 4} and {1, 2, 3} respectively and defined as: Learning style = {active/reflective, sensing/intuitive, visual/verbal, sequential/global} = {1, 2, 3, 4}. Knowledge level is defined as: Knowledge level = {beginner, intermediate, advanced} = {1,2,3}.

Once the learner’s learning style and knowledge level have been obtained, the learner model ontology is automatically updated. The recommendation engine will make use of this additional information during computation of similarities and recommendations. Fig 4 shows the layout of the proposed learner model ontology. On the other hand, when you have a look at Fig 5 the learning resource ontology represents knowledge about the learning resources. Knowledge represented in this ontology include learning resource types (such as lecture notes, exams and assignments) and learning resource format (includes text, image, audio or video). In this model, ontology has been used for personalization of learner profile as well as modelling of the learning resource ontology. The recommendation engine will make use of the learner and learning resources ontology domain knowledge alongside the ratings in computing similarity and predictions for the target learner. Eventually, after the learner and learning resource ontologies have been created, they are prepared and pre-processed alongside the web logs into a format that is required by the CF recommendation engine (Fig. 3). Fig. 5 shows the layout of the proposed learning resource ontology.

**5.2**  **Computing similarities and predictions**

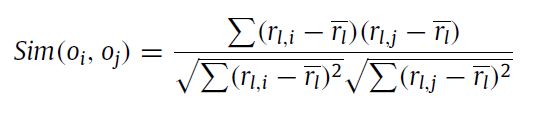
The core of the Collaborative Filtering stage in a Recommender System is the choice of k- nearest neighbours to the learner to whom you wish to make recommendations. This process must be achieved by using similarity measures which make the rightful use of the available information. Our aim is to try to improve results using contextual information (drawn from the entire body of users) as well as categorizing the rating values.

The traditional similarity measures between two users (Pearson correlation, cosine, constrained Pearson correlation, Spearman rank, mean squared differences, etc.) are calculated considering only the ratings made by these two users. In our project we have used the Pearson Correlation formula [21]. for the calculation of similarity between users based on analysing the ratings of each item made by all learners.

 ……. (1)

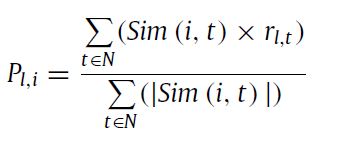
In Eq 1 here, *r* is the similarity coefficient between *a* and *b*, while *ai*the ontological characteristic of learner *a*, and *bi* is the ontological characteristic of learner *b*.

Once ontology domain knowledge about the learner and learning resources as well as the learner ratings on learning resources has been pre-processed, the recommendation engine analyses them and computes the similarities of learning resources as well as predictions for the target learner. In computing the knowledge-based similarities between learners, both the ontology domain knowledge and ratings are considered. To compute ontological similarities, we use an extension of Adjusted Cosine Similarity [20]. Ontological similarity Sim (*oi, oj*) between two learners *i* and *j* is calculated as follows (Eq. (2)):

…… (2)

where *r1,i* is the rating given to learning object *i* by learner *l, rl , j* is the rating given to learning object *j* by learner *l, r l* is the ontological mean rating of all the ratings provided by *l*. Unlike in pure Collaborative Filtering, ontological information is utilized in computing the mean rating *r* in the proposed approach.

The next step is to compute predictions of ratings of the learning objects for the target learner. The predicted ratings are computed from the k most similar learning objects (k nearest neighbours) obtained in (2). The objective is to predict the rating *r l, i* of a target learner *l* for a learning object *i* using the rating given to *i* by learners most similar to *l* (nearest neighbours). To compute predicted rating, we use the following prediction algorithm (Eq. (3)):

 ……….. (3)

where N represents the learning object *i*’s similar learning object set, and *r l, t* is the rating given to learning object *t* by learner *l*.

In the CF model the similarity prediction among learners is done using Pearson Correlation formula in Eq 1. The similarity coefficients generated via this formula are used to get recommendations for the model. In the CF + Onto model, similarity is calculated not only between learners but also among the learning resources. This is where the knowledge-based recommender system exceeds the result evaluation. Similarity for both (learner ontology and learning resource ontology characteristics) is calculated using the Adjusted Cosine Similarity formula in Eq 2. The average of learners’ similarity coefficients received from Pearson Correlation and Adjusted Cosine is calculated and used to get recommendations.

