VAAL UNIVERSITY OF TECHNOLOGY

Faculty of Applied and Computer Sciences

Virtual Group Movie Recommendation System Using Social Network Information

Dissertation

Submitted in partial fulfilment of the requirements for the degree

MAGISTER TECHNOLOGIAE

In the Department of Information and Communication Technology

Student Name: Lefats'e Manamolela

Student Number: 218255128

Supervisors: Professor Tranos Zuva, DR. Apiah Martin

27th November 2019

DECLARATION

I solely declare that this dissertation is an original report of my research, has been composed by myself and has not been submitted for any previous degree. The experimental work is almost entirely my own work; the collaborative contributions have been indicated clearly and acknowledged. Due references have been provided on all supporting literatures by means of comprehensive list of references.

I further state that I am aware of and understand the university's policy on plagiarism and I certify that all the policies regarding plagiarism have been adhered to.

Signed [illegible] on this 27th day of November 2019

ACKNOWLEDGEMENTS

Firstly, I have to give special thanks to the almighty God for making it all possible. I would also like to express my deep gratitude to my master's dissertation supervisors Professor Tranos Zuva and Doctor Apia Martin. When Professor Zuva introduced me to the exciting world of recommendation systems, I did not have much knowledge in this area only to find that with time I would develop a deep love for the field. He would spend countless hours guiding me on how to choose the area to focus on, how to search the literature, how to come up with the topic, things to consider during the design and development phase of the prototype and how to write a paper. I am grateful to Doctor Apia Martin for spending time reading this dissertation and providing useful suggestions about this dissertation. Having been under the leadership of Professor Tranos Zuva and Doctor Apia Martin for the past few months, I have indeed learned a lot. They are both hard-working professionals and I believe their academic achievements will continue to increase.

During this period of one year, my friends have also been helpful. I have to acknowledge all my colleagues in research room T 0-16 of the Vaal University of technology for their assistance in many aspects that cannot be listed here because of the space.

Life at a graduate institution, as a student is not always great. It is very fortunate, and blessed of me to have the kind of parents I have by my side. I owe much more than thanks to my parents, MR Thabiso Manamolela and MRS Mamahlomola Manamolela for their emotional and financial support. Without them, it was not going to be possible for me to finish my dissertation and graduate masters impeccably. They have always been there to encourage me throughout my life.

CONTENTS

GLOSSARY

The following Abbreviations are used in this study:

AI	Artificial Intelligence
ANN	Artificial Neural Network
CBF	Content-Based Filtering
CF	Collaborative Filtering
HF	Hybrid Filtering
IDE	Integrated Development Environment
KDE	Kernel Density Estimation
KNN	K-Nearest Neighborhood
LDA	Latent Dirichlet Allocation
LSA	Latent Semantic Analysis
MAE	Mean Absolute Error
MLP	Multi-Layer Perceptron
PCA	Principle Component Analysis
RMSE	Root Means Square Error
RS	Recommendation Systems
SNSs	Social Network Sites
SVD	Singular Value Decomposition
TF-IDF	Term Frequency-Inverse Document Frequency

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Since their emergence in the 1990's, recommendation systems have transformed the intelligence of both the web and humans. A pool of research papers has been published in various domains of recommendation systems. These include content based, collaborative and hybrid filtering recommendation systems. Recommendation systems suggest items to users and their principal purpose is to increase sales and recommend items that are predicted to be suitable for users. They achieve this through making calculations based on data that is available on the system. In this study, we give evidence that the research on group recommendation systems must look more carefully at the dynamics of group decision-making in order to produce technologies that will be more beneficial for groups based on the individual interests of group members while also striving to maximise satisfaction. The matrix factorization algorithm of collaborative filtering was used to make predictions and three movie recommendation for each and every individual user. The three recommendations were of three highest predicted movies above the pre-set threshold which was three. Thereafter, four virtual groups of varied sizes were formed based on four highest predicted movies of the users in the dataset. Plurality voting strategy was used to achieve this. A publicly available dataset based on Group Recommender Systems Enhanced by Social Elements, constructed by Lara Quijano from the Group of Artificial Intelligence Applications (GIGA), was used for experiments. The developed recommendation system was able to successfully make individual movie recommendations, generate virtual groups, and recommend movies to these respective groups. The system was evaluated for accuracy in making predictions and it was able to achieve 0.7027 MAE and 0.8996 RMSE. This study was able to recommend to virtual groups to enable social network group members to engage in discussions of recommended items. The study encourages members in engaging in similar activities in their respective physical locations and then discuss on social network.

1 INTRODUCTION OF THE STUDY

1.1 BACKGROUND

Recommendation systems or recommender systems (RSs) are software tools and techniques aimed at providing suggestions to support users in various decision-making processes such as what items to buy, what music to listen, or what news to read (Ricci, et al., 2010). With the rapid development of the internet, more and more online services inevitably suffer from information overload, which makes it difficult for users to find the information they require (Guo, 2017). Recommendation systems have proven to be valuable means for online users to cope with the information overload and have become one of the most powerful and popular tools in electronic commerce (Ricci, 2011). Recommendation systems provide assistance to users by identifying items that match a user's needs, preferences, and goals from a usually long list of potentially interesting items. Several recommendation techniques have been proposed in the literature (Ricci, et al., 2015).

Content-based Filtering(CBF) tries to recommend items similar to those the user has liked in the past, by considering their features (Lops, et al., 2010); (Pazzani & Billsus, n.d.); The common approach is to represent both the users and the items under the same feature space. In this space similarity scores could be computed between users and items. The recommendation is made based on the similarity scores of a user towards all the items. The Content-based Filtering methods usually perform well when users have plenty of historical records for learning.

Collaborative Filtering (CF) is an approach of automatically predicting (filtering) the interests of a user by collecting interests from many related users (collaborating). Collaborative Filtering methods are usually adopted when the historical records for predicting are scarce. Collaborative filtering or recommendation systems use a database about user preferences to predict additional topics or products a new user might like (Breese, et al., 1998). As one of the most successful approaches to building recommendation systems, collaborative filtering use the known preferences of a group of users to make recommendations or predictions of the unknown preferences for other users (Su & Khoshgoftaar, 2009).

Hybrid recommendation systems are based on the combination of the content based collaborative filtering techniques. A hybrid system combines collaborative and content based filtering techniques and tries to use the advantages of CBF to fix the disadvantages of CF. For instance, CF experiences problems with new items, i.e., it cannot recommend items that have no ratings. This does not limit content-based approaches since the prediction for new items is based on their descriptions (features) that are typically easily available. Given two (or more) basic RSs techniques, several ways have been proposed for combining them to create a new hybrid system (Ricci, et al., 2010); (Zhongqi, et al., 2015); (Kim, et al., 2006); (Robin, 2002); (Salehi, 2013).

Social network sites are web-based services that allow individuals to (1) construct a public or semi-public profile within a bounded system, (2) articulate a list of other users with whom they share a connection, and (3) view and traverse their list of connections and those made by others within the system. The nature and nomenclature of these connections may vary from site to site (Boyd & Ellison, 2008). (Abhyankar, 2011) iterates that Social Networking sites have conventional features. Every user registered with a social networking site creates his profile containing some basic information with very little time and effort. He can add new contacts, upload pictures and/or audio/videos, set status messages, post comments, join various groups of people that share common interests, join forums for discussion etc.

While the majority of recommendation systems suggest items based on the preferences of an individual consumer, group recommendation systems suggest items taking into account the preferences and personalities of the members of a group (Jameson & Smyth, 2017). In order to generate effective recommendations for a group the system must satisfy, as far as possible, the individual preferences of the members in the group (Baltrunas, et al., 2010).

1.2 PROBLEM STATEMENT

A variety of techniques have been proposed for generating recommendations, including content-based filtering (CBF), collaborative filtering (CB) and hybrid recommendation systems (Burke, 2007); (Porcel et al., 2012); (Williams, et al., 2007). These techniques assist users in finding information, products, or services (such as books, movies, music, digital products, Web sites, and TV programs, to name a few), by aggregating and analyzing suggestions from other users, reviews from various authorities, and user attributes (C). Collaborative filtering (CF) is known to be a successful recommendation technique (Al-Barznjl & Atanassov, 2017). It makes recommendations to users based on other users' ratings on items, putting more weights on those from similar users (i.e., other users having similar personal attributes or product preferences).

