Towards partial fulfilment for Undergraduate Degree level Programmed
Bachelor of Technology in Computer Engineering

A Project Report on:

'IMPLEMENTATION OF WEB BASED RECOMMENDATION SYSTEMS'

Prepared by:

Admission No.	Student Name
U12CO056	ABHISHEK KUMAR
U12CO074	RAGHUNATHA RAO GC
U12CO098	D. M. T. B. DISSANAYAKE
U12CO065	HITESH GOYAL

Class	:	B.TECH. IV Computer Engineering
Year	:	2015 - 2016
Guided by	:	Dr. U. P. RAO

DEPARTMENT OF COMPUTER ENGINEERING

SARDAR VALLABHBHAI NATIONAL INSTITUTE OF TECHNOLOGY

SURAT – 395 007 (GUJARAT, INDIA)

Student Declaration

This is to certify that the work described in this project report has been actually carried out and implemented by our project team consisting of

Sr.	Admission No.	Student Name
1	U12CO056	ABHISHEK KUMAR
2	U12CO074	RAGHUNATHA RAO G C
3	U12CO098	D.M.T.B DISSANAYAKE
4	U12CO065	HITESH GOYAL

Neither the source code there in, nor the content of the project report have been copied or downloaded from any other source. We understand that our result grades would be revoked if later it is found to be so.

Signature of the Students:

Sr.	Student Name	Signature of the Student
1	ABHISHEK KUMAR	
2	RAGHUNATHA RAO G C	
3	D.M.T.B DISSANAYAKE	
4	HITESH GOYAL	

Certificate

*This is to certify that the project entitled '**Implementation of Web Based Recommendation Systems** 'is prepared and presented by*

Sr.	Admission No.	Student Name
1	U12CO056	ABHISHEK KUMAR
2	U12CO074	RAGHUNATHA RAO G C
3	U12CO098	D.M.T.B DISSANAYAKE
4	U12CO065	HITESH GOYAL

Final Year of Computer Engineering and their work is satisfactory.

SIGNATURE:

GUIDE JURY HEAD OF DEPT.

Abstract

Due to evolution of Internet and connectivity in world humans have endless possibilities such as increased websites on internet, more choices of music to listen, explosion of social networking. This revolution has just disturbed humans as now they have to select from a humungous bag. To address this major issue Recommender Systems have been designed by the computer scientists. The Recommender Systems are engines which interact with large scale complex information spaces to provide a personalized view of such spaces, prioritizing items likely to be of interest to the user. Recommendation Systems have grown enormously in the variety of problems and their techniques are deployed in many practical applications. This wealth of practical application studies has provided inspiration to researchers to extend the reach of Recommender Systems into new and challenging areas. The report discusses the implementation of the Content Based Recommendation System based on the different feature sets, and a Collaborative Filtering Recommendation System by user based clustering.

Keywords: Items; Users; Content-based; Collaborative-filtering; Datasets; Clustering; Classification; Cold start problem;

Table of Contents

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

RS:	Recommendation Systems
CF:	Collaborative Filtering
PCA:	Principal Component Analysis
SVD:	Singular Value Decomposition
kNN:	k Nearest Neighbour
VSM:	Vector Space Model
RMSE:	Root Mean Square Error

CHAPTER I

INTRODUCTION

The world has witnessed explosion of information and entertainment technologies, and thus choices of a person in the past decade. Today people choose from dozens to hundreds of television channels, thousands of videos, millions of books, CDs, and multimedia, interactive documents on the World Wide Web, and seemingly countless other consumer items presented in catalogues or advertisements in one medium or another. The web in particular offers various possibilities- in addition to interactive documents, there are conversation to join, news feeds to read, items to purchase, music and movies to stream and so on. Not only is there a vast number of possibilities, but they vary widely in quality and quantity. Evaluating all these alternatives, however, still takes about the same time and effort it always has. Therefore, individuals cannot hope to evaluate all available choices by themselves unless the topic of interest is severely constrained.

So what can we do? When people have to make a choice without any personal knowledge of the alternatives, a natural course of action is to rely on the experience and opinions of others. We seek recommendations from people who are familiar with the choices we face, who have been helpful in the past, whose perspectives we value, or who are recognized experts. We might turn to friends or colleagues, the owner of a neighbourhood bookstore, movie reviews in a newspaper or magazine, or Consumers Union product ratings. And we may find the social process of meeting and conversing with people who share our interests as important as the recommendations we receive.

Today increasing numbers of people are turning to computational recommender systems. Emerging in response to the technological possibilities and human needs created by the World Wide Web, these systems aim to mediate, support, or automate the everyday process of sharing recommendations. This has led the industry to build their own recommendation engines which provide personalised items to their users. The main goal is to identify challenges and suggest new opportunities. We begin by developing a conceptual framework recommender systems that builds on everyday examples of recommendation and identifies basic concepts and design issues. The bulk of this introduction is devoted to surveying the need for recommendation systems in this present new digital world.

Everyone can witness the examples of recommendation. You might think of reading movie reviews in a magazine to decide which movie to see. Or you might recall visits to your local bookstore, where you've talked to the owner about your interests and current mood, and she then recommended a few books you'd probably like. Finally, you might remember walking through your neighbourhood and noticing that a particular sidewalk cafe always is crowded. You think that its popularity must be a good sign, so you decide to give it a try. Reflecting on these examples helps to clarify the concept of recommendation. A person is faced with a decision, which for our purposes is a choice among a universe of alternatives. If the person doesn't have sufficient personal knowledge to make the choice, he or she may seek recommendations from others. Recommendation, therefore, is a communicative act.

Recommendation is based on the preferences of the recommender. Generally, a preference is an individual mental state concerning a subset of items from the universe of alternatives. Individuals form preferences based on their experience with the relevant items, such as listening to music, watching movies, tasting food, etc. Of course, preferences can be more wide and complicated: I might prefer one item over another (The Game of Thrones over The Big Bang Theory) or even think in terms of some scoring system ("on a scale of 1 to 10, a movie which has been visited twice is 10").

A recommendation may be directed to specific individuals or "broadcast" to anyone who's interested. For the person who receives it, a recommendation is a resource that helps in making a choice from the universe of alternatives. Thus recommendation serves as a view or filter onto the whole. A recommendation may be based not just on the recommender's preferences, but also on those of the recommendation seeker. For example, in recommending books to someone, we might find out which genres he/she like (e.g., Science Fiction, Mystery or Romance) and even which books he/she have really enjoyed (e.g., Suzanne Collins's Hunger Games trilogy). We then can recommend books that are both good (in my opinion) and will meet his/her preferences. We even can recommend books based on the preferences of others: maybe we are not science fiction fans, but we have friends that are, so we can make recommendations based on what they like. Further, we may put you in touch with people who share your interests: maybe there's a Science Fiction reading group you might like to join. Finally, a recommendation may include explanatory material that helps the recommendation seeker evaluate it (why you'd like the Hunger Games trilogy, what's good about The Divergent, and why it's better than The Twilight, etc.).

We must distinguish between the roles played by the recommendation system on behalf of the service provider from that of the user of the Recommendation System. For instance, a travel recommender system is typically introduced by a travel intermediary (e.g., Expedia.com) or a destination management organization (e.g., incredibleindia.com) to increase its turnover (Expedia), i.e., sell more hotel rooms, or to increase the number of tourists to the destination. Whereas, the user's primary motivations for accessing the two systems is to find a suitable hotel and interesting events/attractions when visiting a destination. In fact, there are various reasons as to why service providers may want to exploit this technology [1]:

- *Increase the number of items sold.* This is probably the most important function for a commercial recommendation system, i.e., to be able to sell an additional set of items compared to those usually sold without any kind of recommendation.
- *Sell more diverse items.* Another major function of a recommendation system is to enable the user to select items that might be hard to find without a precise recommendation. For instance, in a movie recommendation system such as Netflix, the service provider is interested in renting all the DVDs in the catalogue, not just the most popular ones.
- *Increase the user satisfaction.* A well designed recommendation system can also improve the experience of the user with the site or the application. The user will find the recommendations interesting, relevant and, with a properly designed human-computer interaction, she will also enjoy using the system.
- *Increase user fidelity.* A user should be loyal to a Web site which, when visited, recognizes the old customer and treats him as a valuable visitor. This is a normal feature of a recommendation system since many recommendation systems compute recommendations, leveraging the information acquired from the user in previous interactions, e.g., her ratings of items.
- *Better understand what the user wants.* Another important function of a recommendation system is to use the description of the user's preferences, either collected explicitly or predicted by the system. The service provider may then decide to re-use this knowledge for a number of other goals such as improving the management of the item's stock or production.

We mentioned above some important motivations as to why e-service providers introduce Recommendation Systems. But users also may want a Recommendation Systems, if it will effectively support their tasks or goals. Consequently a Recommendation Systems must balance the needs of these two players and offer a service that is valuable to both. Core tasks that are normally associated with a Recommendation Systems is to offer suggestions for items that may be useful to a user [1].

- *Find Some Good Items:* Recommend to a user some items as a ranked list along with predictions of how much the user would like them (e.g., on a 1-5 star scale). This is the main recommendation task that many commercial systems address.
- *Find all good items:* Recommend all the items that can satisfy some user needs. In such cases it is insufficient to just find some good items. In these situations, in addition to the benefit derived from carefully examining all the possibilities, the user may also benefit from the Recommendation Systems ranking of these items or from additional explanations that the Recommendation Systems generates.
- *Annotation in context:* Given an existing context, e.g., a list of items, emphasize some of them depending on the user’s long-term preferences. For example, a TV recommender system might annotate which TV shows displayed in the electronic program guide are worth watching.
- *Recommend a sequence:* Instead of focusing on the generation of a single recommendation, the idea is to recommend a sequence of items that is pleasing as a whole. Typical examples include recommending a TV series; a book on Recommendation Systems after having recommended a book on data mining; or a compilation of musical tracks.
- *Recommend a bundle:* Suggest a group of items that fits well together. For instance a travel plan may be composed of various attractions, destinations, and accommodation services that are located in a delimited area. From the point of view of the user these various alternatives can be considered and selected as a single travel destination.
- *Just browsing:* In this task, the user browses the catalogue without any imminent intention of purchasing an item. The task of the recommender is to help the user to browse the items that are more likely to fall within the scope of the user’s interests for that specific browsing session.
- *Find credible recommender:* Some users do not trust recommender systems thus they play with them to see how good they are in making recommendations. Hence, some

system may also offer specific functions to let the users test its behaviour in addition to those just required for obtaining recommendations.

- *Improve the profile:* This relates to the capability of the user to provide (input) information to the recommender system about what he likes and dislikes. This is a fundamental task that is strictly necessary to provide personalized recommendations. If the system has no specific knowledge about the active user then it can only provide him with the same recommendations that would be delivered to an “average” user.
- *Express self:* Some users may not care about the recommendations at all. Rather, what it is important to them is that they be allowed to contribute with their ratings and express their opinions and beliefs. The user satisfaction for that activity can still act as a leverage for holding the user tightly to the application.
- *Help others:* Some users are happy to contribute with information, e.g., their evaluation of items (ratings), because they believe that the community benefits from their contribution. This could be a major motivation for entering information into a recommender system that is not used routinely.
- *Influence others:* In Web-based RSs, there are users whose main goal is to explicitly influence other users into purchasing particular products. As a matter of fact, there are also some malicious users that may use the system just to promote or penalize certain items.

1.1 APPLICATIONS OF RECOMMENDATION SYSTEMS

Recommender systems are now popular both commercially and in the research community, where many approaches have been suggested for providing recommendations. Many companies are using recommendation systems in their products to attract the users. The suggestions relate to various decision-making processes, such as what items to buy, what music to listen to, or what online news to read.

Recommender System research is being conducted with a strong emphasis on practical and commercial applications, since, aside from its theoretical contribution, is generally aimed at practically improving commercial Recommendation Systems. Thus, Recommendation Systems research involves practical aspects that apply to the implementation of these systems. These aspects are relevant to different stages in the life cycle of a Recommendation Systems, namely, the design of the system, its implementation and its maintenance and enhancement during system operation. The aspects that apply to the design stage include factors that might affect the choice of the algorithm. The first factor to consider, the application’s domain, has a major effect on the algorithmic approach that should be taken. Based on these specific application domains, we define more general classes of domains for the most common recommender systems applications [1]:

- **Entertainment:** Recommendations for movies, music, and IPTV. The recommendation systems are responsible for the users of these systems to provide appropriate entertainment items using recommendation principles. The user is recommended the highly ranked item accordingly. MovieLens, Ringo, Bellcore Video recommendation, IMBD, Apple Music Store, Youtube are some examples which are equipped with recommendation systems which provide recommendations.
- **Content**: Personalized newspapers, recommendation for documents, recommendations of Web pages, e-learning applications, and e-mail filters. Here, the recommendation systems are useful to filter the streams of data according to user needs. Webpages, E-mails and web data are filtered according to users experience and is posted for him.
- **E-commerce**: Recommendations for consumers of products to buy such as books, cameras, PCs etc. These systems help the users in purchasing the appropriate products which provides maximum benefit to both users and the vendors. Flipkart, Amazon, WebSell, CDNow are some examples which consumers can rely on purchasing items.
- **Services**: Recommendations of travel services, recommendation of experts for consultation, recommendation of houses to rent, or matchmaking services. The recommendation systems are ubiquitously used by many different companies to promote or recommend their services or products to users. MakemyTrip, Commonfloor.com, Expedia.com, Bharatmatrimony.com are some examples which use recommendation systems to provide their services to the users.

1.2 MOTIVATION

In today’s world, due to exploitation of information and revolution of internet has provided a universe of choices to the people. This created a trouble for many people in choosing their desired items and this is fascinating to see how these recommendation systems are helping users to take best out of them.

1.3 OBJECTIVES

- Identifying different types of Recommendation Systems.
- Identifying different Algorithms to build Recommendation Systems.
- Designing the architecture of the Recommendation System.
- Implementing and concluding the results of Algorithms.

1.4 CONTRIBUTION

We have explored the background and literature behind the Recommendation System. We have identified different types of recommendation systems and decided to concentrate on content-based and collaborative-filtering systems as these are the models that are generally implemented in the world [1]. Also, we have identified different techniques and algorithms to build up the recommendation system.

We have implemented the recommendation system for movies in both Content Based and Collaborative Filtering techniques. The content based system uses genre feature set (i.e. Comedy, Action, Romance, etc…) for predicting the recommendations for the user. The technique has been discussed and implemented on the Movie Lens dataset [6] and the results and its efficiency are discussed in the report. The Collaborative Filtering technique is based on clustering by K means and similarity calculation using Pearson and Cosine similarity measures.

1.5 ORGANISATION OF REPORT

Chapter 2 discusses the literature and the background necessary for the recommendation systems. All the architecture, different types and the algorithms necessary are discussed in this chapter. Chapter 3 discusses the implementation of content based recommendation systems using feature sets in the dataset i.e., Genre (feature set), results and analysis of this system is discussed here. Chapter 4 discusses the implementation of the collaborative filtering recommender systems and the results on this system have been summarised. Chapter 5 discusses the conclusion of both the systems developed in chapter 3 and 4 and advises some future work for further exploration.

CHAPTER II

LITERATURE SURVEY

In this section, we are going to discuss the Architecture and Different types of Algorithms necessary for the Recommendation System. All these aspects are illustrated below.

Recommender Systems are software tools and techniques which provide suggestions for items which are used to a user. These suggestions relate to various decision-making processes, such as what items to buy, what music to listen to, or what online news to read. For example a book recommender system like Amazon.com assists users to select a book to read. This site employs a recommendation system to personalize the online store for each customer. Since recommendations are usually personalized, different users or user groups receive diverse suggestions. In addition there are also non-personalized recommendations. These are much simpler to generate and are normally featured in magazines or newspapers. Typical examples include the top ten selections of books, CDs etc. While they may be useful and effective in certain situations, these types of non-personalized recommendations are not typically addressed by Recommendation System researchers.

Ultimately a Recommendation System addresses this phenomenon by pointing a user towards new, not-yet-experienced items that may be relevant to the user's current task. Upon a user's request, depending on the recommendation approach, by the user's context and need, Recommendation System generate recommendations using various types of knowledge and data - users, the available items, and previous transactions stored in customized databases. The user can then browse the recommendations. She/he may accept them or not and may provide, an implicit or explicit feedback immediately or at a next stage. All these user actions and feedback scan be stored in the recommender database and may be used for generating new recommendations in the next user-system interactions.

Data and Knowledge Sources

Recommendation Systems are information processing systems that actively gather various kinds of data in order to build their recommendations. Data is primarily about the items to suggest and the users who will receive these recommendations. Since the data and knowledge sources available for recommender systems can be very diverse, ultimately, it depends on recommendation technique whether they can be exploited or not.

In any case, data used by RSs refers to three kinds of objects: *items*, *users*, and *transactions* (i.e., relations between users and items) [1].