**5.3 Generation of top N list of recommended learning resources**

The top N recommendation list of learning objects is generated by the CF recommendation engine based on the predicted ratings for the target learner and ontology domain knowledge. The recommendation problem in the context of e-learning is defined as the problem of predicting ratings for the learning objects that have not been seen by the learner [19].. In our proposed hybrid recommendation algorithm, the computation of similarity, prediction and recommendations is based on:

• Ratings of the target learner on learning resources.

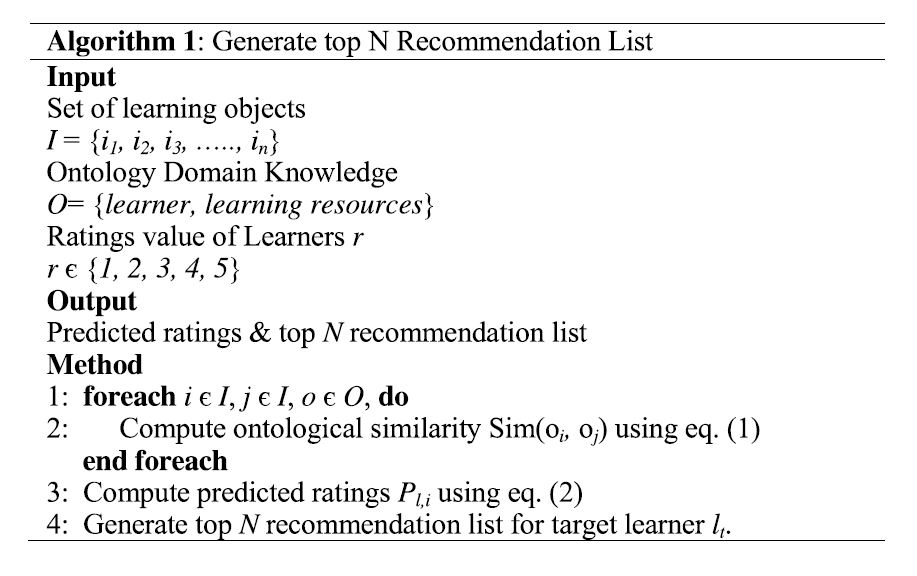
• Ratings given to a learning resource by other learners.

• Ontology domain knowledge of learner and learning resources.

The recommendation list (top N) generated is ranked according to the degree of similarity in ratings of learning objects by the target learner and other learners.

**5.4 Algorithm for searching top-N recommendations**

Algorithm 1 shows how the top N recommendation list is generated. Let *rl,i* be the ratings of learning object *i* by the learner *l* and *Pl,I* be the predicted rating for learning object *i* for the target learner *lt* . The steps for generating the list of recommended learning objects (top N) is shown in Algorithm 1.



**5.5 Final recommendations generation**

The last step is applying the GSP algorithm to the top N recommendation list to filter the recommendations according to the learners’ historical learning sequential patterns. GSP algorithm is a sequence mining algorithm. The Generalized Sequence Pattern algorithm was created from a simpler algorithm for mining sequences, but it has some extra bells and whistles added so it can be more flexible for different situations like Taxonomies/Hierarchies, Min-Gap, Max-Gap, Window Size.

In our work, GSP algorithm was adapted due to its suitability to recommendation of learning materials. GSP algorithm is a sequence mining algorithm that can be used to discover the learners’ sequential learning patterns by mining the learners’ history of rated resources. E-learning recommender system should accurately predict and recommend the learning resource to the target learner based on previous sequential learning pattern.

How the GSP algorithm works in mining the sequence from given sequences is explained in the following three steps:

Pass one: determine the support of each learning object, i.e., the number of data sequences to know which learning objects are frequent.

Candidate sequences generation: generate new potentially frequent sequences (candidate sequences) and determine which of the candidates are actually frequent.

Pruning phase: delete candidate sequences whose support count is less than the minimum support.