To date, recommendation systems have focused mainly on recommending items to individuals rather than groups of people intending to participate in a group activity (Kim, et al., 2010). In recommendation domains such as shopping and asset investment, it is not a limitation because users in general behave individually and only their personal interests should be considered. In other domains such as movies, trips, book clubs, and restaurants, however, existing recommendation systems have difficulty in aggregating individual users' tastes into a group's preference properly.

From this information, it is clear that content based filtering has some limitations. These limitations can be countered by using the strengths of collaborative filtering to solve the weaknesses of content based filtering. The study investigates existing filtering techniques, hence applying the collaborative filtering technique to recommend movies to groups of participants.

This research therefore attempts to make use of the advantages of the collaborative filtering technique to overcome the limitations of the content-based filtering technique to recommend movies to individuals, and eventually making movie suggestions to diversified virtual groups using information gathered from social network, Facebook.

1.3 RESEARCH QUESTION

Primary research question:

▪ How can a virtual group movie recommendation system be developed using social network information?

In order to answer the main question, the following secondary questions were answered:

a. What recommendation algorithms have been used in literature?

b. How can the system be developed in a way that it meets the needs of individuals in a virtual group?

c. How will the effectiveness of the developed system be measured?

1.4 RESEARCH PROJECT OBJECTIVES

The objectives of this research were:

a. To investigate recommendation systems algorithms in literature.

b. To propose an algorithm that can be used for virtual group movie recommendation system using social network data.

c. To develop a prototype for virtual group movie recommendation system.

d. To evaluate the prototype for the effectiveness.

1.5 Research Justification

Recommendation systems are present in many web applications to guide our choices. They increase sales and benefit sellers, but whether they benefit customers by providing relevant products is questionable (Yeung, 2015). The relevance of this proposal is to assist diversified groups of users by engaging them in the process of movie selection, acquiring information from their social network profiles, making individual movie recommendations, and ultimately making movie suggestions for their respective diversified groups with the aim of meeting an acceptable level of member satisfaction.

The result of this research highlighted challenges with current recommendation systems and outlined propositions on resolving such challenges.

Movie recommendation systems have been beneficial to viewers for years. The virtual group movie recommendation system will be of paramount importance and change user’s perception by providing the necessary information, not just to a single user, but also to diversified groups of viewers who will be watching movies together, in form of virtual groups. The system carried out series of calculations with the aim of reaching an acceptable level of satisfaction for all the virtual group members. The objective is that with time, and as technology evolves, the proposed virtual group movie recommendation system will be updated by either adding or removing features so that it may adopt to technological changes.

1.6 Significance Of The Study

The utilization of recommendation systems, specifically movie recommendations systems has been studied with various recommendation systems approaches employed. However, this study is expected to make a major contribution towards how recommendation systems recommend movies to users. The study further focused on how individual user ratings can be considered when making group movie recommendations. In addition, the study attempted to form virtual groups based on the similarities of users in terms of their ratings. The aim was to recommend a movie to each of the formed virtual groups.

1.7 Scope And Limitations Of The Study

Despite the fact that this research was sensibly planned, the author acknowledges the fact that some restrictions arose. Those restrictions are outlined as follows:

The research was conducted based on the data that is collected from the Facebook application known as HappyMovie. The research was conducted over a period of two years. It was based 50 movies and 58 users. It is therefore evident that 50 movies might be a significantly low number, and 58 users is not that much either.

1.8 Research Design

In this study, literature was explored after which the experiments were run using publicly available dataset based on Group Recommender Systems Enhanced by Social Elements, constructed by Lara Quijano from the Group of Artificial Intelligence Applications (GIGA) obtained at http://gaia.fdi.ucm.es/research/happymovie/download. The dataset consists of a sample of 58 users and 50 movies selected from the MovieLens dataset. Data was pre-Processed, analyzed, and interpreted by the researcher. The pre-processed data was thus used in experiments to produce individual movie recommendations to the users, after which virtual groups were formed and movies were recommended to those groups. Jupyter notebook IDE was used for experiments.

1.9 Dissertation Layout

The Master's dissertation chapters are structured as follows:

1: Introduction of the study;

2: Literature review;

3: Research methodology;

4: Experiments, Results and interpretation;

5: Conclusion and future work.

1.10 Publications

The following publications were extracted from this study:

- A Survey of Content-Based Filtering Technique for Personalized Recommendations
- Collaborative Filtering Recommender System Algorithms, Strengths and Open Issues
- A collaborative Filtering Based Approach Enhanced by Matrix Factorization for Group Movie Recommendations
- Aggregation Strategies for Group Recommendations

2 LITERATURE REVIEW

2.1 INTRODUCTION

This chapter seeks to clarify between different approaches to recommendation systems in literature. Looking at the field through methodologies of content-based filtering, collaborative filtering and hybrid filtering techniques, this literature review demonstrates how these different approaches have particular strengths and weaknesses in their historiography, as well as their modes of aesthetic and formal analysis. In so doing, the literature review focuses on both individual and group recommendation systems.

2.2 RECOMMENDATION SYSTEMS

Recommendation Systems (RS) are computer programs and methods that give that suggest items to clients or users. A suggestion can be within number of decision-making procedures, including but not limited to, which things to buy, music to listen to, which movie to watch, which holiday destination to visit, which restaurant to visit of which online news article to read. The term "Item" in this case is used as a representation of what the system suggests to users. RSs usually focus on a particular kind of item, for example, a news RS, its design, its graphical interface, and the main recommendation method used to produce the suggestions are all personalized to provide suggestions that are suitable and effective for a particular kind of item (Ricci, et al., 2011).

Over the years, and in various web domains, RSs have gained popularity. Their main objective is to provide recommendations to users based on their needs. A RS normally focuses on a certain specified type of item, its design, its graphical user interface and the principal method of making suggestions used to produce the suggestions are all customized to deliver beneficial and effective recommendations for that specific type of item. With time, the importance of such suggestions has increased hugely. Because of this vast increase, RSs have become autonomous research field since the mid-1990s (Adomavicius & Tuzhilin, 2005).

RSs are now widely used within the business sector and in the research community. Consequently, a lot of techniques for making suggestions are being proposed. This implies that a developer of the RS is compelled to select amongst a number of candidate methods. A first step towards selecting a suitable algorithm is to decide which properties of the application to focus upon when making this choice. Indeed, recommendation systems have a variety of properties that may affect user experience, such as accuracy, robustness, scalability, and so forth (Shani & Gunawardana, 2011).

RSs in a web-based environment suggests items to a user based on item ratings and similarity. In recent years, researchers have revealed that using RS tends to result in improved sales volumes both in the short term and in the long term or help to increase sales diversity by directing customers to other parts of the available product catalog (Fleder & Hosanagar, 2009). In a recommendation system application, there are two classes of entities which are referred to as users and items. Users have preferences for certain items, and these preferences must be teased out of the data (Koren, et al., 2009).

(Balabanovic & Shoham, 1997) classified recommendation systems into 3 major categories as shown by figure 2.1. The classification is made based on how the recommendations are made. Content-based recommendation systems recommend items similar to the ones the user preferred in the past; Collaborative recommendation systems recommends items that people with similar tastes and preferences liked in the past; Hybrid approaches recommendation systems combine collaborative and content-based methods to make recommendations.

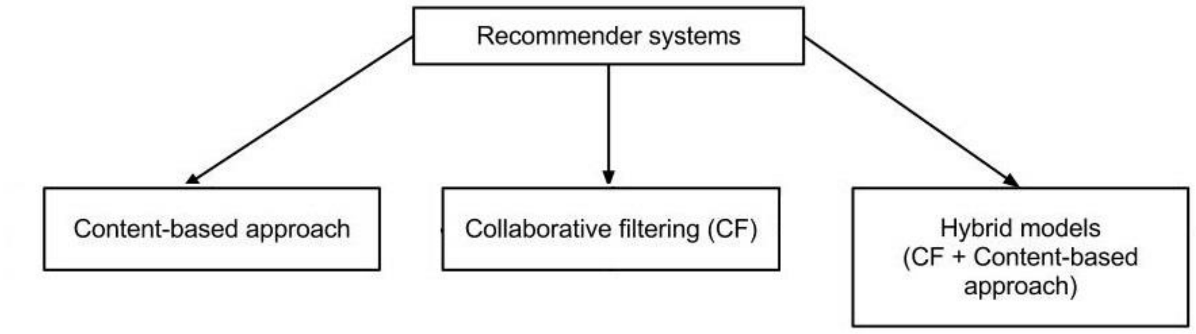


Figure 2.1 Filtering techniques in recommendation systems adapted from (Al-Barznji & Atanassov, 2018).