Items: Items are the objects that are recommended. Items may be characterized by their value or utility. The value of an item may be positive if the item is useful for the user, or negative if the item is not appropriate and the user made a wrong decision when selecting it. We note that when a user is acquiring an item she/he will always incur in a cost, which includes the cognitive cost of searching for the item and the real monetary cost eventually paid for the item.

For example in a movie recommender system, the genre (such as comedy, thriller, etc.), as well as the director, and actors can be used to describe a movie and to learn how the utility of an item depends on its features. Items can be represented using various information and representation approaches, e.g., in a minimalist way as a single id code, or in a richer form, as a set of attributes.

Users: In order to personalize the recommendations, Recommendation engines exploit a range of information about the users. This information can be structured in various ways and again the selection of what information to model depends on the recommendation technique.

Users can also be described by their behaviour pattern data, for example, site browsing patterns in a Web-based recommender system, or travel search patterns in a travel recommender system. Moreover, user data may include relations between users such as the trust level of these relations between users. A Recommendation Systems might utilize this information to recommend items to users that were preferred by similar or trusted users.

Transactions: We generically refer to a transaction as a recorded interaction between a user and the RS. Transactions are log-like data that store important information generated during the interaction and which are useful for the recommendation generation algorithm that the system is using. For instance, a transaction log may contain a reference to the item selected by the user and a description of the context (e.g., the user goal/query) for that particular recommendation. If available, that transaction may also include an explicit feedback the user has provided, such as the rating for the selected item.

In fact, ratings are the most popular form of transaction data that a RS collects. These ratings may be collected explicitly or implicitly. In the explicit collection of ratings, the user is asked to provide her opinion about an item on a rating scale. Ratings can take on a variety of forms [1]:

• *Numerical ratings* such as the 1-5 stars provided in the book recommender associated with Amazon.com.

• *Ordinal ratings*, such as “strongly agree, agree, neutral, disagree, strongly disagree” where the user is asked to select the term that best indicates her opinion regarding an item.

• *Binary ratings* that model choices in which the user is simply asked to decide if a certain item is good or bad.

• *Unary ratings* can indicate that a user has observed or purchased an item, or otherwise rated the item positively.

In transactions collecting implicit ratings, the system aims to infer the users opinion based on the user’s actions. For example, if a user enters the keyword “Yoga” at Amazon.com she will be provided with a long list of books. In return, the user may click on a certain book on the list in order to receive additional information. At this point, the system may infer that the user is somewhat interested in that book.

2.1 MODEL OF RECOMMENDATION SYSTEM

The generic recommendation process is summarized in Fig 1. A recommendation seeker may ask for a recommendation, or a recommender may produce recommendations with no prompting. Seekers may volunteer their own preferences, or recommenders may ask about them. Based on a set of known preferences – his/her own, the seeker’s, and those of other people, often people who received recommendations in the past – the recommender recommends items the seeker probably will like. In addition, the recommender may identify people with similar interests. The seeker may use the recommendation to select items from the universe or to communicate with like-minded others.

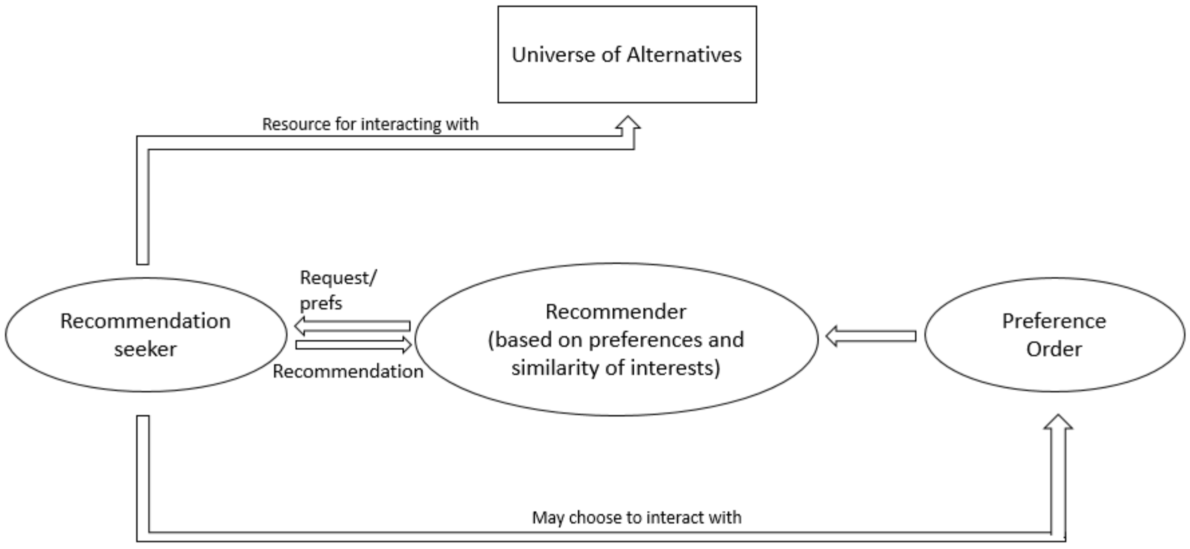

Universe of Alternatives
Resource for interacting with
Request/ prefs
Recommendation seeker
Recommendation
Recommender (based on preferences and similarity of interests)
Preference Order
May choose to interact with

Figure 1 Generic Recommendation process [3]

2.2 ISSUES FOR RECOMMENDATION SYSTEMS

A recommender system automates or supports part of the recommendation process. An automated recommender system assumes the recommender role: it offers recommendations to users based on their preferences. We identify the main issues related to recommender systems. We introduce each briefly at this point [3].

Preferences

Recommendation is based on preferences. Thus, an automated recommender system must obtain preferences from people concerning the relevant domain. This raises a number of questions, including:

• Whose preferences are used? Those of the person seeking the recommendation, those of previous users of the system?

• How are preferences obtained? For example, do recommendation users have to express their own preferences as part of the process of seeking a recommendation?

• What incentives are there for people to offer preferences?

• What is the form of a preference? How are preferences represented?

Roles & Communication

• Is the recommender role filled by a computational system or a person?

• Do people play distinct roles, or do all users of a system play the same role? Are roles fixed, or do they evolve?

• How is the interaction between the recommendation user and the recommender initiated?

• What information about the people whose preferences are used in computing a recommendation is revealed to the recommendation user?

Algorithms for Computing Recommendations

• How does an automated recommender system determine whose preferences to use in computing a recommendation? If we think of all the people who have expressed their preferences for a given domain as being placed in a large, multi-dimensional space, this is the problem of finding neighbours in that space for the person seeking a recommendation.

• How are recommendations computed? For example, given that a set of neighbours for the recommendation seeker has been determined, how are the preferences of these neighbours weighted and combined?

2.3 RECOMMENDATION TECHNIQUES

To illustrate the prediction step of a Recommendation System, consider, for instance, a simple, non-personalized, recommendation algorithm that recommends just the most popular songs. The rationale for using this approach is that in absence of more precise information about the user’s preferences, a popular song, i.e., something that is liked (high utility) by many users, will also be probably liked by a generic user, at least more than another randomly selected song. Hence the utility of these popular songs is predicted to be reasonably high for this generic user.

It is also important to note that sometimes the user utility for an item is observed to depend on other variables. For instance, the utility of an item for a user can be influenced by the domain knowledge of the user (e.g., expert vs. beginning users of a digital camera), or can depend on the time when the recommendation is requested. Or the user may be more interested in items (e.g., a restaurant) closer to his current location. Consequently, the recommendations must be adapted to these specific additional details and as a result it becomes harder and harder task to correctly estimate what the right recommendations are.

The different types of recommender systems that are used in market vary in terms of the addressed domain, the knowledge used, but especially in regard to the recommendation algorithm, i.e., how the prediction of the utility of a recommendation. Other differences relate to how the recommendations are finally assembled and presented to the user in response to user requests. Recommendation Systems depending on the utility, algorithm aspects and context where they are used, these are divided into six different classes:

Content-based: The system learns to recommend items that are similar to the ones that the user liked in the past. The similarity of items is calculated based on the features associated with the compared items. For example, if a user has positively rated a movie that belongs to the comedy genre, then the system can learn to recommend other movies from this genre.

Collaborative filtering: The implementation of this approach recommends to the active user the items that other users with similar tastes liked in the past. The similarity in taste of two users is calculated based on the similarity in the rating history of the users. This is the reason why refers to collaborative filtering as "people-to-people correlation." Collaborative filtering is considered to be the most popular and widely implemented technique in Recommendation Systems.

Demographic: This type of system recommends items based on the demographic profile of the user. Many Web sites adopt simple and effective personalization solutions based on demographics. For example, users are dispatched to particular Web sites based on their language or country.

Knowledge-based: Knowledge-based systems recommend items based on specific domain knowledge about how certain item features meet user's needs and preferences and, ultimately, how the item is useful for the user. Knowledge-based systems tend to work better than others at the beginning of their deployment but if they are not equipped with learning components they may be surpassed by other shallow methods that can exploit the logs of the transactions.

***Community-based**:* This type of system recommends items based on the preferences of the users friends. This technique follows the epigram "Tell me who your friends are, and I will tell you who you are". People tend to rely more on recommendations from their friends rather than anonymous individuals. This observation, combined with the growing popularity of open social networks, is generating a rising interest in community-based systems.

Hybrid recommender systems: These systems are based on the combination of the above mentioned techniques. A hybrid system combining techniques A and B tries to use the advantages of A to fix the disadvantages of B. For instance, Collaborative Filtering methods suffer from new-item problems, i.e., they cannot recommend items that have no ratings. This does not limit content-based approaches since the prediction for new items is based on their description (features) that are typically easily available. Given two (or more) basic techniques, several ways have been proposed for combining them to create a new hybrid system.

Main Steps In Recommendation Systems

The entire structure of the recommendation system is summarized in Figure 2. The structure shows all the necessary phases in the implementation which are Data pre-processing, Analysis and Interpretation.

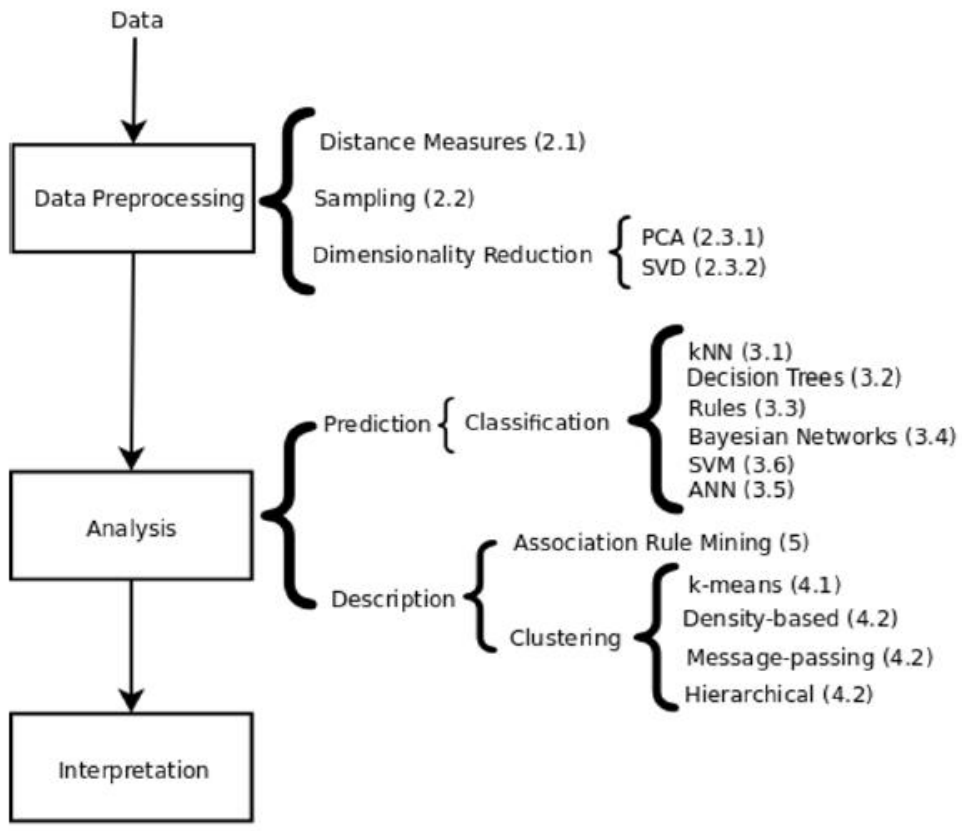


Figure 2 Process of Recommendation Systems [1]

Data Pre-Processing:

Data is collection of objects and their attributes, where an attribute is defined as a property or characteristic of an object. Real life data is too large and unclassified. Therefore it needs to be pre-processed (e.g. cleansed, filtered, transformed) in order to be used by the machine learning techniques in the analysis step. In this section, we focus on three issues that are of particular importance when designing a recommendation engine. First, we review different similarity or distance measures. Next, we discuss the issue of sampling as a way to reduce the number of items in very large collections while preserving its main characteristics. Finally, we describe the most common techniques to reduce dimensionality.

Similarity Measures: The simplest and most common example of a distance measure is the Euclidean distance: where n is the number of dimensions (attributes) and x_k and y_k are the kth attributes (components) of data objects x and y, respectively

$$d(x,y) = \sqrt{\sum_{k=1}^{n}(x_k - y_k)^2} \dots\dots (1)$$

The Mahalanobis distance is defined as:

$$d(x,y) = \sqrt{(x-y)\sigma^{-1}(x-y)^T} \dots\dots (2)$$

where σ is the covariance matrix of the data.

Another very common approach is to consider items as document vectors of an n-dimensional space and compute their similarity as the cosine of the angle that they form:

$$\cos(x,y) = \frac{(x.y)}{||x||\,||y||} \dots\dots (3)$$

where • indicates vector dot product and ||x|| is the norm of vector x. This similarity is known as the cosine similarity or the L2 Norm.

The similarity between items can also be given by their correlation which measures Linear relationship between objects. While there are several correlation coefficients that may be applied, the Pearson correlation is the most commonly used.

Given the covariance of data points x and y Σ, and their standard deviation σ, we compute the Pearson correlation using:

$$pearson(x,y) = \frac{\Sigma(x,y)}{\sigma_x * \sigma_y} \dots\dots (4)$$

Recommendation Systems have traditionally used either the cosine similarity or the Pearson correlation.

Sampling: It is used for selecting a subset of relevant data from a large data set. Processing the entire data set is computationally expensive therefore sampling is used. The dataset obtained is used for training and testing purpose. The training dataset is used to learn the parameters, while the testing dataset is used to evaluate the model, making sure that it performs well with previously unseen data.
Subset chosen should be such that it reflects the entire set's characteristics [1].

- ***Random Sampling:*** It is the easiest type of sampling and based on probability any set can be chosen.
- ***Stratified Sampling:*** Dataset is split into several portions with common characteristics. The final dataset is chosen based on probability in each portion.
- ***Sampling without Replacement:*** When an item is selected, it is removed from the population i.e. the item is not selected again if selected once.
- ***Sampling with Replacement:*** When an item is selected, it is not removed from the population, once they have been selected, allowing for the same sample to be selected more than once.

It is common practice to use standard random sampling without replacement with 20% of the instances for the testing set and leave the remaining 80% for training. The training process may be repeated several times to avoid over specialization of the dataset. The training and test sets are created from the original data set, the model is trained using the training data and tested with the examples in the test set. Finally, the average performance of the K learned models is reported, when the process is repeated K times. This process is known as cross-validation. Cross-validation may be unreliable if the data set is not sufficiently large.

A common approach in RS is to sample the available feedback from the users – e.g. in the form of ratings – to separate it into training and testing. We might, for instance, decide to sample only from most recent ratings – since those are the ones we would be predicting in a real-world situation.

Reducing Dimensionality*:* The density and distance between points, which are critical for clustering become less meaningful in highly dimensional spaces. Dimensionality reductions techniques help overcome this problem by transforming the original high-dimensional space into a lower-dimensionality. Applying dimensionality reduction makes such a difference and its results are so directly applicable to the computation of the predicted value, that this is now considered to be an approach to recommendation engine design, rather than a pre-processing technique. The common algorithms used in dimensionality reduction are: ***Principal Component Analysis (PCA)*** and ***Singular Value Decomposition (SVD).***

- **Principal Component Analysis [1]:** PCA obtains a sorted decreasing list of components that account for the largest amount of the variance from the data in terms of least square errors. We can reduce the dimensionality of the data by neglecting those components with a small contribution to the variance. PCA is a powerful technique, but it does have important limitations. PCA relies on the empirical data set to be a linear combination of a certain basis – although generalizations of PCA for non-linear data have been proposed. Matrix factorizations techniques such as SVD or Non-Negative Matrix Factorization are preferred for recent work but earlier works used PCA. E.g. A system starts from a standard matrix of user ratings to items. They then select a set by choosing the subset of items for which all users had a rating. This new matrix is then used to compute the global correlation matrix where a standard 2-dimensional PCA is applied.