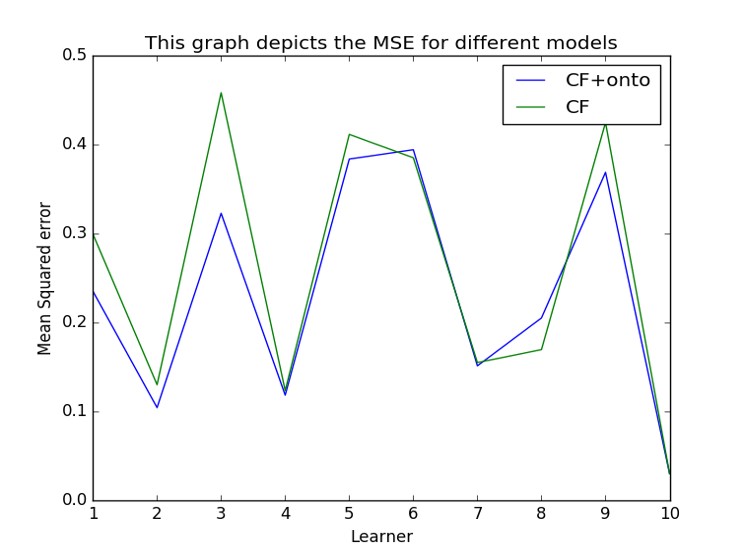
The GSP algorithm was applied to the top N recommendation list. This is a weighted method that applies the GSP algorithm to the results generated from CF with ontology and discovers the sequence learning pattern in the item sets. In the context of recommender systems, some sequences are more important, and others are less important in a sequential pattern. Furthermore, the number of frequent sequential patterns becomes huge as the minimum support becomes lower and vice versa. As a result, it becomes difficult to find more important sequences in a sequential pattern. To overcome this problem, we use a weighted GSP approach where the weights are assigned to items to reflect their relative importance in the sequence. A weight of a learning item *i* is a non-negative real number w that shows the importance of each learning item. Important sequential patterns are generated by giving more weights to items within important sequences. Moreover, weights help in adjusting the number of sequential patterns. Here the weights assigned to reflect importance are ratings given by the learners to them. The advantage of this approach is that each item is assigned a certain weight according to its importance in the sequence, hence even if there are few or more sequences in the initial sequential patterns, only the most important sequences will be used in generating the final recommendations, hence improving the quality of recommendation results.

**CHAPTER 6**

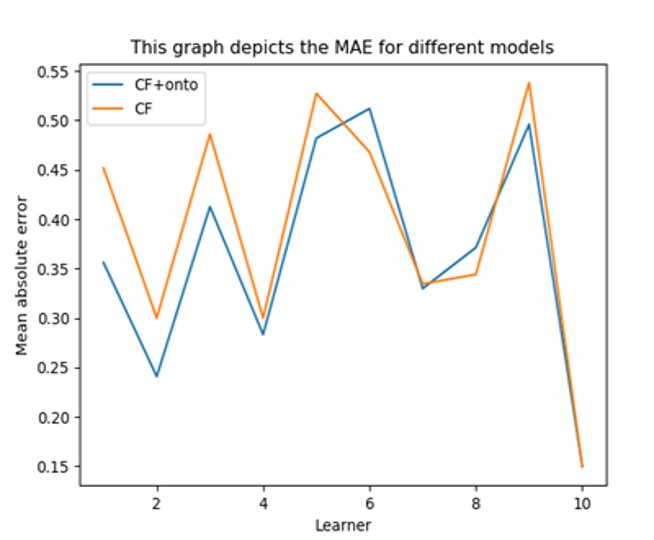
**RESULTS AND EVALUATION**

**6.1 Graphical Comparisons of the models**

**6.1.1 Mean Squared Error for different models**



**6.1.2 Mean Absolute Error for different models**



**CHAPTER 7**

**SNAPSHOTS AND CODE SNIPPETS**

* 1. **Codes**

**7.1.1 Only Collaborative filtering**

def sim\_pearson(prefs,p1,p2):

si={}

for item in prefs[p1]:

if item in prefs[p2]:

si[item]=1

n=len(si)

if n==0: return 0

sum1=sum([prefs[p1][it] for it in si])

sum2=sum([prefs[p2][it] for it in si])

sum1sq=sum([pow(prefs[p1][it],2) for it in si])

sum2sq=sum([pow(prefs[p2][it],2) for it in si])

pSum=sum([prefs[p1][it]\*prefs[p2][it] for it in si])

num=pSum-(sum1\*sum2/n)

den=sqrt((sum1sq-pow(sum1,2)/n)\*(sum2sq-pow(sum2,2)/n))

if den==0: return 0

r=num/den

return r

def topMatches(prefs,person,n=5,similarity=sim\_pearson):

scores=[(similarity(prefs,person,other),other)

for other in prefs if other!=person]

scores.sort()

scores.reverse()

return scores[0:n]

def getRecommendations(prefs,person,similarity=sim\_pearson):

totals={}

simSums={}

for other in prefs:

if other==person: continue

sim=similarity(prefs,person,other)

if sim<=0: continue

for item in prefs[other]:

if item not in prefs[person] or prefs[person][item]==0:

totals.setdefault(item,0)

totals[item]+=prefs[other][item]\*sim

simSums.setdefault(item,0)

simSums[item]+=sim

rankings=[(total/simSums[item],item)

for item,total in totals.items( )]

sub = list()

rating = list()

rankings.sort( )

rankings.reverse( )

for it in rankings:

a,b = it

sub.append(a)

rating.append(b)

return [sub,rating]