2.3 RECOMMENDATION SYSTEMS TECHNIQUES

Employing accurate and efficient RS is vital for a system that is intended to suggest good and helpful items to its users. It explains the vitality of comprehending the potentials and features of various techniques of making recommendations (Isinkaye, et al., 2015). The three main recommendation techniques are content based filtering, collaborative filtering and hybrid filtering.

2.3.1 CONTENT BASED FILTERING

Content-based recommendation systems attempt to suggest items alike to those a given user has previously liked or rated positively. CBF system matches 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 (Lops, et al., 2010). In essence, Content-based methods make recommendations by analyzing the description of the items that have been rated by the user and the description of items to be recommended (Pazzani, 2000).

CBF systems are centered on the perception that users rate similarly items with alike neutral features. It makes use of the content to analyze using a suitable model, it recommends the item that is equivalent to the user profile (Hameed, et al., 2012). CB recommendation system attempts to recommend items that are similar to those items a particular user has liked in the past. It mainly entails pairing up the features of the user profile versus the feature of a content of the object (Lops, et al., 2010). The feature holds small numeric values signifying certain parts of the item such as price or color, etc. some dis-similarity gages between the features vectors can be used to calculate the similarity of two objects/items.

2.3.1.1 Content-Based Filtering Architecture

There are three principal components of content-based filtering namely content analyzer, a profile learner and a filtering component (Ricci, et al., 2011) as demonstrated by figure 2.2.

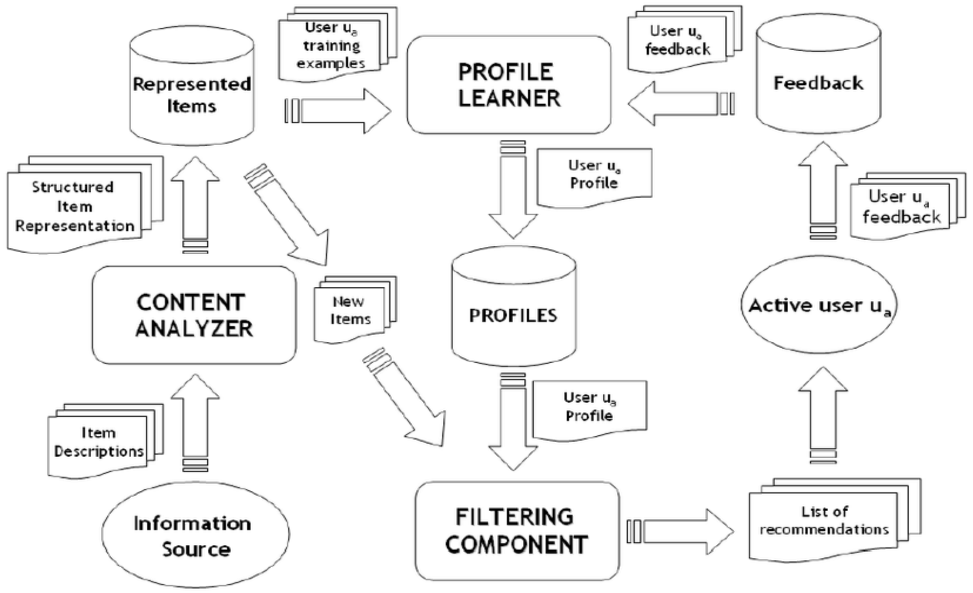


Figure 2.2 Content-based recommendation system high level architecture (Ansgar, et al., 2015)

Content Analyser- It is the first step of the recommendation process. The role of this component is to process and present the content of items in the format that could be easily understood by the next processing component. This is because some information like text has no structure so it needs to be processed so as to extract structured relevant information. Feature extraction techniques which are usually borrowed from Information Retrieval (IR) systems are used to analyse data items in order to transform it from its original information space to the target one.

Profile Learner- This component is the one responsible for building user profile. It collects data representation of user preferences and tries to generalize this data. The Machine Learning techniques are usually employed to derive a model of user interests based on the items the user liked or disliked previously.

Filtering Component- This module is responsible for matching the users' profile against that of the candidate items and then suggests relevant items. Similarity metrics are employed in this regard for an example cosine similarity can be used to match the similarity between the prototype and the item vectors.

Building User Profile

Generally, the process of inducing user profiles in Content-based is achieved by using Machine Learning techniques which are well-suited for text categorization (Sebastiani, 2002). This process can be cast as a binary text categorization task using the Machine Learning techniques (Sebastiani, 2002). This Machine Learning approach automatically builds a text classifier by learning the features of the categories from a set of training documents. The documents are categorized into two, based on the users' preference as to whether they are interesting or not to him. The set of categories can be written as $c = \{c^+, c^-\}$, where c^+represents the user-likes and c^- represents user-dislikes.

There are many learning algorithms employed in Content-based Recommendation Systems (Sebastiani, 2002); (Montaner, et al., 2003); (Mooney & Roy, 2000). However, in this context the research will discuss the probabilistic method in Na¨ıve Bayes and a Relevance Feedback method.

Na¨ıve Bayes- This is a Bayesian classifier probabilistic approach to inductive learning. It utilizes the previously presented data to generate a probabilistic model (Lops, de Gemmis & Semeraro, 2011). Naïve Bayes classifier draws its applications from the Bayes theorem (Bayesian) in statistics and is characterized by assumptions that are fully independent. Its assumption is based on the fact that a certain feature in a class is not related to the presence or absence of another feature (Russek, et al., 1983).

A probabilistic model of Naïve Bayes- (Zhang & Feng, 2011) looked into a Naïve Bayes classifier for Text classification. Let $D = \langle d_i \rangle \; where \; i = 1, 2, \dots, n$, represents a dataset to be classified, and d_i is an actual value in the dataset and there exists predefined classes, which is a set of $C = \{c_1, c_2, \dots, c_k\}$. Here, the classification of the data includes assigning a label of the class $c_j, \; where \; j = 1,2, \dots k$ from the set C to a dataset. (Zhang & Feng, 2011) represented Bayes classifier as shown by the equation 2.1:

$$P(D) = \frac{P(c_j)P(D|c_j)}{P(D)} \tag{2.1}$$

where $P(c_j)$ is the preceding information of the appearing probability of the class c_j, $P(D)$ is the information that is obtained from the observations that make up the knowledge from the values in the dataset to be classified and $P(D|c_j)$ is the distribution probability of dataset D in the class spaces.

Bayes classifier works by integrating this information and thereafter computing the *posteriori* separately of the dataset D that falls directly into each of the classes c_j and proceeds by assigning the dataset to the class that has the highest probability, as (Zhang & Feng, 2011) demonstrates on equation 2.2:

$$c * (D) = \frac{\arg arg\, P(D)}{j} \tag{2.2}$$

(Zhang & Feng, 2011) assumes here that the components $d_i$of the dataset D are independent with each other since the conditional probability, represented here by $P(D|c_j)$ cannot be directly computed practically Therefore, equation 2.3 will hold:

$$P(c_j) = \prod_i P(d_i|c_j) \tag{2.3}$$

Equation 2.3 is a representation of a naive Bayes model and this will have an impact on equation 2.1, as (Zhang & Feng, 2011) depicts on equation 2.4:

$$P(D) = \frac{P(c_j)\prod_i P(d_i|c_j)}{P(D)} \tag{2.4}$$

The sample information $P(D)$ is identical to each of the class $c_j,\ where\, j =\ 1,2, \ldots k$, then according to (Zhang & Feng, 2011) equation 2.2 will change to equation 2.5:

$$c*(D) = \left(\frac{arg\,arg\,max}{j}\right) P(c_j) \prod_i P(d_i|c_j) \quad (2.5)$$

Relevance feedback- This is an information retrieval technique used to refine users queries by incrementally refining their queries based on their previous search results. It relies on users giving feedback into the system decision about the relevance of the retrieved documents with respect to their information needs. Content-based recommendation systems adopt the famous Rocchio's formula in relevance feedback (Rocchio, 1971). The general principle here is to get feedback from the users on the documents suggested by the Recommendation System with respect to their query in the form of ratings. The feedback is used to refine incrementally the user profile or to train learning algorithm that deduces the user profile as a classifier.

Some classifiers may have an explicit profile or prototypical document of the category (Sebastiani, 2002). However, Rocchio's method is used for linear, profile-style classifiers. This algorithm models documents as vectors and the similarity between documents is indicated by the similarities between vectors. Terms or words in the document are represented by the components of that vector. The TF-IDF term weighting scheme is employed to compute the weight of each component. The document vectors are combined into a prototype vector for each class in the set of class C for learning. A new document *d* is classified by calculating the similarities between the prototype vector and the corresponding document vector representing *d*, then the class with the highest similarity value for the document vector is assigned *d*.