- **Singular Value Decomposition [1]:** It finds a lower dimensional feature space where the new features represent "concepts" and the concepts are computable. SVD allows to automatically deriving semantic "concepts" in a low dimensional space. It is mostly used in collaborative filtering. First, SVD can be used to uncover latent relations between customers and products. In order to accomplish this goal, they first fill the zeros in the user-item matrix with the item average rating and then normalize by subtracting the user average. This matrix is then factored using SVD and the resulting decomposition can be used to compute the predictions. The other approach is to use the low-dimensional space resulting from the SVD to improve neighbourhood formation. The main advantage of SVD is that there are incremental algorithms to compute an approximated decomposition. This allows acceptance of new users or ratings without having to re-compute the model that had been built from previously existing data.

Finally, it should be noted that different variants of Matrix Factorization (MF) methods such as the Non-negative Matrix Factorization (NNMF) have also been used. These algorithms are, in essence, similar to SVD. The basic idea is to decompose the ratings matrix into two matrices, one of which contains features that describe the users and the other contains features describing the items. Matrix Factorization methods are better than SVD at handling the missing values by introducing a bias term to the model.

Classification

A classifier is a mapping between a feature space and a label space, where the features represent characteristics of the elements to classify and the labels represent the classes. In case of restaurants, for example, it can be implemented by a classifier that classifies restaurants into two categories (good, bad) based on a number of features that describe it. Mainly there are two types: ***Supervised Classification*** and ***Unsupervised Classification***.

In ***Supervised Classification***, a set of labels or categories is known in advance and we have a set of labelled examples which constitute a training set.

In ***Unsupervised Classification***, the labels or categories are unknown in advance and the task is to suitably (according to some criteria) organize the elements at hand. kNN and decision trees are type of supervised classification whereas clustering is type of unsupervised classification.

Nearest Neighbour: Classifiers type which memorizes the entire training set and classifies only if the attributes of the new record match one of the training examples exactly is an instance-based classifier. This is also known as rote learner. An instance-based classifier is the nearest neighbour classifier (kNN). The kNN classifier finds the k closest points (nearest neighbours) from the training records. It then assigns the class label according to the class labels of its nearest-neighbours. The idea is that if a record falls in a particular neighbourhood where a class label is predominant it is because the record is likely to belong to that very same class. They are the simplest of all machine learning algorithms and does not build model explicitly. Classifying unknown records is relatively expensive. Nearest Neighbour is highly used in Collaborative Filtering as both are based on the same idea i.e. Finding like-minded users (or similar items) .One other advantage is that it does not require to learn and maintain a given model. Therefore the system can adapt to rapid changes in the user ratings matrix.

The '?' in left Fig 3 is to be classified in cluster 1 or cluster 2. The right figure shows that if k=1(inner circle) '?' is classified as square else when k=7 the '?' is classified as 'circle' based on majority rule

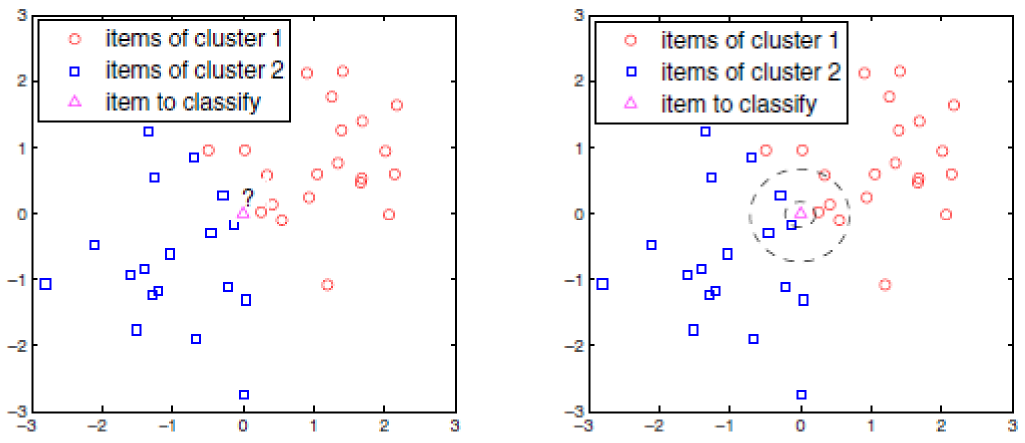


Figure 3 Classification and Clustering [1]

Decision Trees: Decision trees are classifiers on a target attribute in the form of a tree structure. The observations (or items) to classify are composed of attributes and their target value. The nodes of the tree can be:

a. Decision nodes, in these nodes a single attribute-value is tested to determine to which branch of the sub tree applies.
b. Leaf nodes which indicate the value of the target attribute.

It relies on the test condition applied to a given attribute that discriminates the observations by their target values. Once the partition induced by the test condition has been found, the algorithm is recursively repeated until a partition is empty or all the observations have the same target value.

Decision tree induction stops once all observations belong to the same class (or the same range in the case of continuous attributes). This implies that the impurity of the leaf nodes is zero. However, most decision trees implementations use pruning by which a node is not further split if its impurity measure or the number of observations in the node are below a certain threshold.

The main advantages of building a classifier using a decision tree is that it is inexpensive to construct and it is extremely fast at classifying unknown instances.

Another aspect of decision tree is that they can be used to produce a set of rules that are easy to interpret while maintaining accuracy. Decision trees may be used in a model-based approach for a RS. One possibility is to use content features to build a decision tree that models all the variables involved in the user preferences. As it could be expected, it is very difficult and unpractical to build a decision tree that tries to explain all the variables involved in the decision making process.

Decision trees, however, may also be used in order to model a particular part of the system. The Decision Tree is used as a filter to select which users should be targeted with recommendations. They are also used as item ranking in RS.

Cluster Analysis

Even if we reduce dimensionality of features, we might still have many objects to compute the distance so clustering algorithms are used. Clustering improves efficiency because the number of operations is reduced. Clustering is also referred to as unsupervised learning, consists of assigning items to groups so that the items in the same groups are more similar than items in different groups. Similarity is determined using similarity measures.

K-Means: K-Means clustering is a partitioning method i.e. data is divided into non overlapping clusters. The function partitions the data set of N items into k disjoint subsets Sj that contain Nj items so that they are as close to each other as possible according a given distance measure. Thus, we can define the k-means algorithm as an iterative process to minimize $= \sum_1^k \sum_{n \in S_j} d(x_n, \lambda_j)$, where x_n is a vector representing the n-th item, λ_j is the centroid of the item in S_j and d is the distance measure.

The algorithm works by randomly selecting k centroids. Then all items are assigned to the cluster whose centroid is the closest to them. The new cluster centroid needs to be updated to account for the items which have been added or removed from the cluster and the membership of the items to the cluster updated. This operation continues until there are no further items that change their cluster membership.

The basic k-means is an extremely simple and efficient algorithm. However, it does have several shortcomings:

- It assumes prior knowledge of the data in order to choose the appropriate k;
- The final clusters are very sensitive to the selection of the initial centroids;
- It can produce empty cluster.

K-means also has several limitations with regard to the data: it has problems when clusters are of differing sizes, densities, and non-globular shapes.

Association Rule Mining: Association Rule Mining focuses on finding rules that will predict the occurrence of an item based on the occurrences of other items in a transaction. We define an item set as a collection of one or more items (e.g. (Milk, Beer, and Diaper)). A k-item set is an item set that contains k items. The frequency of a given item set is known as support count (e.g. (Milk, Beer, Diaper) = 131). And the support of the item set is the fraction of transactions that contain it (e.g. (Milk, Beer, Diaper) = 0.12). A frequent item set is an item set with a support that is greater or equal to a threshold. An association rule is an expression of the form X =>Y, where X and Y are item sets. (E.g. Milk, Diaper => Beer). In this case the support of the association rule is the fraction of transactions that have both X and Y. On the other hand, the confidence of the rule is how often items in Y appear in transactions that contain X. Given a set of transactions T, the goal of association rule mining is to find all rules having support ≥ minimum support threshold and confidence ≥ minimum confidence threshold. Several techniques exist to optimize the generation of frequent item sets. The most common approach though, is to reduce the number of candidates using the Apriori principle. It states that if an item set is frequent, then all of its subsets must also be frequent. This is verified using the support measure because the support of an item set never exceeds that of its subsets.

Next we will discuss the two types of models which are popular in recommendation systems.

2.4 CONTENT BASED SYSTEMS

Content based recommendation systems try to recommend items similar to those a given user has liked in the past. Indeed, the basic process performed by a content-based recommender consists in matching up the attributes of a user profile in which preferences and interests are stored, with the attributes of a content object (item), in order to recommend to the user new interesting items.

Users need a personalized support in going through large amounts of available information according to their tastes and interest, content based systems play a main role in achieving this in a web system.

Ex: At Amazon (Figure 4) recommendation algorithms are used to personalize the online store for each customer. Example as shown in YouTube Figure 5.

- Baby toys to a new mother
- Programing titled books to a software engineer

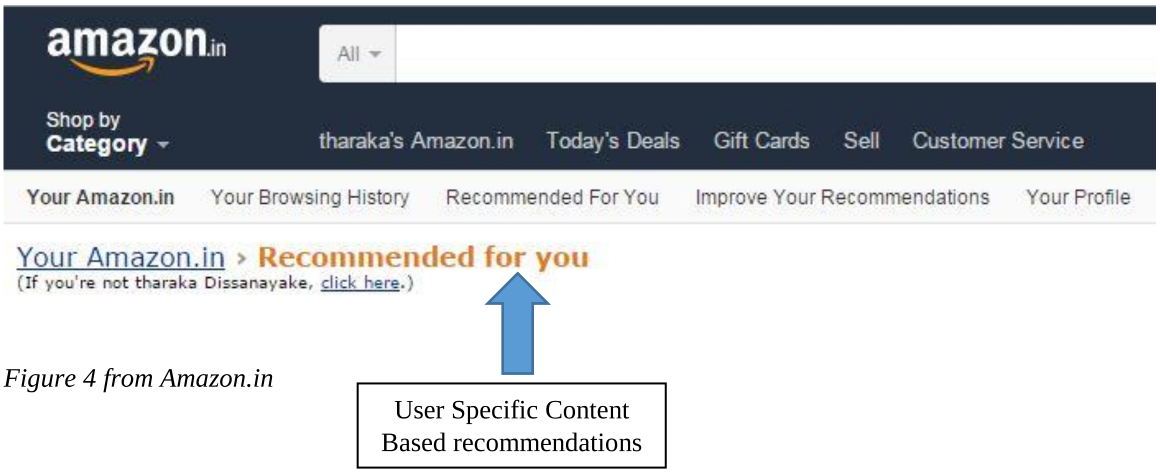


Figure 4 from Amazon.in

Basic steps in building a content based system.

- Analyse a set of documents or descriptions of items previously rated by a user.
- Build a model or profile of user interests based on the features of the objects rated by that user. Ex: Personalized Search

Filter results by deciding whether a user is interested in a specific webpage or not and if not prevent it from displaying as shown in Figure 5.

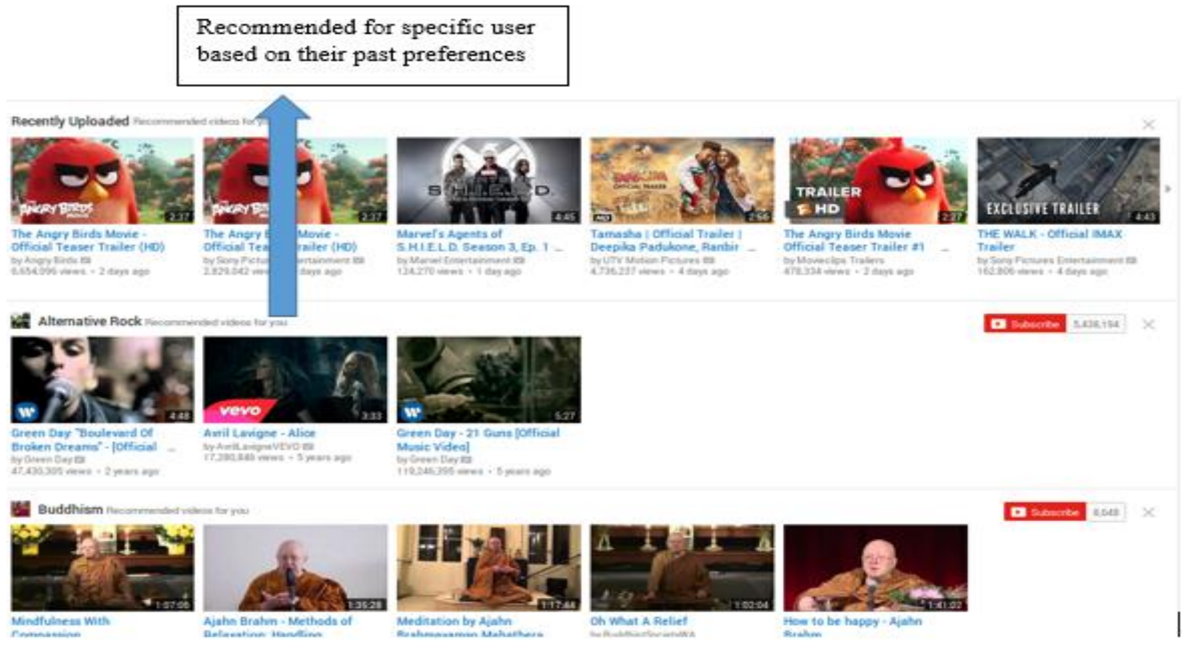
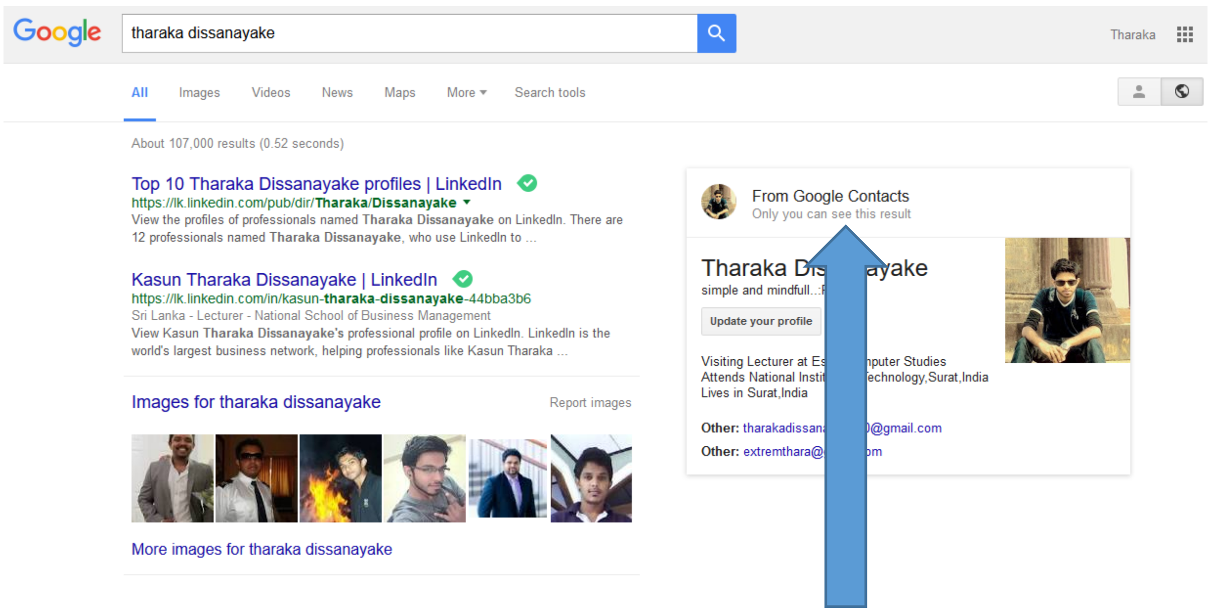

Recommended for specific user based on their past preferences
Recently Uploaded
The Angry Birds Movie - Official Teaser Trailer (HD)
Marvel's Agents of S.H.I.E.L.D. Season 3, Ep. 1
Tamasha | Official Trailer | Deepika Padukone, Ranbir
The Angry Birds Movie Official Teaser Trailer #1
THE WALK - Official IMAX Trailer
Alternative Rock
Green Day "Boulevard Of Broken Dreams" - [Official
Avril Lavigne - Alice
Green Day - 21 Guns [Official Music Video]
Buddhism
Mindfulness With Compassion
Ajahn Brahm - Methods of Relaxation
Meditation by Ajahn Brahmavamso Mahathera
Oh What A Relief
How to be happy - Ajahn Brahm

Figure 5 from youtube.com


tharaka dissanayake

Tharaka

All Images Videos News Maps More Search tools

About 107,000 results (0.52 seconds)

Top 10 Tharaka Dissanayake profiles | LinkedIn
https://lk.linkedin.com/pub/dir/**Tharaka/Dissanayake**
View the profiles of professionals named **Tharaka Dissanayake** on LinkedIn. There are 12 professionals named **Tharaka Dissanayake**, who use LinkedIn to ...