**7.1.2 Collaborative filtering + Ontology**

def sim\_cosine(prefs,p1,p2):

si={}

for item in prefs[p1]:

if item in prefs[p2]:

si[item]=1

n=len(si)

if n==0: return 0

sum1sq=sum([pow(prefs[p1][it],2) for it in prefs[p1]])

sum2sq=sum([pow(prefs[p2][it],2) for it in prefs[p2]])

pSum=sum([prefs[p1][it]\*prefs[p2][it] for it in si])

num = pSum

den = sqrt(sum1sq)\*sqrt(sum2sq)

if den==0: return 0

r=num/den

return r

def getOntoRecommendations(pref1s,prefs2, person,similarity1=sim\_pearson, similarity2=sim\_cosine):

totals={}

simSums={}

for other in prefs1:

if other==person: continue

sim1=similarity1(prefs1,person,other)

sim2=similarity2(prefs2,person,other)

sim = (sim1+sim2)/2

if sim<=0: continue

for item in prefs1[other]:

if item not in prefs1[person] or prefs1[person][item]==0:

totals.setdefault(item,0)

totals[item]+=prefs1[other][item]\*sim

simSums.setdefault(item,0)

simSums[item]+=sim

rankings=[(total/simSums[item],item)

for item,total in totals.items( )]

rankings.sort( )

rankings.reverse( )

return rankings

for it in rankings:

a,b = it

sub.append(a)

rating.append(b)

return [sub,rating]

**7.1.3 Flask implementation**

from flask import Flask, render\_template, request, jsonify, redirect, url\_for, make\_response

import os

from math import sqrt

app = Flask(\_\_name\_\_)

@app.route('/')

def home():

return render\_template('home.html')

@app.route('/run', methods=['POST'])

def run\_file():

prefs = dict()

prefs = loading250()

user = request.form['name']

if user:

print user

sub = list()

rating = list()

rating,sub = getRecommendations(prefs, user)

print 'CF Recommendations:'

print sub

return jsonify({

'result1':sub[0],

'result2':sub[1],

'result3':sub[2],

'result4':sub[3],

})

return jsonify({'error':'mising data'})

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**7.1.4 Graph plot of results**

import numpy as np

import matplotlib.pyplot as plt

rows=[]

from sklearn.metrics import mean\_absolute\_error

def loading():

import csv

data1=csv.reader(open('act\_pred\_model2.csv','rb'))

user,y\_true,y\_pred =[],[],[]

error1 = []

data1.next()

user\_now = 'L1'

for row in data1:

user.append(row[0])

l = len(user)

if user[l-1] == user\_now:

y\_true.append(float(row[3]))

y\_pred.append(float(row[2]))

else:

error1.append(mean\_absolute\_error(y\_true,y\_pred))

y\_true,y\_pred =[],[]

y\_true.append(float(row[3]))

y\_pred.append(float(row[2]))

l = len(user)

user\_now = user[l-1]

error1.append(mean\_absolute\_error(y\_true,y\_pred))

avg1=np.sum(error1)/10

print "Average MAE for model 2"

print avg1

print "MAE for model 2"

print error1

data=csv.reader(open('act\_pred\_250.csv','rb'))

user,y\_true,y\_pred =[],[],[]

error = []

data.next()

user\_now = 'L1'

for row in data:

user.append(row[0])

l = len(user)

if user[l-1] == user\_now:

y\_true.append(float(row[3]))

y\_pred.append(float(row[2]))

else:

error.append(mean\_absolute\_error(y\_true,y\_pred))

y\_true,y\_pred =[],[]

y\_true.append(float(row[3]))

y\_pred.append(float(row[2]))

l = len(user)

user\_now = user[l-1]

error.append(mean\_absolute\_error(y\_true,y\_pred))

avg2=np.sum(error)/10

print "\nAverage MAE for model1"

print avg2

print "MAE for model1"

print error

arr=[1,2,3,4,5,6,7,8,9,10]

plt.xlabel('Learner')

plt.ylabel('Mean absolute error')

plt.title('This graph depicts the MAE for different models')

plt.plot(arr,error1,label='CF+onto')

plt.plot(arr,error,label='CF')