Rocchio's (Rocchio, 1971) method can be formally presented as $\vec{c_i} = \langle \omega_{1i}, .., \omega_{|T|i} \rangle$ for the category c_i (T is the set of distinct terms in the training set (*vocabulary*)) by equation 2.6:

$$\omega_{ki} = \beta \cdot \sum_{\{d_j \in POS_i\}} \frac{\omega_{kj}}{|POS_i|} - \gamma \cdot \sum_{d_j \in NEG_i} \frac{\omega_{kj}}{|NEG_i|} \quad (2.6)$$

where ω_{kj} is the TF-IDF weight of the term t_k in document $d_{j,}$ $NEG_i$and POS_irepresents the sets of positive and negative examples in the training set for the specific class c_j. γ and β are control parameters to allow the relative importance of all negative and positive examples to be set. Class $\tilde{c}$ is assigned to $d_j$as the c_i with the highest value for similarity after the similarity between each prototype vector $\vec{c_i}$ and the document vector $\vec{d_i}$ is computed. The Rocchio-based classification lacks theoretical support but performance and convergence are guaranteed (Brusilovsky, Kobsa & Nejdl, 2007).

Items Representation

In Content-based Recommendation Systems, as mentioned earlier, items are recommended to users based on their attributes. If we take a movie recommendation system for example, the attributes would be genre, actors/actresses, directors, release year and so on. If items can be represented by the same set of attributes with the known set of values associated to this attributes, then that item is said to have a structured data. When an item has a structured data Machine Learning algorithms can be employed to learn a user profile (Brusilovsky, et al., 2007).

According to (Lops, et al., 2010), items' attributes are textual features extracted from either web pages, emails, product descriptions or news articles. This type of data is unstructured because there are no common attributes with known values. This poses a challenge in trying to learn the users' profile due to the ambiguity of the natural language. The major problem is that the semantics of user interest is not captured by the traditional keyword-based profiles because they rely primarily on string matching operation. A match is assumed if some morphological variant or a string is found in both the profile and the document, which is the reason for why the document will be considered relevant. However, this string matching is much challenged by polysemy (the term or word with different meanings) and synonymy (different terms having the same meaning). This can result in relevant information being missed out if the profile does not use the same keywords in the document or different words could be matched due to polysemy resulting in wrong documents being deemed relevant.

The proposed solution to these challenges is *semantic analysis*. The basic principle here is to adopt the knowledge bases such as ontologies or lexicons to annotate items and represent profiles so as to have a semantic interpretation of the user information needs. This document discusses a basic Keyword-based approach for document representation and an overview of techniques for semantic analysis.

Keyword-Based Vector Space Model

According to (Lops, et al., 2010), the majority of Content-based Recommendation Systems utilize simple retrieval models such as vector space or Keyword matching model with TF-IDF (Term Frequency-Inverse Document Frequency) weighting. Vector space model represents text documents spatially. The model represents each document by a vector in an *n-dimensional* space and each term in a given document's vocabulary corresponds to each dimension of the space. This can be illustrated formally by assuming a term weight vector to represent every document and the degree of association between the term and document to be indicated by each weight. If $D = \{d_1,d_2,\ldots,d_n\}$ is set to denote the documents' set and $T = \{t_1,t_2,\ldots,t_n\}$ the set of words in the document, tokenization, stemming or stop-words removal can be utilized to obtain

T (Ricardo & Berthier, 1999). A vector in a *n-dimensional* vector space represents each document d_j such that $d_j = \{w_{1j}, w_{2j}, \ldots, d_{nj}\}$ and document d_j's weight for term t_k is represented by w_{kj}.

The representation of documents in vector space model faces two challenges: the weighting of the terms and the measuring of the feature vector similarity. The most commonly used TF-IDF is based on the empirical observations regarding text (Salton, 1989):

- The IDF assumption is that rare terms are as relevant as frequent terms
- The TF assumption is that multiple occurrence of a term in a document are as relevant as single occurrences
- Normalization assumption is that the preference between long documents and short documents is the same.

This implies that, terms that occur frequently in one document but rarely in the rest of the corpus are likely to be more relevant to the document's topic. (Salton, 1989) noted that the likelihood of longer documents having a chance to be retrieved most of the time is minimized by normalization of resulting weight vectors. The TF-IDF to underpin the assumption by (Salton, 1989) is shown in equation 2.7.

$$TF - IDF(t_k, d_j) = TF(t_k, d_j) \cdot \log \frac{N}{n_k} \tag{2.7}$$

where N is the number of documents in the corpus, and n_k is the number of documents in the collection in which the term t_k occurs at least once.

$$TF(t_k, d_j) = \frac{f_{k,j}}{\max_z f_{z,j}} \tag{2.8}$$

where the frequencies $f_{k,j}$ of the document d_j's terms t_z are used to compute the maximum, (Salton, 1989) utilized equation 2.8 to obtain the value of *TF*. The cosine normalization is utilized to normalize the weights obtained. (Salton, 1989) utilized Equation 2.9 to ensure that the weight fall in the [0, 1] interval and also to ensure that equal length vectors represent the documents.

$$W_{k,j} = \frac{TF - IDF(t_k, d_j)}{\sqrt{\sum_{s=1}^{|T|} TF - IDF(t_k, d_j)^2}} \tag{2.9}$$

A similarity measure is utilized to compute the closeness of the two documents; it can either be the cosine similarity measure or Pearson correlation. In General, the Content-based Recommendation Systems that rely on vector space model, items and user profile are represented as weighted term vector.

2.3.1.2 Strengths of Content-Based Filtering Technique

According to (Keenan, 2019) content-based filtering technique does have a number of benefits including that of results tend to be highly relevant because content-based recommendations rely on characteristics of objects themselves, they are likely to be highly relevant to a user's interests. This makes them especially valuable for organizations with massive libraries of a single type of content (think subscription and streaming media services). Again, Recommendations are transparent as the process by which any recommendation is generated can be made transparent, which may increase users' trust in their recommendations or allow them to tweak them. With collaborative-filtering, the process is more of a black box–the algorithm and users alike may not really understand why they're seeing the recommendations they are. In addition, Users can get started more quickly because content-based filtering avoids the cold-start problem that often bedevils collaborative-filtering techniques. While the system still needs some initial inputs from users to start making recommendations, the quality of those early recommendations is likely to be much higher than with a system that only becomes robust after millions of data points have been added and correlated. Moreover, new items can be recommended immediately. Related to the cold-start problem, another issue with collaborative-filtering is that new objects added to the library will have few (if any) interactions, which means they won't be recommended very often. Unlike collaborative-filtering systems, content-based recommenders don't require other users to interact with an object before it starts recommending it. Finally, CBF is technically easier to implement. Compared to the sophisticated math involved in building a collaborative-filtering system, the data science behind a content-based system is relatively straightforward. The real work, as we've seen is in assigning the attributes in the first place.

2.3.1.3 Challenges Faced by Content-Based Filtering Technique

On the other hand (Tuan, 2019) states that CBF has some flaws in that it has limited content analysis: if the content does not contain enough information to discriminate the items precisely, the recommendation will be not precisely at the end. Secondly, CBF has Over-specialization problem: content-based method provides a limit degree of novelty, since it has to match up the features of profile and items. A totally perfect content-based filtering may suggest nothing "surprised." Finally, the new user problem: when there's not enough information to build a solid profile for a user, the recommendation cannot be provided correctly.

2.3.2 COLLABORATIVE FILTERING

As one of the most successful approaches to building recommendation systems, CF uses the known preferences of a group of users to make recommendations or predictions of the unknown preferences for other users (Su & Khoshgoftaar, 2009). Such users build a group called neighbourhood. A user gets recommendations to those items that he has not rated before but that were already positively rated by users in his neighbourhood. Recommendations that are produced by CF can be of either prediction or recommendation. Prediction is a numerical value, expressing the predicted score of an item for the user, while Recommendation is a list of top N items that the user will like the most (Isinkaye, et al., 2015); (Breese, et al., 1998).