Kasun Tharaka Dissanayake | LinkedIn
https://lk.linkedin.com/in/kasun-**tharaka-dissanayake**-44bba3b6
Sri Lanka - Lecturer - National School of Business Management
View Kasun **Tharaka Dissanayake's** professional profile on LinkedIn. LinkedIn is the world's largest business network, helping professionals like Kasun Tharaka ...

Images for tharaka dissanayake — Report images

More images for tharaka dissanayake

From Google Contacts
Only you can see this result

Tharaka Dis ayake
simple and mindfull..:
Update your profile

Visiting Lecturer at Es puter Studies
Attends National Insti echnology,Surat,India
Lives in Surat,India

Other: tharakadissan)@gmail.com
Other: extremthara@ om

Figure 6 from Google.in

Personalized according to user contacts

High Level Architecture

There are 3 main components in any Content Based System,

They are,

- ***Content Analyser***

When Information has no structure (ex: text) some kind of pre-processing step is needed to extract structured relevant information, this process is done by the content analyser, it basically categorize each item in the database (keywords, n-grams).

Ex: web pages by keywords

Output of the content analyser will be the input to the profile learner and filtering component.

- ***Profile Learner***

This module collects data representative of the user preferences and tries to generalize this data, in order to construct the user profile. Usually, the generalization strategy is realized through machine learning techniques, which are able to infer a model of user interests starting from items liked or disliked in the past.

For instance, the profile learner of a Web page recommender can implement a relevance feedback method in which the learning technique combines vectors of positive and negative examples into a prototype vector representing the user profile. Training examples are Web pages on which a positive or negative feedback has been provided by the user;

- ***Filtering component***

This module exploits the user profile to suggest relevant items by matching the profile representation against that of items to be recommended. The result is a binary or continuous relevance judgment (computed using some similarity metrics), the latter case resulting in a ranked list of potentially interesting items.

U_a – active user

I_k - k^{th} item

R_k - rating of k^{th} item by U_a

TR_a – Training Set, is a set of pairs i.e $<I_k, R_k>$

L_a – Top ranked items list

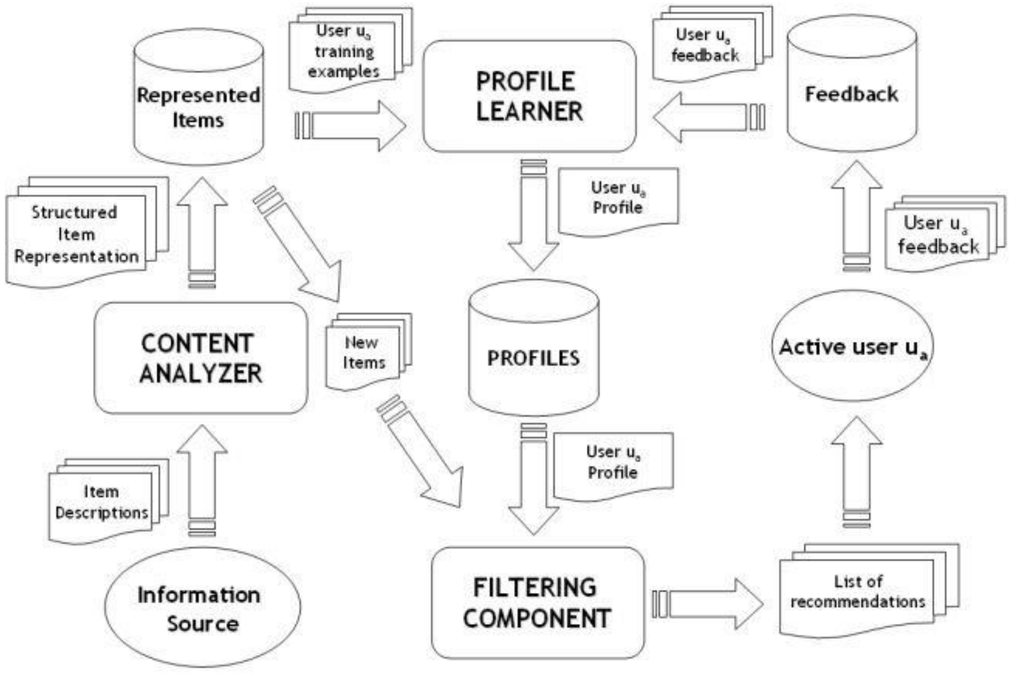


Figure 7 High level architecture of a Content-based Recommender [1]

The user profile or the predictive model will be created by applying supervised learning algorithms and will be stored in the profile repository for the later use of filtering component.

In the filtering component, user profile will be compared with items to be recommended and that will create L_a – the top ranked items list according to the active user, U_a at the time.

Then the L_a will be presented to the user and depending on user feedback, we can calculate the accuracy of the predictions and can update the user profile in an iterative loop. So it will increase the accuracy and it will keep the system up to date according to the users taste. This is important because user's taste may change over time.

After gathering the feedback, the learning process is performed again on the new training set and the resulting profile is adapted to the updated to the updated user interests.

User Feedback plays a main role in this,

- ***Feedback***

  a. Explicit feedback – direct user involvement
     I. Like/Dislike
     II. Rating
     III. Comments
  b. Implicit feedback – no direct user involvement
     I. Deriving from user activities
     II. Saving
     III. Discarding
     IV. Bookmarking

Content Based Models

Keyword-based Vector Space Model [1]: Most content-based recommender systems use relatively simple retrieval models, such as keyword matching or the Vector Space Model (VSM). VSM is a spatial representation of text documents. In that model, each document is represented by a vector in an *n*-dimensional space, where each dimension corresponds to a term from the overall vocabulary of a given document collection.

E.g.: http://www.pandora.com, http://www.imdb.com

Semantic Analysis by using Ontologies [1]: Semantic analysis allows learning more accurate profiles that contain references to concepts defined in external knowledge bases. The main motivation for this approach is the challenge of providing a recommender system with the cultural and linguistic background knowledge which characterizes the ability of interpreting natural language documents and reasoning on their content.

Ex: *SiteIF* is a personal agent for a multilingual news Web site. The external knowledge source involved in the representation process is MultiWordNet, a multilingual lexical database where English and Italian senses are aligned. Each news is automatically associated with a list of MultiWordNet synsets by using Word Domain Disambiguation. The user profile is built as a semantic network whose nodes represent synsets found in the documents read by the user. During the matching phase, the system receives as input the synset representation of a document and the current user model, and it produces as output an estimation of the document relevance by using the Semantic Network Value Technique.

Advantages of Content-based Filtering

User Independence: Content-based recommenders exploit ratings provided by the active user to build her own profile. Instead, collaborative filtering methods need ratings from other users in order to find the "nearest neighbours" of the active user, i.e., users that have similar tastes since they rated the same items similarly. Then, only the items that are most liked by the neighbours of the active user will be recommended.

Transparency: Explanations on how the recommender system works can be provided by explicitly listing content features or descriptions that caused an item to occur in the list of recommendations. Those features are indicators to consult in order to decide whether to trust a recommendation. Conversely, collaborative systems are black boxes since the only explanation for an item recommendation is that unknown users with similar tastes liked that item.

New Item: Content-based recommenders are capable of recommending items not yet rated by any user. As a consequence, they do not suffer from the first-rater problem, which affects collaborative recommenders which rely solely on users' preferences to make recommendations. Therefore, until the new item is rated by a substantial number of users, the system would not be able to recommend it.

Drawbacks of Content-based Filtering

- ***Limited Content Analysis:***

Content-based techniques have a natural limit in the number and type of features that are associated, whether automatically or manually, with the objects they recommend. Domain knowledge is often needed. For instance, for movie recommendations the system needs to know the actors and directors, and sometimes, domain ontologies are also needed. No content-based recommendation system can provide suitable suggestions if the analysed content does not contain enough information to discriminate items the user likes from items the user does not like.

- ***Over-Specialisation:***

Content-based recommenders have no inherent method for finding something unexpected. The system suggests items whose scores are high when matched against the user profile, hence the user is going to be recommended items similar to those already rated. This drawback is also called *serendipity* problem to highlight the tendency of the content-based systems to produce recommendations with a limited degree of novelty. To give an example, when a user has only rated movies directed by Stanley Kubrick, she will be recommended just that kind of movies. A “perfect” content-based technique would rarely find anything *novel*, limiting the range of applications for which it would be useful

- ***New User:***

Enough ratings have to be collected before a content-based recommender system can really understand user preferences and provide accurate recommendations. Therefore, when few ratings are available, as for a new user, the system will not be able to provide reliable recommendations.

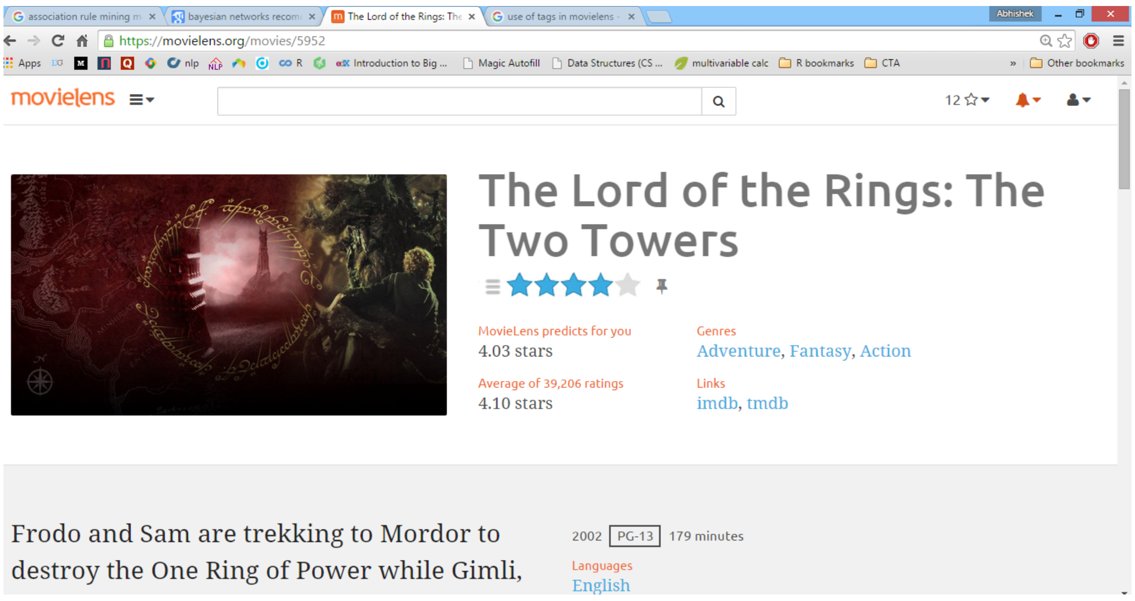
2.5 COLLABORATIVE FILTERING SYSTEMS

Collaborative Filtering (CF) is a popular recommendation algorithm that predicts or recommends on the basis of opinions or ratings provided by other users in the system. While the term collaborative filtering has only been known for a decade or two, the idea has its root far in the centuries- humans sharing its opinions with others. For years, people in the society have discussed on the books they have read, the brand they prefer, movies they like, restaurants with best foods and share their experience with their friends.

But now with the advancement in computer and web allow us to move beyond simple word-of-mouth. Instead of limiting ourselves on only tens of people the Internet allow us to consider the opinions of thousands and millions. The speed of computer allow us to process data in real time and determine the what people thinks about an item but also help to personalize the items for a given user.

Core Concept

The key idea is that the user U will rate item I same as user V if U and v has rated similarly to various items in past i.e. users are similar to each other. For example Movie Lens is a collaborative filtering system for movie recommendation. User of Movie Lens rate a movie on the scale of 1 to 5 star where 1 is “Awful” and 5 is “Must See”. Movie Lens then uses the ratings of the community to recommend other movies that user might be interested in (figure 8), predict what that user might rate a movie and perform other task.



The Lord of the Rings: The Two Towers

MovieLens predicts for you
4.03 stars

Genres
Adventure, Fantasy, Action

Average of 39,206 ratings
4.10 stars

Links
imdb, tmdb

Frodo and Sam are trekking to Mordor to destroy the One Ring of Power while Gimli,

2002 PG-13 179 minutes

Languages
English

Figure 8 from MovieLens.com

To be more precise and formal, rating is an association of both user and item often by means of some value. One way to visualize the rating is as a matrix. Without loss of generality, a ratings matrix consists of a table where each row represents a user, each column represents a specific movie, and the number at the intersection of a row and a column represents the user’s rating value. The absence of a rating score at this intersection indicates that user has not yet rated the item.

The term *User* refers to any individual who provides ratings to a system. And *item can* be anything which we review and give opinion.

Table 1 Movie Lens Rating Matrix

	The Matrix	Speed	Avengers	Avatar
User A	1	2	5	
User B		3	5	4
User C	5	5	4	1
User D	5	5	5	5

Task of Collaborative filtering

- Help to find new items user might like. In a world of overloaded information, User cannot evaluate all things. There should be few choices before user. This has been applied most commonly to consumer items (music, books, movies), but may also be applied to research papers, web pages, or other ratable items.
- Advise User on a particular item. User have a particular item in mind; does the community know whether it is good or bad?
- Help to find a user (or some users) he/she might like. Sometimes, knowing who to focus on is as important as knowing what to focus on. This might help with forming discussion groups, matchmaking, or connecting users so that they can exchange recommendations socially.
- Help user to find something new that he/she might like. CF can help groups of people find items that maximize value to group as a whole. For example, a couple that wishes to see a movie together or a research group that wishes to read an appropriate paper.

Collaborative Filtering and the Adaptive Web

Earlier collaborative filtering systems were designed to explicitly provide users with information about items. That is, users visited a website for the purpose of receiving recommendations from the CF system. Later, websites began to use CF systems behind the scenes to adapt their content to users, such as choosing which news articles a website should be presenting prominently to a user. Collaborative filtering can predict what information users are likely to want to see, enabling providers to select subsets of information to display in the limited screen space. It enables the user to maximize their limited attention. In this way, collaborative filtering enables the web to adapt to each individual user's needs.

Collaborative Filtering Algorithms

Memory-Based Algorithms [1]: Memory-based algorithms requires all ratings, items, and users be stored in memory. Pure memory based algorithms do not scale well for real world application. These are Non-Probabilistic algorithms. Algorithms that are used in this class are Graph-based algorithms, K-Nearest Neighbor and Rule Mining. The most commonly used are User-Based Nearest Neighbor and Item-Based Nearest neighbor.

- ***User-Based Nearest Neighbor Algorithm:*** The core concept of User-Based Nearest Neighbor is to find other users whose past rating behavior is similar to that of the current user and use their ratings on other items to predict what the current user will like. If a User N is similar to a user U, we say that user U is similar to user N. User-based algorithms generate a prediction for an item I by analyzing ratings for I from users in U's neighborhood. Naively, we could average all neighbors' ratings for item I. The rating of user u for item i, pred (u, i), can be calculated as:

$$pred(u,i) = \bar{r_u} + \frac{\sum_{n\subset neighbours(u)} userSim(u,n)\cdot(r_{ni}-\bar{r_n})}{\sum_{n\subset neighbours(u)} userSim(u,n)} \ldots\ldots(5)$$

Where, userSim (u, n) is a measure of similarity between a target user u and a neighbor n.

- ***Pearson Correlation:***

$$userSim(u,n) = \frac{\sum_{i\subset CR_{u,n}}(r_{ui}-\overline{r_u})(r_{ni}-\overline{r_n})}{\sqrt{\sum_{i\subset CR_{u,n}}(r_{ui}-\overline{r_u})^2}\sqrt{\sum_{i\subset CR_{u,n}}(r_{ni}-\overline{r_n})^2}} \ldots\ldots(6)$$

- ***Spearman Rank Correlation:***

In Spearman Correlation, the items a user has rated are ranked such that their highest-rated item is at rank 1 and lower rated items have higher ranks. Items with the same rating are assigned the average rank for their position. The computation is then the same as that of the Pearson correlation, except that ranks are used in place of ratings.

- ***Cosine similarity:***

It is a vector-space approach based on linear algebra. Similarity is measured by the cosine distance between two rating vectors.

Item-Based Nearest Neighbor Algorithm [1]: User-Based Nearest Neighbor suffers scalability problem as user base grows. To extend collaborative filtering to larger user bases and facilitate deployment on e-commerce sites, it was necessary to develop more scalable algorithms. Item-Based collaborative filtering came into existence to solve the problem. Rather than compute similarity between users, similarity between items is calculated. Item-based Nearest Neighbor algorithms are the transpose of User-Based Algorithm. . If two items tend to have the same users like and dislike them, then they are similar and users are expected to have similar preferences for similar items.