#plt.plot(arr,error2,label='250 ratings')

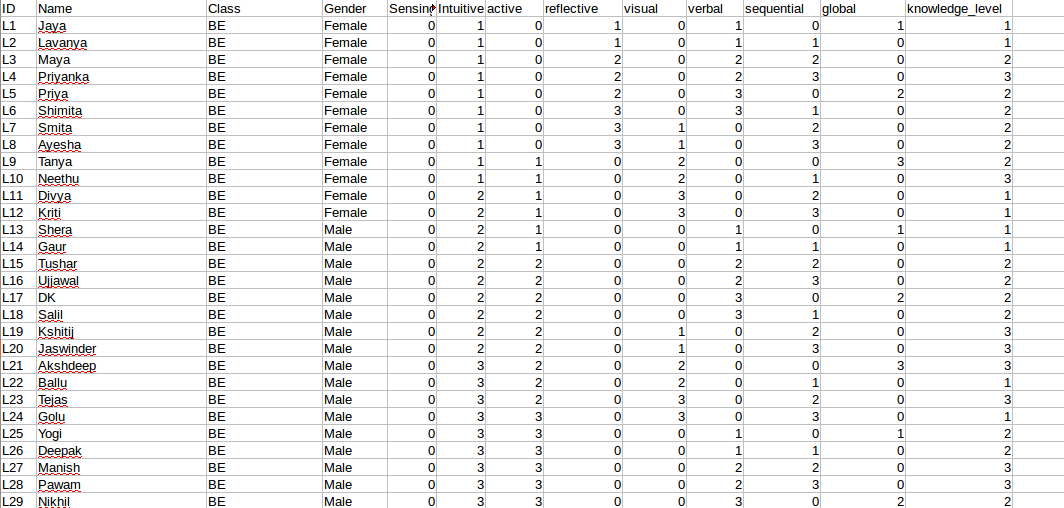
plt.legend()

plt.show()

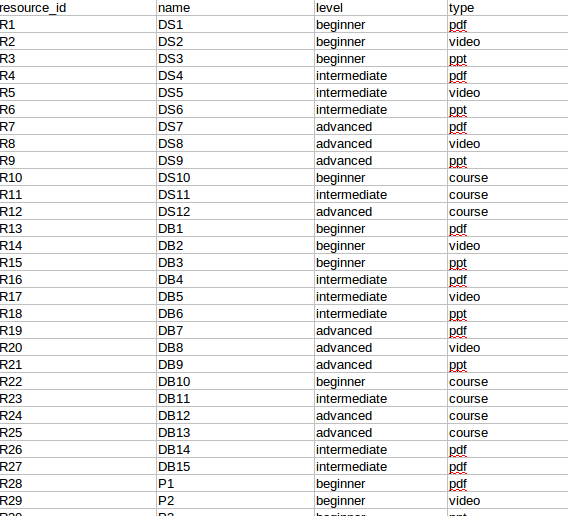
loading()

* 1. **Snapshots**

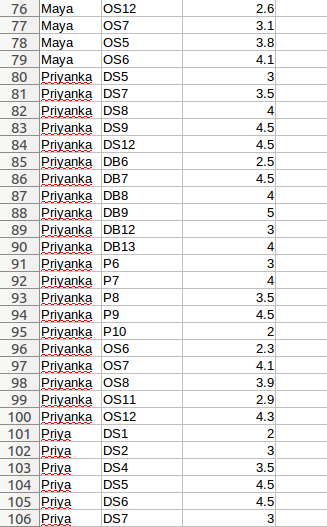
**7.2.1 Data Set Snapshot**

7.2.1.1 Learner Ontology

7.2.1.2 Learning Resource Ontology



7.2.1.3 Learners’ Ratings Table

****

**CHAPTER 9**

**CONCLUSION**

First, we have proposed a hybrid knowledge-based recommendation approach for recommending learning resources to the learners considering the ontology domain knowledge about the learner and learning resources as well as the learners’ historical sequential learning patterns. Aggregation of this additional information into the recommender system will result in generation of recommendations that are more personalized to the learner.

Secondly, in computing similarity of learners and generating predictions, ontology domain knowledge is considered alongside the ratings, hence improving the accuracy of predictions.

Thirdly, the proposed hybrid knowledge-based approach addresses the cold-start problem by making use of available ontology domain knowledge arising from integration of ontology into the recommendation process. Additionally, it can alleviate sparsity problem by making use of the learner’s sequential access patterns to predict learner’s preferred learning resources in cases where their ratings are sparse.