(Mustafa, et al., 2017) and (Lee, et al., 2012) state that based on the method of implementation, recommendation systems generally can be divided into two categories, memory-based and model-based as shown by figure 2.3. Memory-based method performs recommendation by accessing the database directly, while model-based method uses the transaction data to create a model that can generate recommendation (Bobadilla, et al., 2013). By accessing directly to database, memory-based method is adaptive to data changes, but requires large computational time according to the data size. As for model-based method, it has a constant computing time regardless the size of the data but not adaptive to data changes.

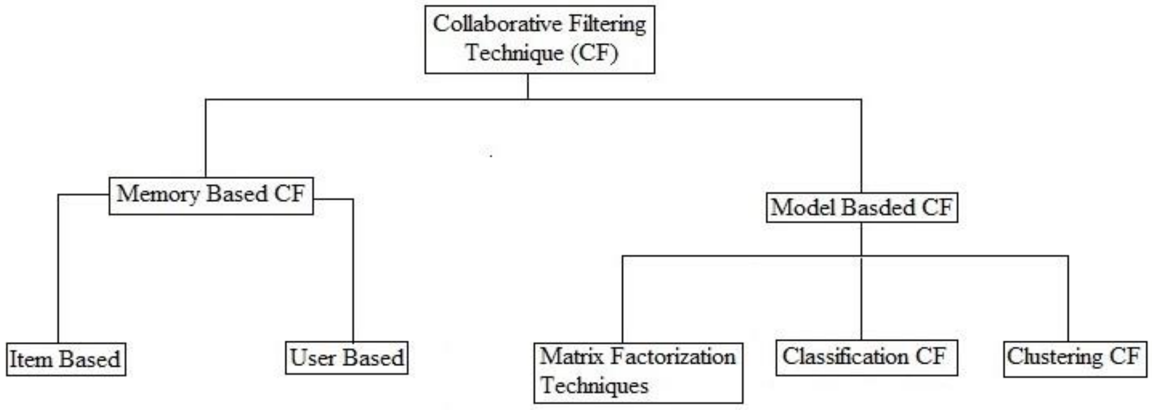


Figure 2.3 Collaborative filtering techniques adapted from (Al-Barznji & Atanassov, 2018)

2.3.2.1 Memory Based Collaborative Filtering

There is no denying that people generally trust recommendations from like-minded friends. Memory-based collaborative filtering applies a neighbor-like pattern to estimate a user's ratings based on the ratings given by like-minded users (Jannach, et al., 2011). The prediction process of memory-based CF typically comprises of three steps: user similarity measurement, neighborhood selection, and estimation generation. This filtering method can be divided into two major categories namely user-item and item-item. User-item approach takes a certain user and searches for other users who have similar rating patterns and suggest items that those similar users have liked. Item-item approach takes an item, search for users who liked that item, and then search for other items that those users or similar users liked. It takes items and output other items as suggestions (Xiaoyuan & Taghi, 2009). For example, in the user based approaches, the value of ratings user u gives to item i is calculated as an aggregation of some similar user's rating of the item. (Adomavicius & Tuzhilin, 2005) used equation 2.10 for this aggregation.

$$r_{u,i} = aggr_{u' \in U} r_{u',i} \tag{2.10}$$

where U denotes the set of top N users that are most similar to user u who rated item i. Some examples of the aggregation functions as utilized by (Adomavicius & Tuzhilin, 2005) and (Breese, et al., 1998) include equations 2.11, 2.12, 2.13 and 2.14.

$$r_{u,i} = \frac{1}{N} \sum_{u' \in U} r_{u',i} \tag{2.11}$$

$$r_{u,i} = k \sum_{u' \in U} simil(u, u')\, r_{u',i} \tag{2.12}$$

where k is a normalizing factor defined as

$$k = 1/ \sum_{u' \in U} |simil(u, u')|, \tag{2.13}$$

and

$$r_{u,i} = \overline{r_u} + k \sum_{u' \in U} simil(u, u')\, (r_{u',i} - \overline{r_{u'}}) \tag{2.14}$$

where $\overline{r_u}$ is the average rating of user u for all the items rated u.

2.3.2.1.1 Item-Based CF

Similarity computation in a recommendation system involves identification of the existing users who have the similar tastes to the current user. This is done by using the existing user's reviews. The preferences of the current user are also taken and are compared to the existing user's preferences. Item-based Collaborative Filtering which is also known as Item-Item CF was first introduced by Sarwar and the crew (Sarwar, et al., 2001). The technique is based on the assumption that if two items are rated similarly by similar people, those items are similar to each other. Instead of calculating similarities between users' rating behaviour to predict preferences, this technique makes use of the similarities between the rating patterns of items (Sarwar, et al., 2001).

The technique was introduced after realizing that although User-based Collaborative Filtering is effective to some extent, it is crippled by the growth of the user base. As an effort to address this issue, the research efforts gave birth to Item-based CF. It was important to extend the Collaborative Filtering technique to cover large user bases so that it can easily be deployed into e-Commerce sites.

Although Item-based Collaborative Filtering may seem inadequate in its raw form, basically because still similarity between items has to be computed (k-NN problem), it complements itself by pre-computing the similarity matrix. User-based CF calculates the neighbourhood when predictions or recommendations are needed. In significantly large User base systems, it is wise to compute similarities based on items because even if one user can decide to add or change the rating, it would not significantly affect or change the similarity between the items especially if the items have many ratings. It is therefore important to pre-compute the similarities between items in an Item-based similarity matrix.

In general Item-based Collaborative Filtering computes the recommendations based on the user's own ratings for other items coalesced with those items' similarity to the target item, instead of other users' ratings and user similarities as in User- based Collaborative Filtering. Nevertheless, the similarities can be calculated by using the same equations mentioned earlier on.

Some of the most popular algorithms used are cosine based similarity, correlation based similarity and adjusted-cosine similarity. The formula for Adjusted-based cosine which is the most popular and believed to be the most accurate (Schafer, et al., 2007) is shown by equation 2.15 as used by (Nagpal, et al., 2015).

$$Itemsim(i,j) = \frac{\sum_{u \in U_{i,j}} (R_{u,i} - \overline{R}_u)(R_{u,j} - \overline{R}_u)}{\sqrt{\sum_{u \in U_{i,j}} (R_{u,i} - \overline{R}_u)^2} \sqrt{\sum_{u \in U_{i,j}} (R_{u,j} - \overline{R}_u)^2}} \qquad (2.15)$$

where $R_{u,i}$ and $R_{u,j}$ represents the rating of user u on items i and j respectively, $\overline{R}_u$ is the mean of the u^{th} user's ratings and $U_{i,j}$ represents all users who have rated items i and j.

The prediction calculation for item based nearest neighbor algorithm for user u and item j as carried out by (Bahadorpour, et al., 2017) is shown by equation 2.16.

$${}^{P}_{item\_based}(u_t, j) = \frac{\sum_{i \in R_{u_t}} Itemsim(i,j) * R_{u_t,j}}{\sum_{i \in R_{u_t}} Itemsim(i,j)} \qquad (2.16)$$

If the predicted rating is high, then the system recommends the item to user. The item-based nearest neighbor algorithms are more accurate in predicting ratings than user based nearest neighbor algorithms (Schafer, et al., 2007).

2.3.2.1.2 User-Based CF

User-based Collaborative Filtering is also known as User-User Collaborative Filtering or k-NN (Ekstrand, et al., 2011) and it is one of the first automated CF methods. It is the one which illustrates the interpretation of the core premise of Collaborative Filtering. This algorithm is based on the assumption that it is highly likely that a user would be interested in what his\her friends showed interest in. So it simply recommends items which were liked by the user's neighbours\friends as illustrated by (Saptono, 2010) on equations 2.17 and 2.18.

$$sim(u,u') = \frac{\sum_{i \in I(u,u^{i})} R(u,i) \cdot R(u',i)}{\sqrt{\sum_{i \in I(u,u')} R(u,i)^2} \sqrt{\sum_{i \in I(u,u')} R(u',i)^2}} \qquad (2.17)$$