After collecting set, S, of item similar to i, prediction can be calculated as-

$$pred(u,i) = \frac{\sum_{j \in ratedItems(u)} itemSim(i,j) \cdot r_{ui}}{\sum_{j \in ratedItems(u)} itemSim(u,n)} \dots \dots (7)$$

Similarity, itemSim (i, j) can be calculated by cosine similarity-

$$itemSim(i,j) = \frac{\sum_{u \subset RB_{i,j}} (r_{ui} - \overline{r_u})(r_{uj} - \overline{r_u})}{\sqrt{\sum_{u \subset RB_{i,j}} (r_{ui} - \overline{r_u})^2} \sqrt{\sum_{u \subset RB_{i,j}} (r_{uj} - \overline{r_u})^2}} \dots \dots (8)$$

Association Rule Mining: Association mining techniques build models based on commonly occurring patterns in the ratings matrix. For example, we may observe that users who rated item 1 highly often rate item 2 highly. A particular rule is represented by an input condition (e.g. item 1 rated highly) and a result condition (e.g. item 2 rated highly). The support of a rule represents the fraction of users who have rated both the input and result conditions, and the confidence of a rule is the fraction of users with the input condition that exhibit the result condition. In order to generate a predicted rating for a user u and item i, we first select the rules with a result condition of item i that only include items rated by user u. We then use a heuristic to translate the support, accuracy, and ratings for input conditions into a predicted rating.

Graph-Based Methods: In graph-based approaches, the data is represented in the form of a graph where nodes are users, items or both, and edges encode the interactions or similarities between the users and items. For example (figure 9), the data is modeled as a bipartite graph where the two sets of nodes represent users and items, and an edge connects user u to item i if there is a rating given to i by u in the system. A weight can also be given to this edge, such as the value of its corresponding rating**.**

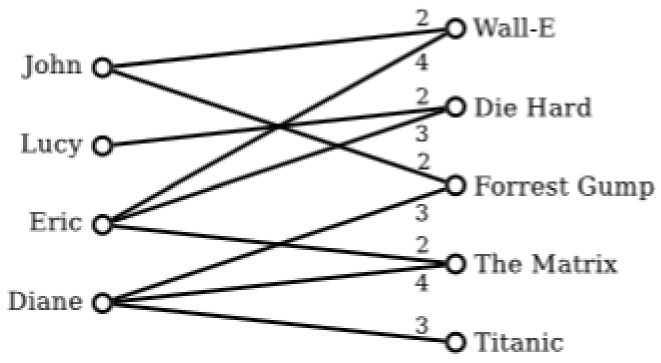


Figure 9 Collaborative Filtering Examples [1]

Model-Based Algorithms: These are the probabilistic models developed using Machine Learning and Data Mining Algorithms Like Bayesian Networks, Clustering Models, Latent Semantic Models such as Singular Value Decomposition, Probabilistic Latent Semantic Analysis, Multiple Multiplicative Factor, Latent Dirichlet Allocation and Markov Decision Process based models.

Most of the models are based on creating a Classification or Clustering techniques to identify the users on the basis of training set.

Advantages of Collaborative Filtering

- Diversity in the recommendation increases because of some different interest of similar user. This allows current user to explore new things that he may like.
- Collaborative filtering does not require content of the item for evaluating the item as user are evaluating the items. Content extraction may be very tedious task.

Shortcomings

- Since it is based on user reviews and ratings of an item so whenever a new item is in the system it do not get recommended even if it is of user interest. This is the condition of "rich becoming richer and poor becoming poorer**".** This can be overcome by Content Based Recommendation system which analyses the content of item rather than relying on user ratings.

CHAPTER III IMPLEMENTATION OF CONTENT BASED SYSTEM

Content-based Recommendation Systems rely on the past user experience in generating items which are relevant to user. In a content-based method each user is uniquely characterized and the user's interest is taken into account in building the user profile. User past history or his past experience with the system is logged into a file. This log file is used to understand the user behaviour based on the features it has and the work he has done with the past items.

3.1 SYSTEM ARCHITECTURE

Figure 10 describes the system architecture developed for our content based system. This recommendation system uses different feature sets [5] and makes computations on them for analysing the user's behaviour.

The implemented recommendation system uses genre feature set in predicting the recommendations which the user can watch/interested. Each user gives his rating to the movies which he has watched and these ratings are used to build the user behaviour. We produce ratings based on these ratings which user has given. In order to predict a rating for user, using feature set, we merge the user specific weights of movie's features. We also produce ratings using all features of the genre sets. We evaluate our recommendation method based on precision and the root-mean square error.

The dataset initially loaded for data pre-processing where we structure data for efficient computation. During pre-processing all the anomalies, missing values, outliners, etc. are removed for structured data. Then according to user interest we recommend him the new movies accordingly.

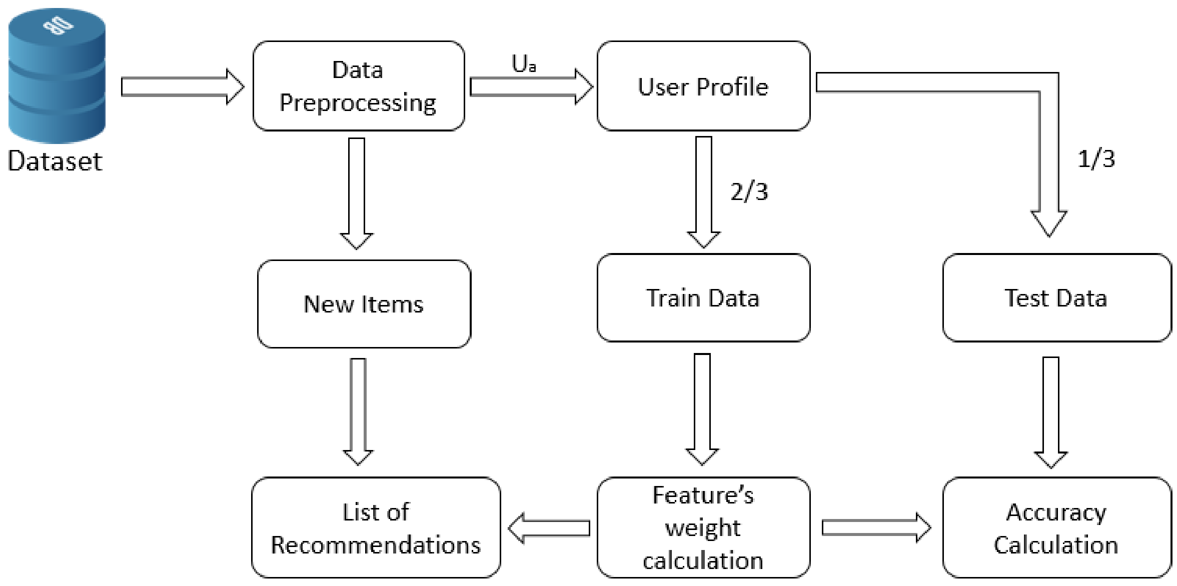


Figure 10 Proposed Content Based Recommendation System Architecture

Pre-processing

In this pre-processing stage the raw data is given as input and we transform it into structured data as shown in Figure 11.

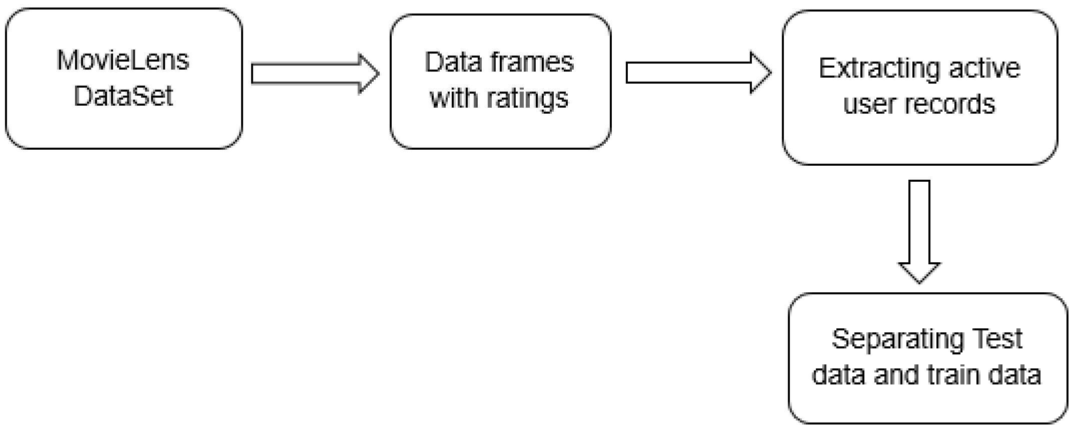

MovieLens DataSet
Data frames with ratings
Extracting active user records
Separating Test data and train data

Figure 11 Data Pre-processing For Content Based Recommendation System

Now, from the movies data frame extracted before, we get the currently active user's records from the rating data frame. This data is then divided into test and training dataset in the ratio of 2:3.

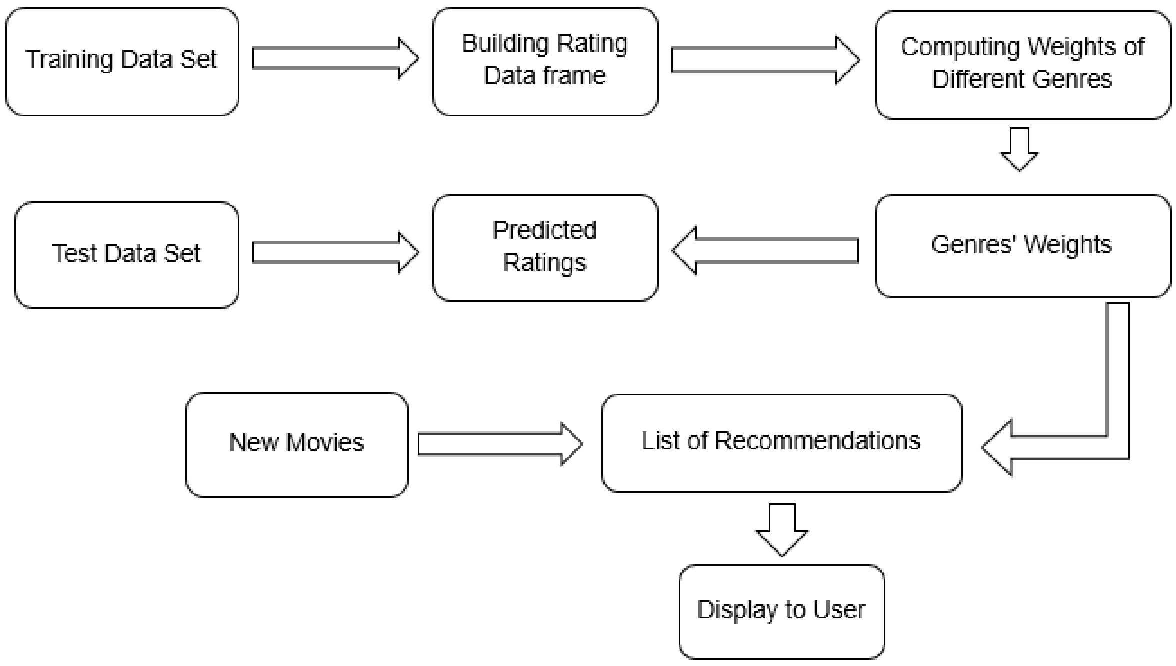


Figure 12 Recommendation Process for Content Based Recommendation System

Recommendation

In this part we explain the entire recommendation procedure done on the training and test data. We take the training dataset frame which we extracted from previous pre-processing part. Now we build the rating matrix by multiplying the ratings given by user with the weights of the genres of corresponding movie id. After this the entire weights of the genre is calculated by summing up the entire column. This gives us the weights of the genres for the corresponding user. Now we have built an entire Item-Feature matrix for the active user.

Let user *u* watch items $i_0,\ldots,i_8$, which have features $j_0,\ldots,j_3\ldots$.. In Table 2, we show the features for the feature set *k* for user *u*. Note that $j_0,\ldots,j_3\ldots$ contain the features that appear on items all users watched in the training set. Rating column shows the rating of user *u* for movies $i_0,\ldots, i_8$.

The weight of feature j in feature set *k* for user *u* is calculated as [5]:

$$w_k(u,j) = \frac{1}{|I_u^{train}|}\sum_{i\in I_u^{train}} x_{k,u}(i,j)r(u,i) \ldots\ldots (9)$$

In equation 9, *k* represents the type of the feature set, *k* {Sci-Fi, Comedy, Action …}. $r(u,i) \in R$ is the implicit rating of user *u* for item *i* and $x_{k,u}(i,j) \in \{0,1\}$ j^{th} feature of item *i*. I_u^{train} is the set of movies watched by user *u* in the training period. If movie *i* has feature j then $x_{k,u}(i,j)$ will be 1 and user rating for movie *i* will contribute to the sum.

Table 2 Item Feature Matrix

User U_1	Comedy (j_1)	Action (j_2)	Drama (j_3)	Musical (j_4)	Etc... (j_5)... (j_n)	Rating
$Movie_1$ (i_1)	1	0	1	0	1	4
$Movie_2$ (i_2)	0	1	1	0	0	5
$Movie_3$ (i_3)	1	1	0	0	0	3
$Movie_4$ (i_4)	1	0	0	1	0	5
$Movie_5$ (i_5)	0	1	1	0	0	2
..... (i_6)..... (i_k)	1	0	0	1	1	4
Weight	2.66	1.66	1.83	1.50	1.33	

After calculating the weight of each feature for each user, we use it to predict the recommendation ratings of contents. We calculate the rating for each feature set separately using the weights $w_k(u, j)$ obtained by using the training data.

We use two steps to generate ratings. As it is seen in the first equation shown above is to sum up the feature weights of the contents. Later this one is normalized this sum, by dividing it to the number of its features.

$$r_k(u,i) = \sum_{j \in D_{k,i}} w_k(u,j) \; \dots \dots (10)$$

$$r'_k(u,i) = \frac{1}{|D_{k,i}|} \sum_{j \in D_{k,i}} w_k(u,j) \; \dots \dots (11)$$

$r_k(u,i)$, represents the rating that user u gives the movie i according to the k feature set. $r'_k(u,i)$, is the normalized rating of the feature weights summation with $D_{k,i}$. $D_{k,i}$, is the features that appear in movie *i* from feature set *k*.

Calculating Accuracy

The $r'_k(u,i)$, $r_k(u,i)$ are the predicted ratings from the user profile by the proposed technique. Now to calculate the efficiency of the prediction we use root mean square method which is described as follows:

$$RMSE = \sqrt{\frac{1}{I_{testdata}} \sum_{i \, \epsilon \, testdata} (r_k(u,i) - a\_r(u,i))^2} \dots \dots (12)$$

In equation 12 where $r_k(u,i)$ is the predicted rating by the recommendation engine and $a\_r(u,i)$ is the actual rating given by the User.

3.2 EXPERIMENTAL SETUP

Operating System: Windows (7/8/10)/Linux (LinuxMint)

System Specifications: RAM of 4GB running Intel Core i-5 processor

Programming Language: Python

Libraries Used: Pandas for handling data frames, SciPy NumPy for scientific equations, matplotlib for plotting the graphs

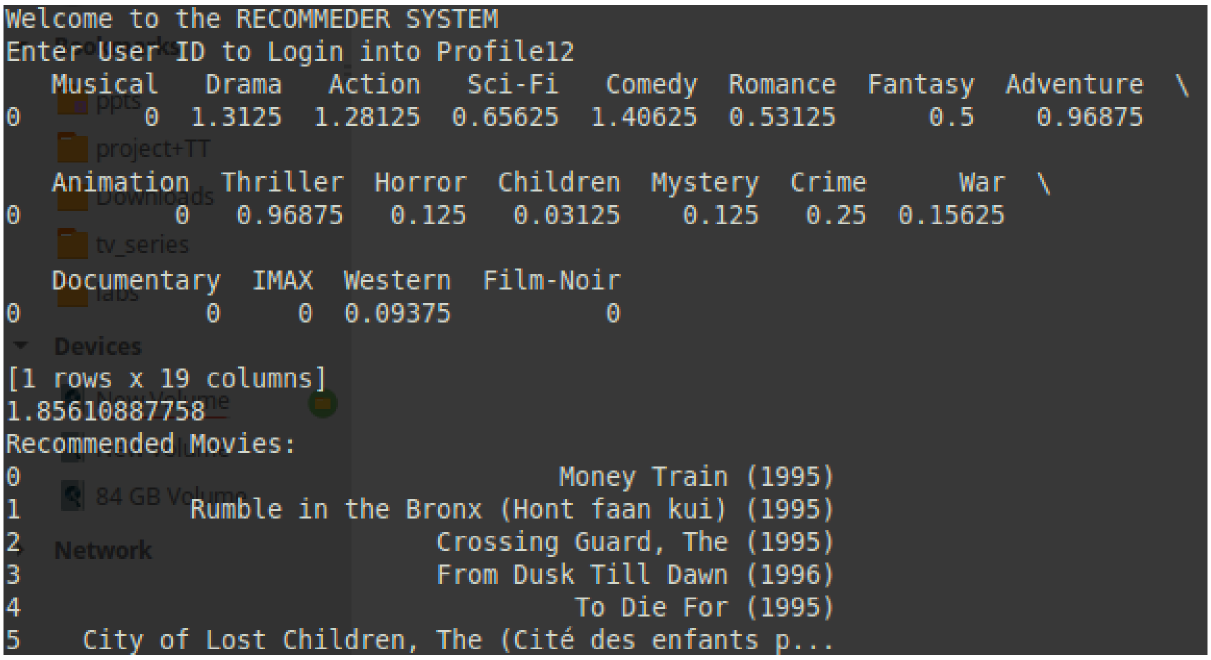
3.3 DATASET

The dataset for movie recommendations is chosen from the GroupLens Research Project at University of Minnesota. This dataset describes 5-star rating from MovieLens [6], a movie recommendation service from GroupLens. It contains 100234 ratings across 8927 movies. These data were created by users between March 26, 1996 and August 06, 2015 Users were selected at random for inclusion. All selected users had rated at least 1 movies. Each user is represented by an id, and no other information is provided.