Lastly, our recommendation approach can outperform those recommendation methods that do not incorporate ontology and learners’ sequential learning patterns in their recommendation process due to hybridisation.

**ANNEXURE A**

**REFERENCES**

1. N. Manouselis, H. Drachsler, R. Vuorikari, H. Hummel, R. Koper, Recommender systems in technology enhanced learning, in: Recomm. Syst. Handb, Springer, US, 2011, pp. 387–415
2. X.L. Zheng, C.C. Chen, J.L. Hung, W. He, F.X. Hong, Z. Lin, A hybrid trust-based recommender system for online communities of practice, IEEE Trans. Learn. Technol. 8 (2015) 345–356.
3. W. Chen, Z. Niu, X. Zhao, Y. Li, A hybrid recommendation algorithm adapted in e-learning environments, World Wide Web 17 (2014) 271–284.
4. N. Pukkhem, LORecommendNet: An ontology-based representation of learning object recommendation, Adv. Intell. Syst. Comput. 265 (2014) 293–303. AISC.
5. D. Mota, C.V. de Carvalho, L.P. Reis, OTILIA—An architecture for the recommendation of teaching-learning techniques supported by an ontological approach, in: 2014 IEEE Front. Educ. Conf. Proc., 2014, pp. 1–7.
6. C. Cobos, O. Rodriguez, J. Rivera, J. Betancourt, M. Mendoza, E. León, E. Herrera Viedma, A hybrid system of pedagogical pattern recommendations based on singular value decomposition and variable data attributes, Inf. Process. Manag. 49 (2013) 607–625.
7. K. Takano, K.F. Li, An adaptive e-learning recommender based on user’s web- browsing behaviour, in: Proc. - Int. Conf. P2P, Parallel, Grid, Cloud Internet Comput. 3PGCIC 2010, 2010, pp. 123–131.
8. Z.A. Shaikh, S.A. Khoja, Towards guided personal learning environments: Concept, theory, and practice, in: Proc. - IEEE 14th Int. Conf. Adv. Learn. Technol. ICALT 2014, Athens, Greece, 2014, pp. 782–784.
9. M.A. Alimam, H. Seghiouer, Y. El Yusufi, Building profiles based on ontology for career recommendation in E-Ieaming context, in: Int. Conf. Multimed. Comput. Syst. -Proceedings, 2014, pp. 558–562.
10. S. Fraihat, Q. Shambour, A framework of semantic recommender system for e- learning, J. Softw. 10 (2015) 317–330.
11. Dheeraj Bokde, Sheetal Girase, Debajyoti Mukhopadhyay, Matrix Factorization Model in Collaborative Filtering Algorithms: A Survey
12. J. Schafer, D. Frankowski, J. Herlocker, S. Sen, Collaborative filtering recommender systems, in: Adapt. Web., 2007, pp. 291–324.
13. L. Yao, Q.Z. Sheng, A.H.H. Ngu, J. Yu, A. Segev, Unified collaborative and content-based web service recommendation, IEEE Trans. Serv. Comput. 8(2015) 453–466.
14. R. Burke, Hybrid web recommender systems, in: Adapt. Web, 2007, pp. 377–408.
15. R. Burke, Hybrid recommender systems: Survey and experiments, User Modell. User Adapt. Interact. 12 (2002) 331–370.
16. M.A. Ghazanfar, Experimenting switching hybrid recommender systems, Intell. Data Anal. 19 (2015) 845–877.
17. B. Amini, R. Ibrahim, M. Shahizan, M. Ali, Expert Systems with Applications A reference ontology for profiling scholar ’ s background knowledge in recommender systems, Expert Syst. Appl. 42 (2015) 913–928.
18. B.A. Soloman, N. Carolina, R.M. Felder, Index of learning styles questionnaire, Learning (1996) 1–5.
19. A.S. Lampropoulos, G.A. Tsihrintzis, Machine Learning Paradigms: Applications in Recommender Systems, 2015.
20. L. Xinyi, S. Hailong, W. Hanxiong, Z. Richong, L. Xudong, Using sequential pattern mining and interactive recommendation to assist pipe-like mashup development, in: Proc. - IEEE 8th Int. Symp. Serv. Oriented Syst. Eng. SOSE 2014, 2014, pp. 173–180.
21. http://mines.humanoriented.com/classes/2010/fall/csci568/portfolio\_exports/sphilip/pear.html