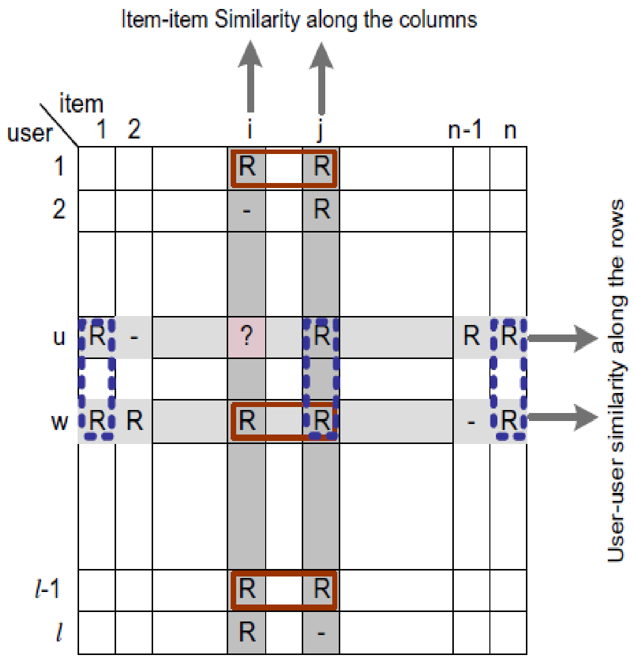
where $I(u,u')$represents the set of all items rated by both user u' and user u. From this similarity calculation, a set of all neighbours of u is formed $N(u)$. The size of this set can vary depending on the overall expected results. Then $R*(u,i)$ is calculated as the adjusted weighted sum of all known ratings $R(u',i)$, and $u' \in N(u)$ (Nakamura & Abe, 1998).

$$R^*(u,i) = \overline{R(u)} + \frac{\sum_{u \in N(u)} sim(u,u') \cdot (R(u',i) - \overline{R(u')})}{\sum_{u \in N(u)} |sim(u,u')|} \tag{2.18}$$

where $\overline{R(u)}$ represents the average rating of user u.

Figure 2.4 shows examples of calculating the similarity of two users, u and w, compared with calculating the similarity of two items, i and j, in the process of predicting the value of item i for user u in the user–item matrix.

Usually, CF systems take two steps: first, the neighbor group, the users who have a similar preference with the target user (for user-based CF) or the set of items that is similar to the item selected by the target user (for an item-based CF), should be determined by using a variety of similarity computing methods. Based on the group of neighbors, the prediction values of particular items, estimating how the target user is likely to prefer the items, are obtained and then the Top-N items with a higher predicted value that will be of interest to the target user are identified.



Item-item Similarity along the columns
item
user
1
2
i
j
n-1
n
1
2
u
w
l-1
l
R
-
?
User-user similarity along the rows

Figure 2.4 Similarity computation in a user-based CF and item-based CF (Kim, et al., 2010).

2.3.2.2 *Model Based Collaborative Filtering*

The main drawback of memory-based technique is the requirement of loading a large amount of in-line memory. The problem is serious when rating matrix becomes so huge in situation that there are extremely many persons using system (Al-Bashiri, et al., 2018). Computational resource is consumed much and system performance goes down; so system can't respond user request immediately. Model-based approach intends to solve such problems. In general, latent factor models offer high expressive ability to describe various aspects of the data. Thus, they tend to provide more accurate results than neighborhood models. There are four common approaches for model-based CF such as clustering, classification, latent model, Markov decision process (MDP), and matrix factorization (Do, et al., 2010); (Koren, et al., 2009).

2.3.2.2.1 Matrix Factorization CF

Matrix factorization technique has been adopted from the numerical linear algebra. It is now used widely in Recommendation Systems due to its capability to improve recommendation accuracy (Adomavicius & YoungOk, 2012). It has been proved to be a better option to address the issues of data sparsity, over-fitting and convergence speed. Most of the best performing algorithms incorporates this technique. It was evident in the earlier mentioned Netflix Prize competition (Koren, et al., 2009) where most of the algorithms presented incorporated technique.

The basic version of the technique relies on the assumption that the user's preference or rating of an item is composed of sum of preferences about various features of that item. The model is stimulated by Singular Value Decomposition (SVD). If users' ratings on items can be represented in the form of matrix *M*, the SVD of that matrix is the factorization into three component matrices in such a way that $M = U\sum T^T$, Σ represents a diagonal matrix with singular values $\sigma_i$of the decomposition. *U* and *T* are orthogonal (their determinants are either 1 or -1. *U* should not be confused with the set of users in this context). This introduces the intermediate vector space represented by Σ. The vectors are transformed from item-space into the intermediate vector space by $\sum T^T$. Classically *U* is m×k, Σ is $k\times k$ and *M* is $m\times n$. *M* has *rank k* where k is a smaller number representing the reduced dimensional of the rating space. The closest approximation to *M* is achieved by truncating Σ to Σ_k by retaining only the *k* largest singular values. The best possible *rank k* approximation can be achieved by using the Frobenius norm to measure the error (Deerwester, et al., 1990).

There are two more things achieved by the truncation; the decrease of the vector space dimensionality which in turn minimizes the storage and computational requirement of the model, and also the reduction of singular values eliminates the redundant noise and leaves only the strongest effects or trends in the model (Ekstrand, et al., 2011). This helps to provide high quality recommendations. The computation of the SVD of the ratings matrix yields the following factorization:$R \approx U\sum T^T$if $m = |U|, n = |I|, and$ Σ is a $k \times k\ matrix.$

The feature preference-relevant model is associated with the computation of *rank-k* SVD of $R \approx U \sum T^T$ where by the rows of matrix U are perceived to represent the user's interest in each of the k features and the rows of I are the items relevance for each feature. The singular values in Σ are taken to be preference weights which represent the influence of a particular feature on user-item preferences across the system. A user's preference for a particular item is therefore computed as the weighted sum of the user's interest in each of the features of the item multiplied by the item's relevance to the features.

It is therefore necessary to compute the matrix factorization first in order to use SVD. Singular Value Decomposition can be computed in numerous ways, there are a lot of algorithms like Lanczo's algorithm, the generalized Hebbian algorithm and expectation maximization (Gorrell, 2006); (Kurucz, et al., 2007); (Sanger, 1989).

There must be a dummy data to fill in the missing values of the rating matrix in order to well define the Singular Value Decomposition. This dummy data has to be reasonable, therefore it is computed by taking the item's average rating (Sarwar, et al., 2001). Nevertheless there are several methods proposed which can estimate the SVD irrespective of the missing ratings (Kurucz, et al., 2007). The most popular method is the gradient descent method. This method trains each feature f in turn using update rules as illustrated by (Miller, et al., 2004) in equations 2.19 and 2.20.

$$\Delta u_{j,f} = \lambda(R(u,i) - R*(u,i))i_{k,f} \tag{2.19}$$

$$\Delta u_{k,f} = \lambda(R(u,i) - R*(u,i))i_{k,f} \tag{2.20}$$

where λ is the learning rate which is typically 0.001. This method also prevents over fitting by allowing regularization (subtracting constant factor to minimise the variance of predicted regression parameters) (Funk, 2006). Although the resulting model does not reflect a true SVD due to the fact that the constituent matrices are no longer orthogonal, it tends to be more accurate in predicting latent preferences than the SVD which is not regularized (Koren, et al., 2009). (Miller, et al., 2004) added additional term to equations 2.19 and 2.20 in an attempt to regularize. This is shown in equations 2.21 and 2.22.

$$\Delta u_{j,f} = \lambda\big((R(u,i) - R*(u,i))i_{k,f} - \lambda u_{k,f}\big) \tag{2.21}$$

$$\Delta u_{k,f} = \lambda\big((R(u,i) - R*(u,i))i_{j,f} - \lambda u_{k,f}\big) \tag{2.22}$$

where λ is the regularization factor which is typically 0.1- 0.2. The ratings can also be normalized by subtracting the user's average rating or any other baseline predictor before computing the SVD. This may improve the accuracy and induce convergence of iterative methods (Funk , 2006); (Sarwar, et al., 2000).

After the computation of SVD, it is essential to update it to show new users, items and ratings. This is achieved by employing a commonly known method called *folding-in.* This method practically works well and shows the recommendation of users who were not even considered during the factorization of the rating matrix (Berry, et al., 1995); (Sarwar, et al., 2002). It computes a feature relevance vector for the new user or item but it does not re-compute the decomposition itself.

Folding-in computes the feature interest vector u for user u in such a way that $\pi_u \approx u\Sigma T^T$. If r_u is taken to be the rating vector, u is calculated as $u = (\Sigma T^T)^{-1} r_u = T\Sigma^{-1} r_u$. the *folding-in* method ignores the ratings which are zeros. The process is symmetrical; by substituting *U* for *T* we get item vectors.

It is necessary to re-compute the complete factorization periodically due to the fact that the accuracy of the SVD deteriorates with time as *folding-in* process updates the user and item vectors. The process can be performed off line in deployed systems when there is less load (Sarwar, et al., 2002).