The data are contained in four files, links.csv, movies.csv, ratings.csv and tags.csv. We use movies.csv and ratings.csv for our predictions.

3.4 RESULTS AND ANALYSIS

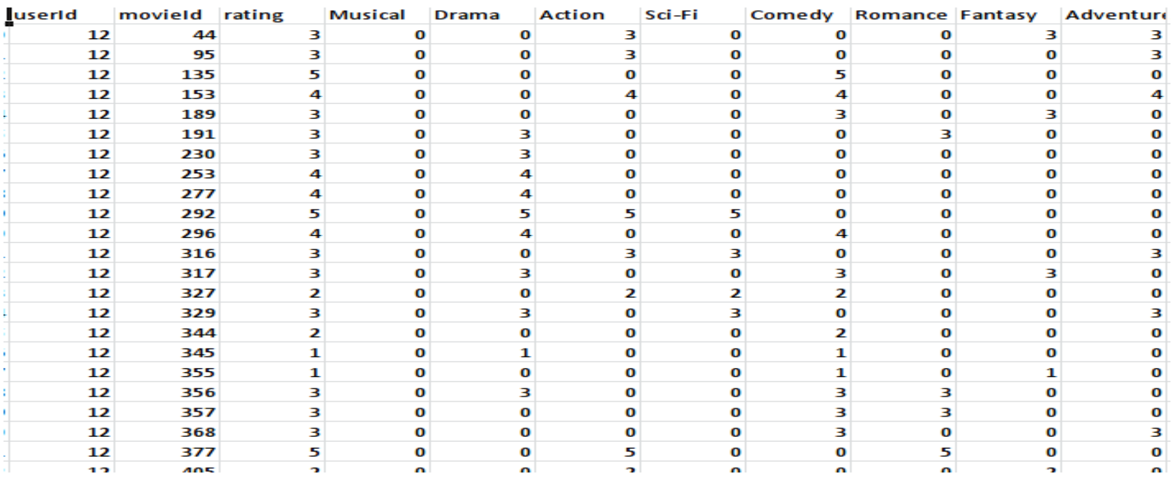
We have implemented the Content based Recommendation System as discussed in the above. The technique was implemented on the Movie Lens dataset [6] taken from the Group Lens Project. The data set has different movies classified according to their genres and it also has user profiles who have rated different movies according to their interests. The results of the recommendation system are summarized below for the implementation.



```
Welcome to the RECOMMEDER SYSTEM
Enter User ID to Login into Profile12
   Musical    Drama    Action    Sci-Fi    Comedy  Romance  Fantasy  Adventure  \
0        0  1.3125  1.28125  0.65625  1.40625  0.53125      0.5    0.96875

   Animation  Thriller  Horror  Children  Mystery  Crime      War  \
0          0   0.96875   0.125   0.03125    0.125   0.25  0.15625

   Documentary  IMAX  Western  Film-Noir
0            0     0  0.09375          0

[1 rows x 19 columns]
1.85610887758
Recommended Movies:
0                                    Money Train (1995)
1            Rumble in the Bronx (Hont faan kui) (1995)
2                            Crossing Guard, The (1995)
3                            From Dusk Till Dawn (1996)
4                                    To Die For (1995)
5    City of Lost Children, The (Cité des enfants p...
```

Figure 13 Recommendations Provided by the Content Based System

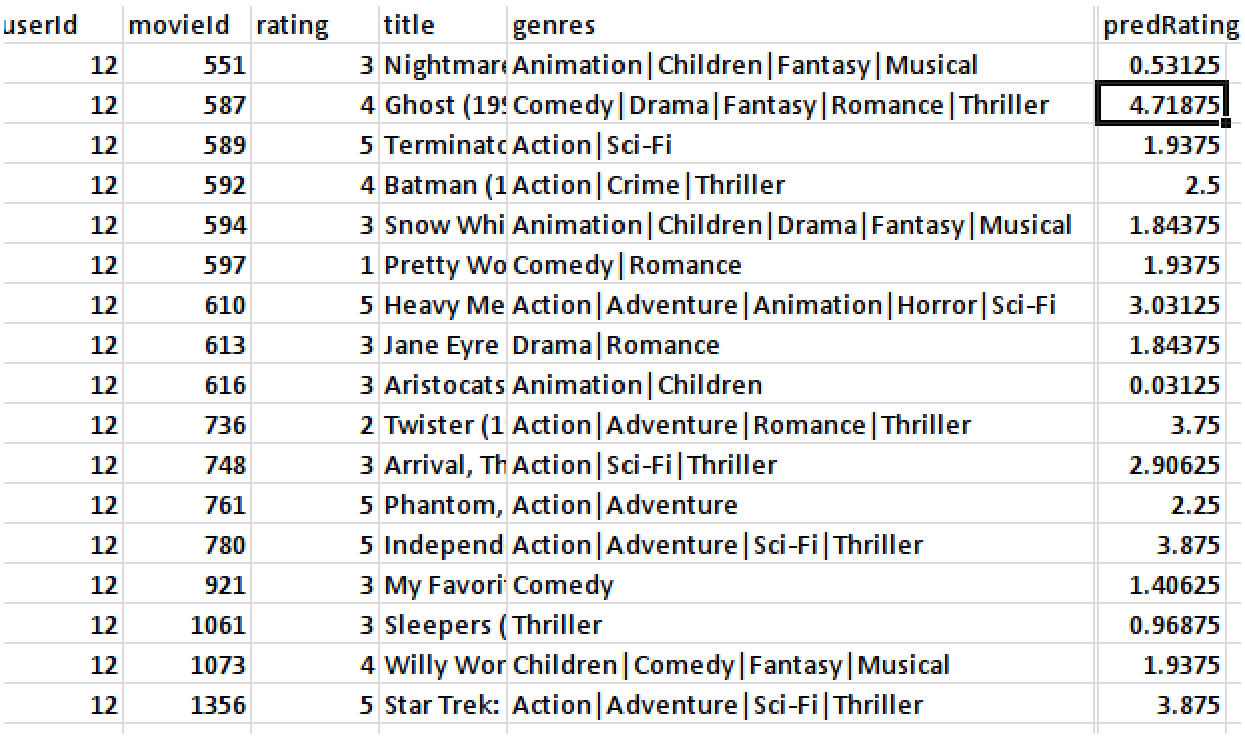
The Figure 13 shows the implementation of the discussed recommender system. The output shows for the user profile 12 as entered by the user. The output shows the weights of different genres calculated for the user profile 12.



userId	movieId	rating	Musical	Drama	Action	Sci-Fi	Comedy	Romance	Fantasy	Adventure
12	44	3	0	0	3	0	0	0	3	3
12	95	3	0	0	3	0	0	0	0	3
12	135	5	0	0	0	0	5	0	0	0
12	153	4	0	0	4	0	4	0	0	4
12	189	3	0	0	0	0	3	0	3	0
12	191	3	0	3	0	0	0	3	0	0
12	230	3	0	3	0	0	0	0	0	0
12	253	4	0	4	0	0	0	0	0	0
12	277	4	0	4	0	0	0	0	0	0
12	292	5	0	5	5	5	0	0	0	0
12	296	4	0	4	0	0	4	0	0	0
12	316	3	0	0	3	3	0	0	0	3
12	317	3	0	3	0	0	3	0	3	0
12	327	2	0	0	2	2	2	0	0	0
12	329	3	0	3	0	3	0	0	0	3
12	344	2	0	0	0	0	2	0	0	0
12	345	1	0	1	0	0	1	0	0	0
12	355	1	0	0	0	0	1	0	1	0
12	356	3	0	3	0	0	3	3	0	0
12	357	3	0	0	0	0	3	3	0	0
12	368	3	0	0	0	0	3	0	0	3
12	377	5	0	0	5	0	0	5	0	0
12	405	2	0	0	2	0	0	0	2	0

Figure 14 Item Feature Matrix for Content Based System

The Figure 14 shows the Item Feature matrix that is generated by multiplying the ratings with the genres of the corresponding movie id in the user profile 12. This matrix is used to build the user profile weights of different genres.



userId	movieId	rating	title	genres	predRating
12	551	3	Nightmar	Animation|Children|Fantasy|Musical	0.53125
12	587	4	Ghost (19	Comedy|Drama|Fantasy|Romance|Thriller	4.71875
12	589	5	Terminat	Action|Sci-Fi	1.9375
12	592	4	Batman (1	Action|Crime|Thriller	2.5
12	594	3	Snow Whi	Animation|Children|Drama|Fantasy|Musical	1.84375
12	597	1	Pretty Wo	Comedy|Romance	1.9375
12	610	5	Heavy Me	Action|Adventure|Animation|Horror|Sci-Fi	3.03125
12	613	3	Jane Eyre	Drama|Romance	1.84375
12	616	3	Aristocats	Animation|Children	0.03125
12	736	2	Twister (1	Action|Adventure|Romance|Thriller	3.75
12	748	3	Arrival, Th	Action|Sci-Fi|Thriller	2.90625
12	761	5	Phantom,	Action|Adventure	2.25
12	780	5	Independ	Action|Adventure|Sci-Fi|Thriller	3.875
12	921	3	My Favori	Comedy	1.40625
12	1061	3	Sleepers (	Thriller	0.96875
12	1073	4	Willy Wor	Children|Comedy|Fantasy|Musical	1.9375
12	1356	5	Star Trek:	Action|Adventure|Sci-Fi|Thriller	3.875

Figure 15 Predicted Ratings by the Content Based System

The Figure 15 shows the predicated ratings based on the weights of the genres calculated from the item feature matrix as shown in Figure 14. These are the rating based on which the movies are recommended to the user.

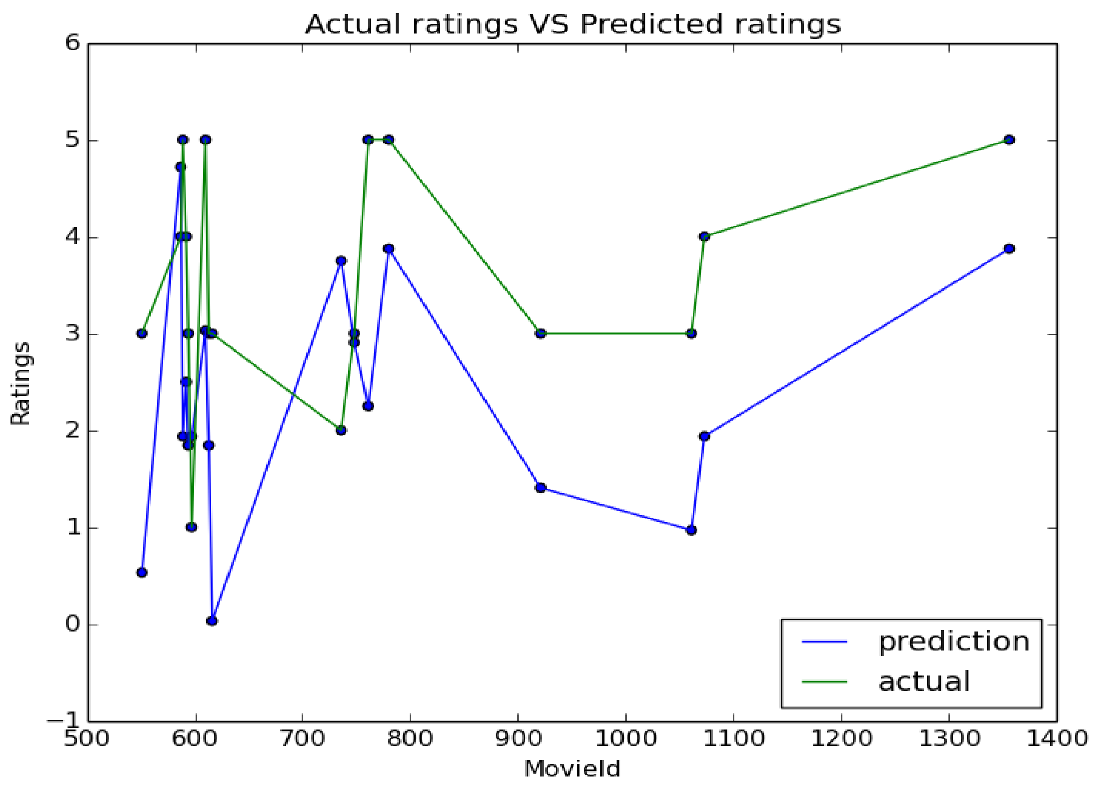


Figure 16 Graph of Predicted and Actual Movie Ratings of User profile 12

The figure 16 plots the graph of the predicted ratings and actual ratings versus their movie ids for the user profile 12. The graph shows that the pattern of both the ratings i.e., actual/prediction are nearly same. The graphs shows there some error in the predicted values. The percentage of error is calculated using the root mean square error method as described above. The percentage deviation of the predicted ratings is calculated from the method below:

$$Percentage\ Deviation = \left(\frac{RMSE}{5}\right) * 100 \ldots\ldots (13),$$

where RMSE is the root mean square error.

Table 3 RMSE for Content Based Systems

User	**RMSE**
U_{12}	1.86
U_1	1.48
U_{10}	1.27
U_{50}	1.38

Our results show an average RMSE of 1.46 and our recommendation engine has a deviation of 29.2% from actual ratings.

CHAPTER IV

IMPEMENTATION OF COLLABORATIVE FILTERING SYSTEM

A recommender system using collaborative filtering is based on the ratings of the users who have similar preferences with respect to a given customer. CF calculates the similarity between the test customer and each of other customers who have rated the items that are already rated by the test customer. Collaborative Filtering calculates the similarity between the users, but it assumes that there must exist at least two items which has already been rated by both the active user and at least one of the other users. There have been many investigations to select proper neighbours based on neighbour selection methods such as the *k* means clustering selection, the threshold-based neighbour selection, and the clustering-based neighbour selection. They are quite popular techniques for recommender systems [citation]. These techniques then predict user's preferences for the items based on the results of the cluster's evaluation on the same items.

In our approach we show that exploiting the attributes of each item improves prediction quality. We analyse the dataset and retrieve the preferences for the attributes.

4.1 SYSTEM ARCHITECTURE

The Figure 17 below describes the layout of our system for collaborative filtering recommendation system. The active user first login into the system and the system extracts his user profile. Now his user profile is used for finding the similar users. This problem of finding similar users is done by clustering the user profiles. Now active user is placed in his appropriate cluster depending on the similarity measure (Euclidean or Cosine Similarity). The cluster of the active user is exploited to find out recommendation and prediction for the active user using equation 14 and 15.

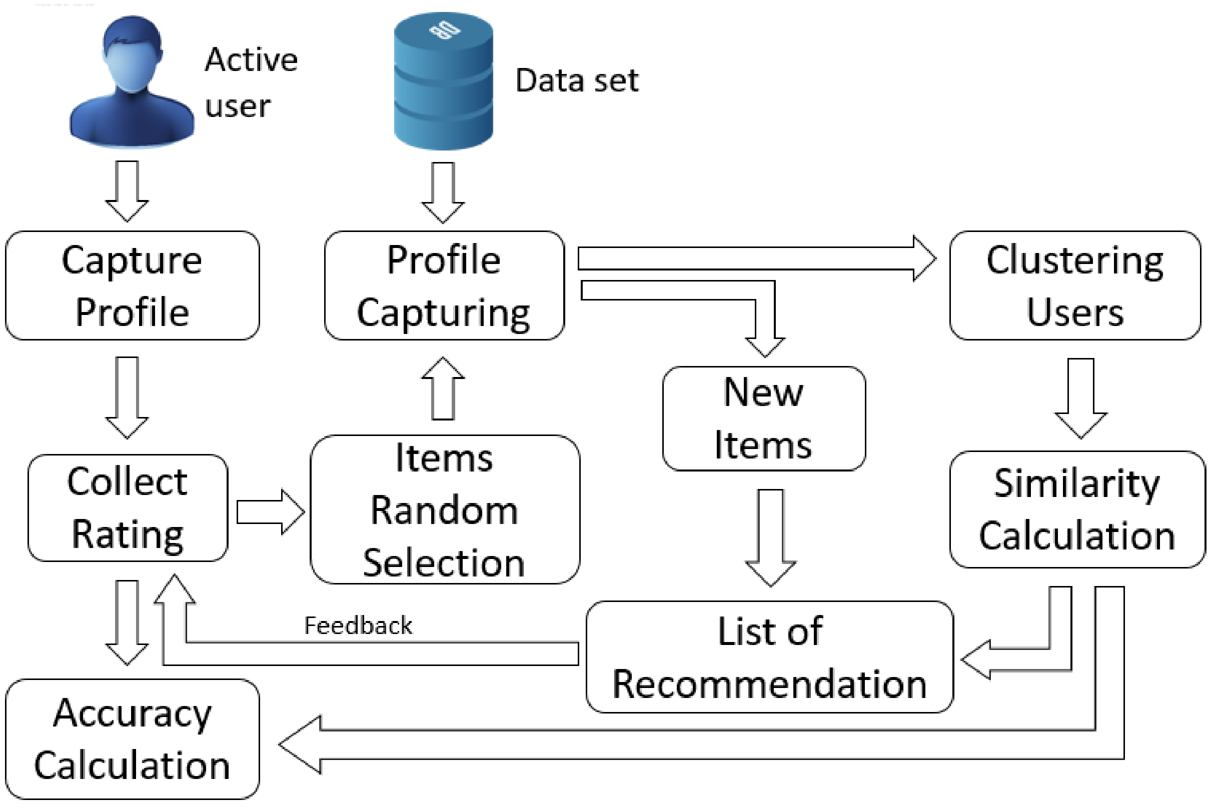


Figure 17 System Architecture for Collaborative Filtering

Steps for finding the Recommendations

We describe a user based filtering approach to recommend movies to active user. The steps for the process are described below.

- Active user's profile is extracted from the dataset.
- Selecting two random movies from the active user's profile.
- Extracting user profiles of all other users who have rated those two movies
- Perform clustering on the users based on their ratings.
- Place active user in the appropriate cluster
- Calculate similarity between active user and the users of the cluster using Pearson Correlation Coefficient ($W_{a,k}$) described below.

$$w_{a,k} = \frac{\sum_j (r_{a,j} - \overline{r_a})(r_{k,j} - \overline{r_k})}{\sqrt{\sum_j (r_{a,j} - \overline{r_a})^2 \sum_j (r_{k,j} - \overline{r_k})^2}} \ldots\ldots (14)$$

- We predict the ratings by the equation described below. These steps are repeated a number of times and average root mean square error is calculated.

$$P_{a,i} = \overline{r_a} + \frac{\sum_k \{w_{a,k} \times (r_{k,i} - \overline{r_k})\}}{\sum_k |w_{a,k}|} \ldots\ldots (15)$$

In the above equations $P_{a,i}$ is the preference of customer a with respect to item i. $\overline{r_a}$ and $\overline{r_k}$ are the averages of customer a's ratings and customer k's ratings, respectively. $r_{k,i}$ and $r_{k;j}$ are customer k's ratings for items i and j, respectively, and $r_{a;j}$ is customer a's rating for item j.

Clustering K means by Lloyd's Algorithm

K-means stores k centroids that it uses to define clusters. A point is considered to be in a particular cluster if it is closer to that cluster's centroid than any other centroid. K-Means finds the best centroids by alternating between (1) assigning data points to clusters based on the current centroids (2) choosing centroids (points which are the center of a cluster) based on the current assignment of data points to clusters.

The two-step procedure continues until the assignments of clusters and centroids no longer change. As already mentioned, the convergence is guaranteed but the solution might be a local minimum. In practice, the algorithm is run multiple times and averaged. For the starting set of centroids, several methods can be employed, for instance random assignation.

There are n points $x^{(i)}$ in the dataset R^n. μ is centroid of a cluster, $c^{(i)}$ represent cluster formed and $l\{c^{(i)} = j\}$ is the Euclidian distance/Cosine similarity measured.

Algorithm

```
1. Initialize cluster centroids $\mu_1, \mu_2, \ldots\ldots, \mu_k \in R^n$ randomly.
2. Repeat until convergence: {
   For every $i$, set,
   $c^{(i)} := \arg\min_n || x^{(i)} - \mu_j||^2$ .
   For each $j$, set,
   $$\mu_j = \frac{\sum_{i=1}^{m} l\{c^{(i)} = j\}x^{(i)}}{\sum_{i=1}^{m} l\{c^{(i)} = j\}}$$
   }
```

Calculating Accuracy

The $r'_k(u,i)$, $r_k(u,i)$ are the predicted ratings from the user profile by the proposed technique. Now to calculate the efficiency/accuracy of the prediction we use root mean square method which is described as follows:

$$RMSE = \sqrt{\frac{1}{I_{testdata}} \sum_{i \in testdata} (r_k(u,i) - a\_r(u,i))^2} \dots \dots (16)$$

where $r_k(u,i)$ is the predicted rating by the recommendation engine and $a\_r(u,i)$ is the actual rating given by the User.

4.2 EXPERIMENTAL SETUP

Operating System: Windows (7/8/10)/Linux (LinuxMint)

System Specifications: RAM of 4GB running Intel Core i-5 processor

Programming Language: Python

Libraries Used: Pandas for handling data frames, SciPy NumPy for scientific equations, matplotlib for plotting the graphs

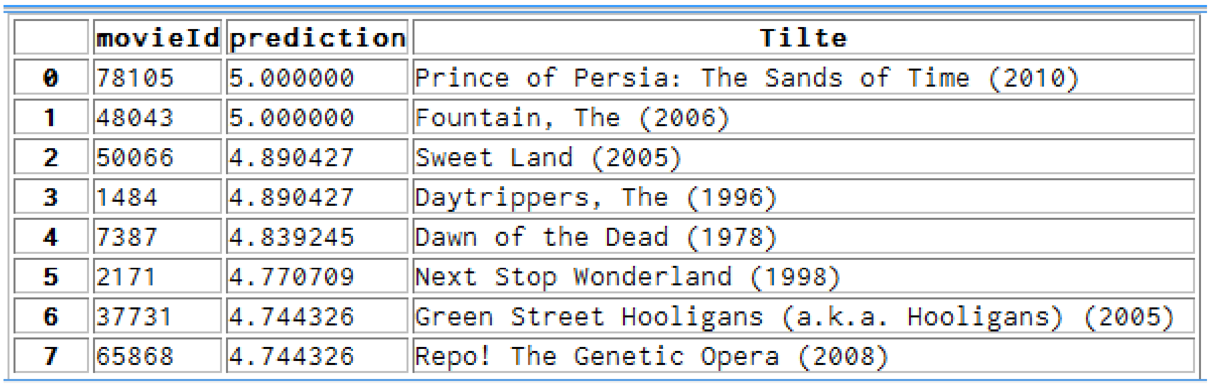
4.3 DATASET

The dataset for movie recommendations is chosen from the GroupLens Research Project at University of Minnesota. This dataset describes 5-star rating from MovieLens [6], a movie recommendation service from GroupLens. It contains 100234 ratings across 8927 movies. These data were created by users between March 26, 1996 and August 06, 2015 Users were selected at random for inclusion. All selected users had rated at least 1 movies. Each user is represented by an id, and no other information is provided.

The data are contained in four files, links.csv, movies.csv, ratings.csv and tags.csv. We use movies.csv and ratings.csv for our predictions [6].

4.4 RESULTS AND ANALYSIS

The above discussed approached is implemented on the mentioned experimental setup and the dataset. The results by the system are summarised on different users and we have predicted the ratings.



	movieId	prediction	Tilte
0	78105	5.000000	Prince of Persia: The Sands of Time (2010)
1	48043	5.000000	Fountain, The (2006)
2	50066	4.890427	Sweet Land (2005)
3	1484	4.890427	Daytrippers, The (1996)
4	7387	4.839245	Dawn of the Dead (1978)
5	2171	4.770709	Next Stop Wonderland (1998)
6	37731	4.744326	Green Street Hooligans (a.k.a. Hooligans) (2005)
7	65868	4.744326	Repo! The Genetic Opera (2008)

Figure 18 Recommendation given by the Collaborative Filtering System

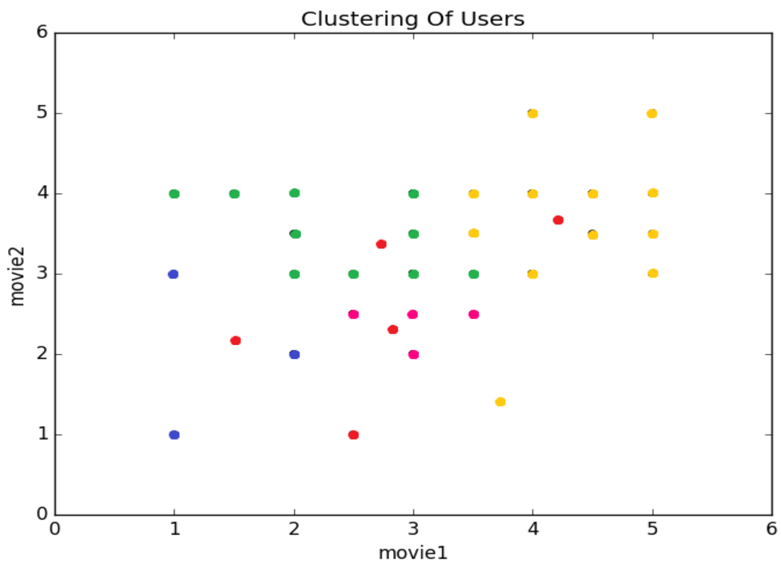


Figure 19 Results of Clusters by K means Lloyd's Algorithm

Figure 18 shows the Recommendations given by our developed system. The Figure 19 above shows the clustering of the users for an active user profile 2. Each clusters are represented by different colours and the active user profile 2 is placed in the appropriate cluster. Now the cluster of the active user is selected and with this cluster we find the recommendations for the active user by sorting the predicted ratings of the different movies.

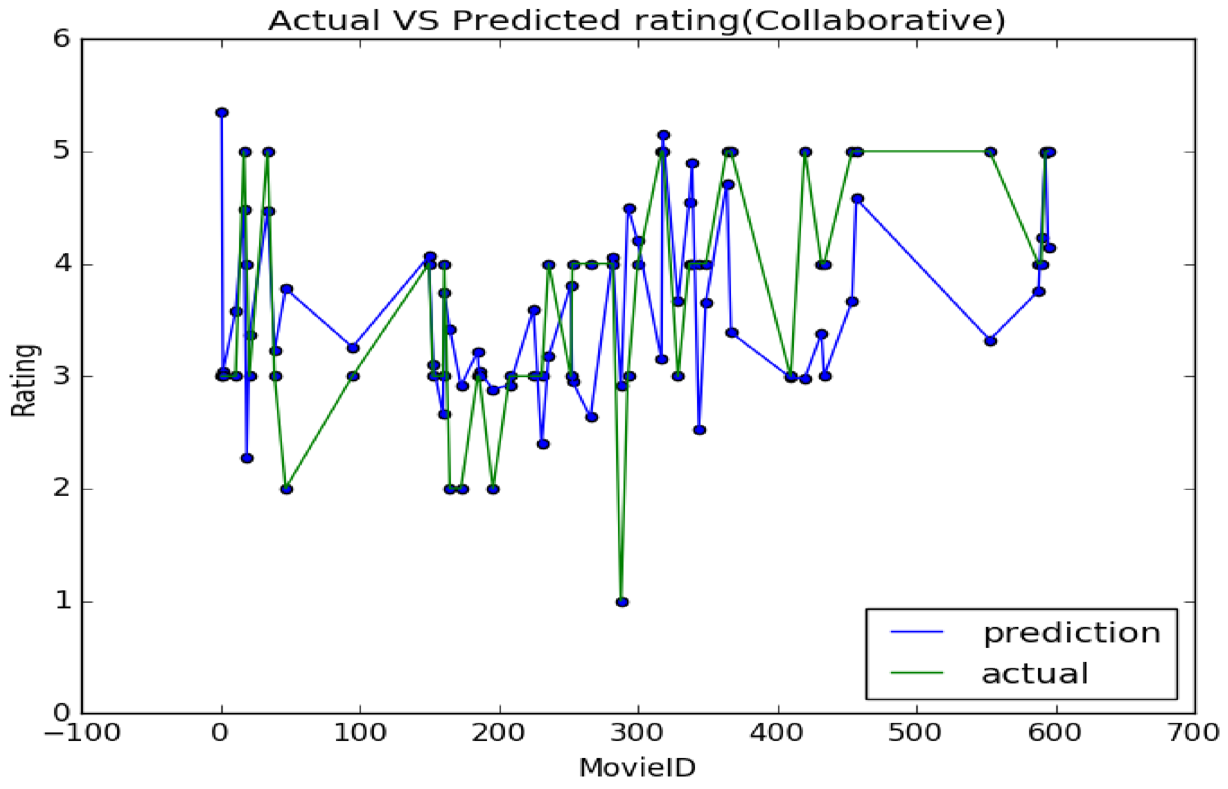


Figure 20 Actual Ratings versus Predicted Ratings by Collaborative Filtering Systems (Euclidean distance)

The Figure 20 above describes the actual ratings versus predicted ratings (by recommendation engine). The system predicts the actual ratings by ***Euclidean Distance*** for similarity. The average Root Mean Square Error (RMSE) is calculated from the results to find the accuracy of the system. Below Table 4 summarises the average RMSE of the different user profiles with different number of clusters.

Table 4 Average RMSE for different Users with different number of Clusters (Euclidean distance)

User	No of Clusters = 3	No of Cluster = 5	No of Clusters = 7
U_1	0.55	0.67	0.58
U_2	0.83	0.83	0.77
U_5	1.52	1.04	1.08
U_{12}	1.22	1.29	1.17

Our results show an average RMSE of 0.88. Our recommendation engine has a deviation of 19.25% from actual ratings by Pearson Correlation Coefficient measure and clusters are built by Euclidean distance.

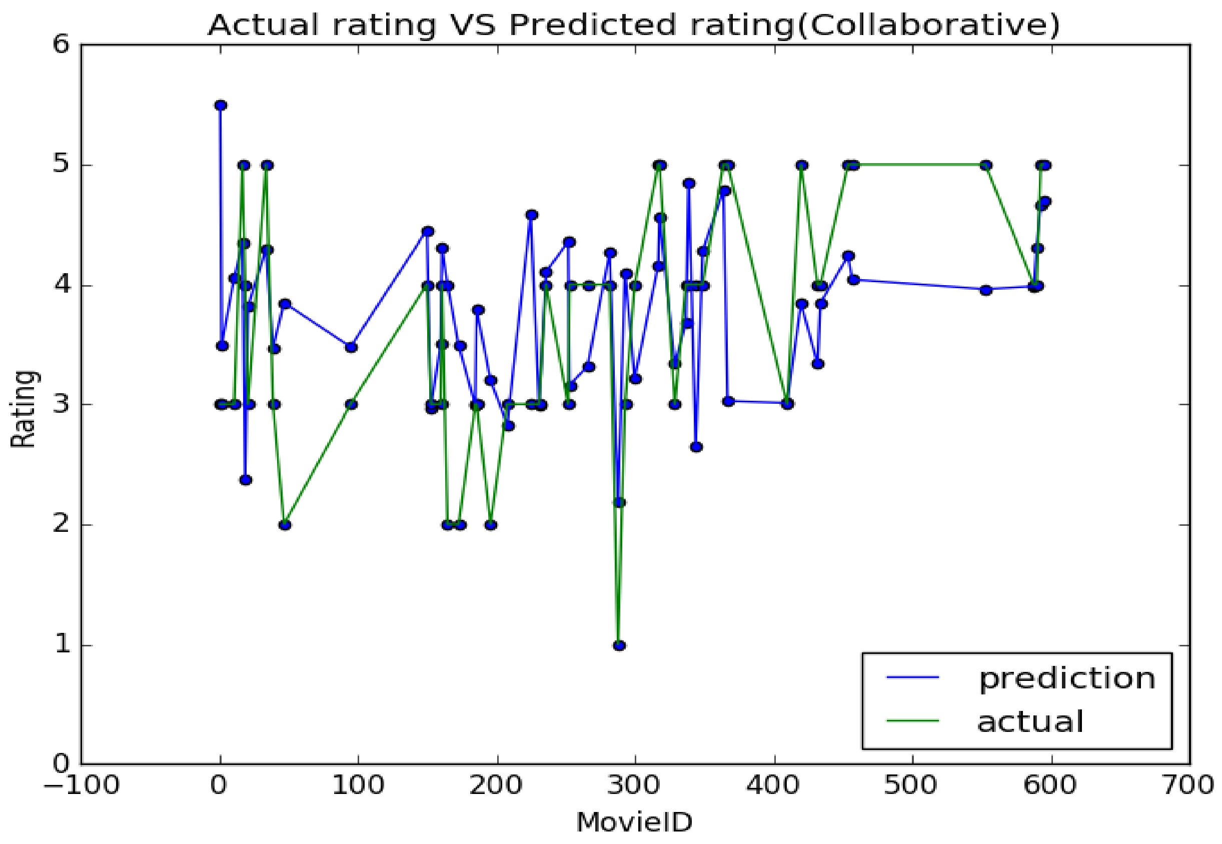


Figure 21 Actual Ratings versus Predicted Ratings by Collaborative Filtering Systems (Cosine Similarity)

The above Figure 21 describes the actual ratings versus predicted ratings (by the recommendation engine). Here our system uses ***Cosine Similarity*** measure for finding the similar users of the active user. Again the average RMSE is calculated for the system to find the accuracy. This system has been implement on different users with different clusters and the results are summarised in the Table 5 below.