Another method for building and maintaining the SVD based on *rank-1*updates was proposed by (Brand, 2003). This method produced faster real-time updates of the SVD. It bootstraps the SVD with the dense portion of the dataset. Users and items are sorted to make a dense corner in the matrix and this dense portion is extracted from that corner.

The weighted dot product of user-feature preference vector ***u*** and the item-feature relevance vector ***i*** are perceived to represent the user *u*'s preference for item *i* as (Miller, et al., 2004) demonstrates by equation 2.23.

$$R*(u,i) = \sum_f u_f \sigma_f i_f \tag{2.23}$$

Then items can be ranked according to their predicted preference and be recommended to the users.

2.3.2.2.2 Classification CF

Classification complications target to find common characteristics that specify the group to which each instance fits. This can be utilized both to recognize the existing data and to predict how new cases will perform. Data mining produces classification models by examining the data that is already classified and inductively discovering a predictive pattern. The existing cases may be derived from ancient databases. They may also be a result of an experiment in which a model of the entire database is tested in the real world and the fallouts used to design a classifier. Sometimes an expert is required to classify a sample of the database, and that sample is used to create the model which will be applied to the entire database (Rajput, et al., 2011). Different classification algorithms are applied and used with different datasets, some of these algorithms are discussed below:

Multi-Layer Perceptron

A Multi-layer Perceptron (MLP) is a class of feedforward artificial neural network. MLP consists of at least three layers of nodes. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called back-propagation for training. It can distinguish data that is not linearly *separable. Multilayer perceptrons are sometimes colloquially referred to as "vanilla" neural networks, especially when they have a single hidden layer* (Hastie, et al., 2009).

Artificial Neural Network (ANN) is a unified group of nodes by means of mathematical approaches to process information. It is a self-adaptive system, which can change its construction based on the internal or external influences. Multiple ANN models have been developed and the most prevalent one is the Multi-Layer Perceptron (MLP) feed forward network (Kavzoglu & Mather, 2003). MLP consists of many layers. The most widely used structure was the three-layer structure, due to its capability to solve most image classification problems. The multiple layers include one input layer, one hidden layer, and one output layer as on shown in Figure 2.5.

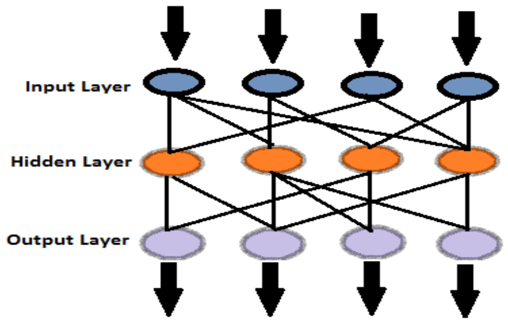


Figure 2.5 Illustration of MLP nodes look like adapted from (Park, et al., 1991).

Each layer is composed of artificial neurons. It is visible in 2.5 that all the nodes are linked with each other, excluding the nodes in the same layer. The input layer, the hidden layer and the output layer are used for data input, processing, and output, respectively. The downside of this algorithm is that creating a neural network is time consuming.

Logistic Regression

Logistic regression, by use of a linear combination of independent variables, is a statistical technique used to predict the possibility of occurrence of an event, i.e. its probability (Jung , et al., 2014). However, it is evident that the algorithm yields low accuracy with high-processing speed, indicating that classification using only a logistic algorithm cannot guarantee the accuracy of the results (Jung , et al., 2014). For this purpose, logistic algorithms need to be used in conjunction with other algorithms to validate the results.

JRIP

JRip also known as (RIPPER) is one of the straightforward and most popular algorithms. It examines classes in increasing size and it generates an initial rule set using incremental reduced error. JRip proceeds by treating all the samples of a specific ruling in the training data as a class, thus discovering a set of rules that conceal all the members of the class. This process is iterated until all classes have been covered (Rajput, et al., 2011).

J48

J48 is a tree classifier. A tree is moreover a leaf node labelled with a class, or a structure containing a test, then linked to two or more nodes also known as subtrees (Shepperd & Kadoda, 2001). To classify some instance, first there is a need to identify its attribute-vector and apply this vector to the tree. The tests are done on the attributes, reaching one or other leaf, to complete the classification process, as illustrated in Figure 2.6.

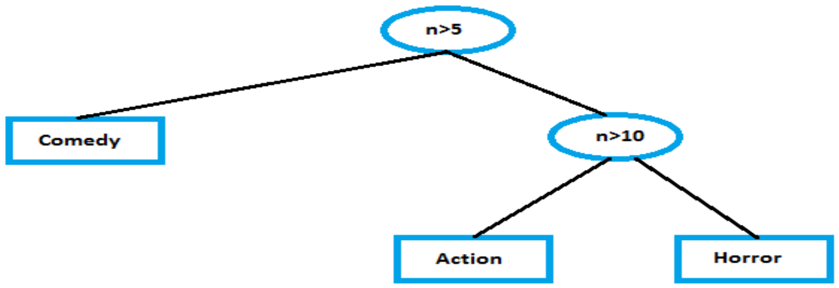


Figure 2.6 Simple tree classifier classification process adapted from (Jadhav & Channe, 2016).

From the example on figure 2.6, let n=5

Then n>5 = true

n>10 =true

therefore, the above will be classified as a comedy.

2.3.2.2.3 Clustering CF

Clustering CF (Ungar & Foster, 1998) is based on assumption that users in the same group have the same interest; so they rate items similarly. Therefore, users are partitioned into groups called clusters which is defined as a set of similar users.

Suppose each user is represented as rating vector denoted $u_i = (r_{i1}, r_{i2}, \dots, r_{in})$. The dissimilarity measure between two users is the distance between them. Minkowski distance (Shrkhorshidi, et al., 2015), Euclidian distance (Jeyasekar, et al., 2016) or Manhattan distance (Zheng, et al., 2016) as shown in equation 2.24, 2.25 and 2.26 respectively may be used.

$$distance_{Minkowski}(u_1, u_2) = \sqrt[q]{\sum_j (r_{1j} - r_{2j})^q} \tag{2.24}$$

$$distance_{Euclidian}(u_1, u_2) = \sqrt{\sum_j (r_{1j} - r_{2j})^2} \tag{2.25}$$

$$distance_{Manhattan}(u_1, u_2) = \sum_j |r_{1j} - r_{2j}| \tag{2.26}$$

The less distance u_1, u_2 is, the more similar u_1 and u_2 are. Clustering CF includes two steps:

1. Partitioning users into clusters and each cluster always contains rating values. For example, every cluster resulted from k-mean algorithm has a mean which is a rating vector like user vector.

2. The concerned user who needs to be recommended is assigned to concrete cluster and her/his ratings are the same to ratings of such cluster. Of course how to assign a user to right cluster is based on the distance between user and cluster.

So the most important step is how to partition users into clusters. There are many clustering techniques such as k-mean and k-centroid. The most popular clustering algorithm is k-mean algorithm (Torres, 2004) which includes three following steps:

1. It randomly selects k users, each of which initially represents a cluster mean. Of course, we have k cluster means. Each mean is considered as the "representative" of one cluster. There are k clusters.

2. For each user, the distance between it and k cluster means are computed. Such user belongs to the cluster to which it is nearest. In other words, if user u_i belong to cluster c_v, the distance between u_i and mean m_v of cluster c_v, denoted$distance(u_i, m_v)$, is minimal over all clusters.

3. After that, the means of all clusters are re-computed. If stopping condition is met then algorithm is terminated, otherwise returning step 2.

This process is repeated until the stopping condition is met. There are two typical terminating conditions (stopping conditions) for k-mean algorithm:

- The k means are not changed. In other words, k clusters are not changed. This condition indicates a perfect clustering task.

- Alternatively, error criterion is less than a pre-defined threshold.

If the stopping condition is that the error criterion is less than a pre-defined threshold, the error criterion is defined as shown in equation 2.27 (Torres, 2004):

$$error = \sum_{v=1}^{k} \sum_{u_i \in c_v} distance(u_i, m_v) \quad (2.27)$$

where c_v and m_v is cluster v and its mean, respectively. However, clustering CF encounters the problem of sparse rating matrix in which there are many missing values, which cause clustering algorithms to be imprecise.

In order to solve this problem, (Ungar & Foster, 1998) proposed an innovative clustering CF which firstly groups items based on which users rate such items and then uses the item groups to help group users. Their method is a formal statistical model.