Table 5 Average RMSE for different Users with different number of Clusters (Cosine Similarity)

User	No of Clusters = 3	No of Clusters = 5	No of Clusters = 7
U_1	0.63	0.62	0.60
U_2	0.93	1.04	0.90
U_5	1.17	1.09	1.53
U_{12}	1.12	1.43	1.17

Our results show an average RMSE of 1.01. Our recommendation engine has a deviation of 20.2% from actual ratings. Cosine Similarity measure is used building clusters and correlation between users is calculated by Pearson Correlation Coefficient.

4.5 COMPARING RESULTS OF COLLABRATIVE FILTERING AND CONTENT BASED

From our results we can conclude saying that Collaborative filtering gives better results & accuracy than the content based system. In the Collaborative based system using Euclidean distance to measure similarity has given better results than the Cosine similarity, Hence we can conclude that Euclidean distance is a better distance measure than Cosine Similarity for our system.

Table 6 Comparison of both developed Recommendation Systems

	Content based system	Collaborative filtering (Euclidean)	Collaborative filtering (Cosine)
Average RMSE	1.46	0.88	1.01
Percentage Deviation	29.2	19.25	20.2

CHAPTER V
IMPLEMENTATION OF COLD START PROBLEM

A popular and effective approach to recommendations is collaborative filtering, which focuses on detecting users with similar preferences and recommending items favoured by the like-minded. However, the system would fail to provide recommendations when a new user arrives and no information is gathered from a new user, which is known as the cold-start problem [12]. Rapid profiling of new users is a key challenge in cold start recommender systems. One direct way is going through an initial interview process. In the interview, the recommender system asks users to provide opinions on selected items and constructs user profiles. Asking too few questions may lead to inaccurate estimation of user profiles and the system is unable to provide satisfactory recommendations, while asking too many questions may cause users to leave the interview. A good interview targets user interests with minimum interactions. Specifically, the process should focus on (1) increasing the recommendation accuracy and (2) minimizing user interaction efforts.

5.1 SYSTEM ARCHITECTURE

The Figure 23 below describes the layout of our system for cold start problem. The New user first login into the system and the system creates interview questions for the user. The movies are selected on the basis of most preferred movies seen by the set of users in the database. The most preferred movie is calculated by the equation[10]: 17,18, 19.The movie that gives the minimum value for the equation is asked to the user and this movie is splitting criteria of the decision node and based on user reply (Like, Hate, Unknown), again the user subset is created. The process of computation is repeated for certain number of times to get an appropriate recommendation for the user.

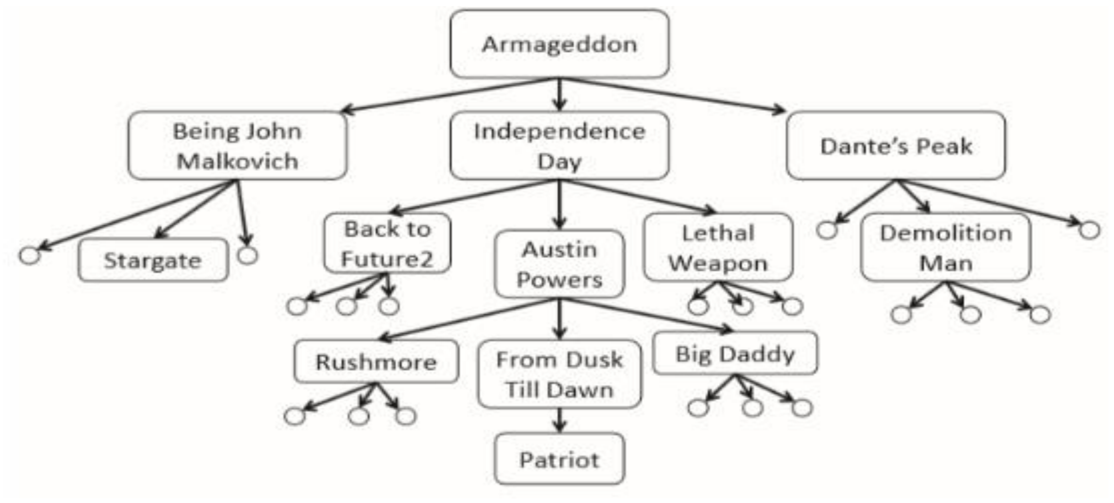


Figure 22 Decision Tree solution for Cold Start Problem [12]

Let t be a tree node and St ⊆ U be its associated set of users. The mean by the node t to item i is:

$$u(t)_i = \frac{1}{|S_t \cap R(i)|} \sum_{u \in S_t \cap R(i)} r_{ui} \dots \dots (17)$$

Where $S_t \cap R(i)$ is number of users who have rated item I, r_{ui} is the rating of item I given by user u.

The Squared error associated with node t for item i is:

$$e(t)_i^2 = \sum_{u \in S_t \cap R(i)} (r_{ui} - u(t)_i)^2 \quad \dots \dots (18)$$

The node splitter is the item that minimizes the squared error equation.

$$splitter\ (t) \stackrel{\text{def}}{=} \arg\min\ e(t)_i^2 \dots \dots (19)$$

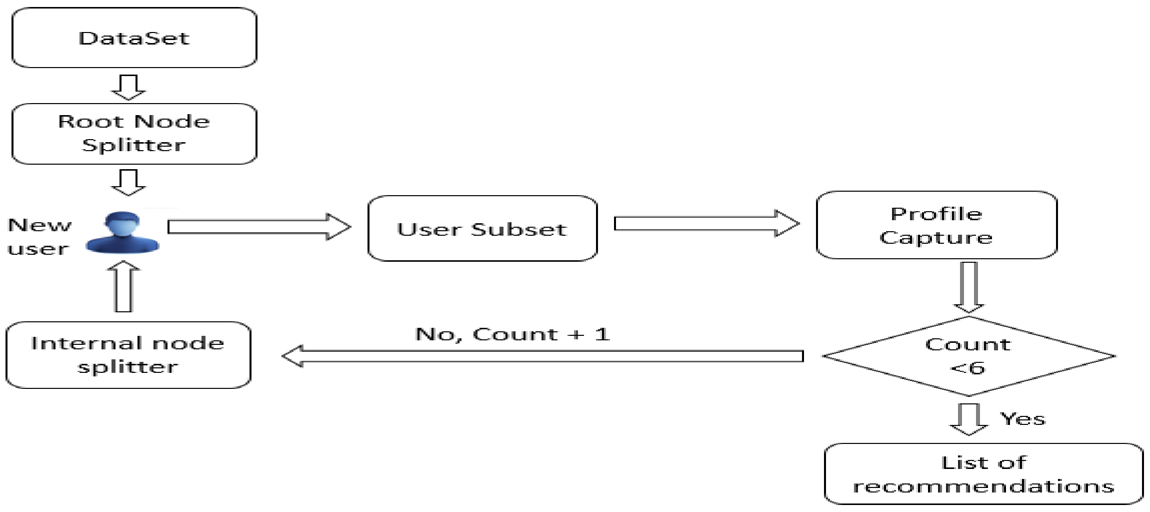


Figure 23 System Architecture for Cold Start Problem

5.2 EXPERIMENTAL SETUP

Operating System: Windows (7/8/10)/Linux (LinuxMint)

System Specifications: RAM of 4GB running Intel Core i-5 processor

Programming Language: Python

Libraries Used: Pandas for handling data frames, SciPy NumPy for scientific equations, matplotlib for plotting the graphs

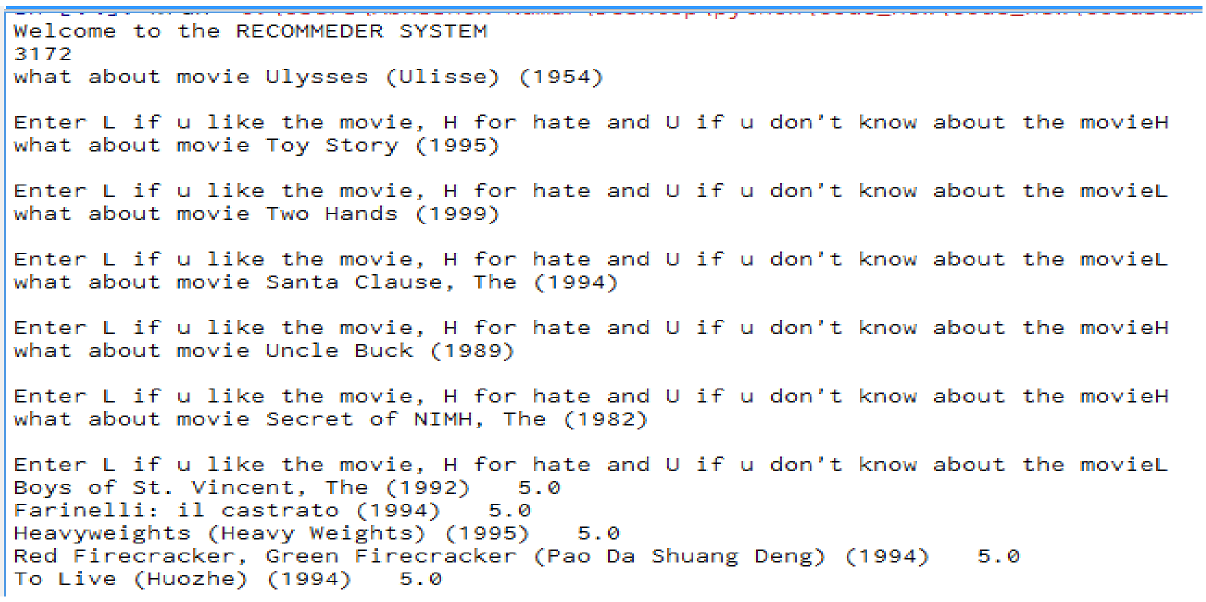
5.3 DATASET

The dataset for movie recommendations is chosen from the GroupLens Research Project at University of Minnesota. This dataset describes 5-star rating from MovieLens [6], a movie recommendation service from GroupLens. It contains 100234 ratings across 8927 movies. These data were created by users between March 26, 1996 and August 06, 2015 Users were selected at random for inclusion. All selected users had rated at least 1 movies. Each user is represented by an id, and no other information is provided.

The data are contained in four files, links.csv, movies.csv, ratings.csv and tags.csv. We use movies.csv and ratings.csv for our predictions [6].

5.4 RESULTS

The above discussed approached is implemented on the mentioned experimental setup and the dataset. The interview questions and recommendation based on interview is shown below.



```
Welcome to the RECOMMEDER SYSTEM
3172
what about movie Ulysses (Ulisse) (1954)

Enter L if u like the movie, H for hate and U if u don't know about the movieH
what about movie Toy Story (1995)

Enter L if u like the movie, H for hate and U if u don't know about the movieL
what about movie Two Hands (1999)

Enter L if u like the movie, H for hate and U if u don't know about the movieL
what about movie Santa Clause, The (1994)

Enter L if u like the movie, H for hate and U if u don't know about the movieH
what about movie Uncle Buck (1989)

Enter L if u like the movie, H for hate and U if u don't know about the movieH
what about movie Secret of NIMH, The (1982)

Enter L if u like the movie, H for hate and U if u don't know about the movieL
Boys of St. Vincent, The (1992)    5.0
Farinelli: il castrato (1994)    5.0
Heavyweights (Heavy Weights) (1995)    5.0
Red Firecracker, Green Firecracker (Pao Da Shuang Deng) (1994)    5.0
To Live (Huozhe) (1994)    5.0
```

Figure 24 Recommendations by the cold start problem for a new user by interactions

CHAPTER VI
CONCLUSION AND FUTURE WORK

In our project we have developed two systems, a content based recommendation system and a collaborative recommendation system.

Content-based recommendation system makes recommendations according to the users' past watching behaviour. Our system produces recommendations based on genre feature set. We have built a mathematical model to predict the ratings and based on the predicted ratings, the system recommends top six movies to the user which he has not watched yet.

In collaborative filtering system we have done clustering by Euclidean distance and Cosine similarity to group similar users and similarity between users of a cluster to active user is used to recommend movies to the active user. We used Pearson Correlation Coefficient to calculate correlation between users.

Our results show that collaborative filtering systems give better recommendations than content based systems. Accuracy of the systems can be increased if more number of ratings are included. Accuracy also depends upon the similarity measure chosen in developing the systems like our collaborative filtering gives more user interested recommendations in case of Euclidean distance than Cosine similarity. Decision Tree approach is a liable solution for the cold start problem as proposed.

For future work, Accuracy of a recommendation system can be increased by building hybrid techniques like combining the content based and collaborative filtering, or with demographic and so on. In each recommendation system accuracy can also be increased by trying out different profile learning algorithms.

REFERENCES

[1] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor. 2010. '*Recommender Systems Handbook'* (1st ed.). Springer-Verlag New York, Inc., New York, NY, USA.

[2] J. Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. 2007. '*Collaborative filtering recommender systems'*. In *The adaptive web*, Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl (Eds.). Lecture Notes In Computer Science, Vol. 4321. Springer-Verlag, Berlin, Heidelberg 291-324.

[3] Loren Terveen and Will Hill. '*Beyond Recommender Systems: Helping People Help Each Other'*. In Jack Carrol, editor, HCI In The New Millenium, pages 1–21, Massachusetts, 2001. Addison-Wesley Verlag.

[4] Rajhans Mishra, Pradeep Kumar, Bharat Bhasker, '*A web recommendation system considering sequential information'*, Decision Support Systems, Volume 75, July 2015, Pages 1-10, ISSN 0167-9236.

[5] Uluyagmur, Mahiye, and Esengul Tayfur. *"Content-based movie recommendation using different feature sets."* Proceedings of the World Congress on Engineering and Computer Science. Vol. 1. 2012.

[6] Available at: www.grouplens.org.

[7] Kim, Taek-Hun, and Sung-Bong Yang. "Using attributes to improve prediction quality in collaborative filtering." *E-Commerce and Web Technologies*. Springer Berlin Heidelberg, 2004. 1-10.

[8] Sarwar, Badrul M., et al. "Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering." *Proceedings of the fifth international conference on computer and information technology*. Vol. 1. 2002.

[9] Kuzelewska, Urszula. "Clustering Algorithms in Hybrid Recommender System on MovieLens Data." *Studies in Logic, Grammar and Rhetoric* 37.1 (2014): 125-139.

[10] Golbandi, Nadav, Yehuda Koren, and Ronny Lempel. "Adaptive bootstrapping of recommender systems using decision trees." *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 2011.

[11] Sarwar, Badrul, et al. "Item-based collaborative filtering recommendation algorithms." *Proceedings of the 10th international conference on World Wide Web*. ACM, 2001.

[12] Sun, Mingxuan, et al. "Learning multiple-question decision trees for cold-start recommendation." *Proceedings of the sixth ACM international conference on Web search and data mining*. ACM, 2013.

ACKNOWLEDGMENT

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals. We would like to extend our sincere thanks to all of them.

We are highly indebted to our mentor ***Dr. Udai Pratap Rao***, Asst. Prof, Computer Engineering Department, SVNIT for his guidance and constant supervision as well as for providing necessary information regarding the project. His constant support in starting the project has helped us in doing a lot of Research and we came to know about so many new things. We like to thank our Head of the Department ***Dr. D. R. Patel*** for his constant encouragement and motivating the final year students. We are also thankful to our project coordinator ***Dr. M. A. Zaveri*** for scheduling the project related activities.

Our thanks and appreciation also goes to fellow students in developing the project and people who have willingly helped us out with their abilities.