2.3.2.3 Strengths of Collaborative Filtering Technique

(Keenan, 2019) states that collaborative filtering technique is beneficial in that It produces more serendipitous recommendations. When it comes to recommendations, accuracy isn't always the highest priority. Content-based filtering approaches tend to show users items that are very similar to items they've already liked, which can lead to filter bubble problems. By contrast, most users have interests that span different subsets, which in theory can result in more diverse (and interesting) recommendations. In addition, it is flexible across different domains. Collaborative filtering approaches are well suited to highly diverse sets of items. Where content-based filters rely on metadata, collaborative filtering is based on real-life activity, allowing it to make connections between seemingly disparate items (like say, an outboard motor and a fishing rod) that nonetheless might be relevant to some set of users (in this case, people who like to fish). Moreover, it can capture more nuance around items. Even a highly detailed content-based filtering system will only capture some of the features of a given item. By relying on actual human experience, collaborative filtering can sometimes recommend items that have a greater affinity with one another than a strict comparison of their attributes would suggest. Finally, it benefits from large user bases. Simply put, the more people are using the service, the better your recommendations will become, without doing additional development work or relying on subject area expertise.

2.3.2.4 Challenges Faced by Collaborative Filtering Technique

While collaborative filtering is commercially the most successful approach to recommendation generation, it suffers from a number of well-known problems (Anand & Mobasher, 2003). These problems are highlighted below:

2.3.2.4.1 Data Sparsity

Usage of recommendation system increases very rapidly. Many commercial recommendation systems make use of large datasets. Therefore, the user-item matrix used for filtering could be very large and sparse and because of that performance of recommendation process may get degrade. The cold start problem is caused by the data sparsity. In collaborative filtering method recommendation of item is based on past preferences of users, so that new users will need to rate enough count of items to allow the system to catch their preferences accurately and thus allows for authentic recommendations (Lü, et al., 2012); (Madhukar, 2014).

2.3.2.4.2 Coldstart

One of the most known problems in RSs is the cold start problem. The cold start problem is related to the sparsity of information (i.e., for users and items) available in the recommendation algorithm. The problem happens in recommendation systems due to the lack of information, on users or items: there is relatively little information about each user, which results in an inability to draw inferences to recommend items to users. The provision of a high QoR in cold start situations is a key challenge in RS (Park & Chu, 2009). Three types of cold start problems could be identified: (a) recommendations for new users, (b) recommendations for new items, and (c) recommendations on new items for new users (Madhukar, 2014).

2.3.2.4.3 Scalability

Traditional CF algorithms will suffer from scalability problems as the numbers of users and items increases. For example, consider a ten millions of customers O(M) and millions of items O(N), with that the complexity of algorithm is „n" which is already too large. As recommendation systems play an important role in E-commerce applications where systems must respond to the user requirement immediately and irrespective of user's ratings history and purchases system must make recommendations, which requires a higher scalability. Twitter is large web company to scale the recommendations of their millions of users it uses clusters of machines (Lü, et al., 2012); (Madhukar, 2014).

2.3.2.4.4 Diversity

Recommendation systems are anticipated to increase diversity because they help us to discover new products. Some algorithms, may accidentally do the opposite. Here recommendation system recommends popular and highly rated items which are appreciated by particular user. This lead to lower accuracy in recommendation process. To overcome this problem there is need to develop new hybrid approaches which will enhance the efficiency of recommendation process (Lü, et al., 2012).

2.3.2.4.5 Vulnerability to Attacks

Security is one of major issues in any system which is deployed on web. Recommendation systems play an important role in e-commerce applications and because of that recommendation systems are probably targets of harmful attacks trying to promote or inhibit some items. This is one of major challenge faced by the developers of recommendation system (Lü, et al., 2012).

2.3.2.4.6 Synonymy

Synonymy refers to the tendency of a number of the same or very similar items to have different names or entries. Most recommendation systems are unable to discover this latent association and thus treat these products differently. For example, the seemingly different items “children movie” and “children film” are actual the same item, but memory-based CF systems would find no match between them to compute similarity. Indeed, the degree of variability in descriptive term usage is greater than commonly suspected (Shinde & Shedge, 2013).

2.3.2.4.7 Gray Sheep

Gray sheep refers to the users whose opinions do not consistently agree or disagree with any group of people and thus do not benefit from collaborative filtering (Shinde & Shedge, 2013).

2.3.2.4.8 Shilling Attacks

In cases where anyone can provide recommendations, people may give tons of positive recommendations for their own materials and negative recommendations for their competitors. It is desirable for CF systems to introduce precautions that discourage this kind of phenomenon (Shinde & Shedge, 2013).

2.3.3 HYBRID FILTERING

The collaborative and content-based techniques have several limitations and drawbacks. To overcome these limitations, hybrid recommendation systems are introduced. The hybrid systems combine the aforementioned techniques to enhance the advantages which are achieved. In the situations that there is no information about users or their ratings, the content-based part of the hybrid recommendation system can be helpful to retrieve useful information to generate recommendation. On the other hand, when information about the contents associated to the items is not sufficient, the collaborative part of the hybrid recommendation system can be supportive. Consequently, the cold start and data sparseness problems of recommendation systems will be resolved (Kardan & Ebrahimi, 2013). A number of the hybrid recommendation systems operate based on switching hybrid approach, which apply the content-based or collaborative recommendation technique depending on several criteria and available data. For example, this approach have been used in the hybrid recommendation system proposed by (Porcel, et al., 2012) and (Serrano-Guerrero, et al., 2011). In these systems, when a new item is added to the system, the content-based recommendation technique is used to find the similar and related items. Similarly, when a new user enters to the system, collaborative recommendation approach is applied to find similar users and find their interested items to be recommended. (Castro-Herrera, et al., 2009) presents a hybrid recommendation system which clusters similar contents based on their keywords by TFIDF technique. The main goal of this system is to find similar and relevant users in forums based on their contributed posts. The collaborative filtering part of this system computes the similarity of users with the same interests who have contributed in common posts. The similarity criterion of contents in this research is based on the number of shared keywords, and also the weight and frequency of their happenings rather than their semantic similarities. This will produce two contents with the same keywords but with different concepts being placed in the same cluster.

There are many researches regarding hybrid recommendation systems for other domains. One approach to create the hybrid systems is to use the results obtained from the collaborative filtering part and feeding them as an input to the content-based filtering part of a hybrid system. Salter and Antonopoulos employ this approach to recommend movies to users (Salter & Antonopoulos, 2006). This system first finds user's neighbours based on their ratings of movies. Then according to the film metadata, such as actors, director, similarity between the users' data, and the rating score of a new and unrated movie is calculated. This system suffers from rating sparseness problem because it uses co-rated items to identify the neighbours in the collaborative filtering part of its suggested system. In the research carried out by (Semeraro, et al., 2005). The obtained results from the content-based part is utilized to improve the neighbourhood foundation in the collaborative filtering part of the system. This approach uses Word Sense Disambiguation technique to cluster users based on interested and uninterested items. Dependency of this approach to the users' ratings in the process of finding users' interested items or uninterested ones and eventually predicting the score of the active user on a specific item will result in decreasing the accuracy of the recommendation system.

(Burke & Felfernig, 2011) indicated that there are seven different ways in which hybrid combinations can be archived to resolve among others, the problem of cold start. Although these combinations mainly focus on combining information across different sources, it does not rule out the fact that there still exist situations where several different techniques of the same type have been combined or hybridized. The example is NewsDude which uses both Naïve Bayes and KNN classifier to recommend news. The seven combination types are:

- Weighted: It numerically combines the scores of different recommendation components.
- Switching: The system applies the selected recommendation component which is chosen among others.
- Mixed: combination of different recommendations from different recommenders
- Feature Combination: The single recommendation algorithm is fed with combined features derived from different knowledge base.

- Feature Augmentation: One recommendation technique is utilized to process a set or one feature which in turn is an input to another technique.
- Cascade: Recommenders are prioritised and the lower priority break ties in the scoring of the higher ones.
- Meta-level: A model build from pressing with one technique is fed as an input to another technique.

2.3.3.1 Weighted

In order to illustrate the weighted technique, a movie Recommendation System in (Mobasher, Jin & Zhou, 2004) will be used. This Recommendation System has two components, the other one uses Collaborative Filtering technique while the other one uses simple semantic knowledge about the attributes of a movie. The collaborative technique identifies similarities between rating profiles and predicts the ratings based on that information. The second component used latent semantic analysis and recommends movies which are semantically similar to the ones the user liked. The linear weighting scheme is used to combine the two outputs from these components. This design is perceived to be the simplest one for hybrid systems.

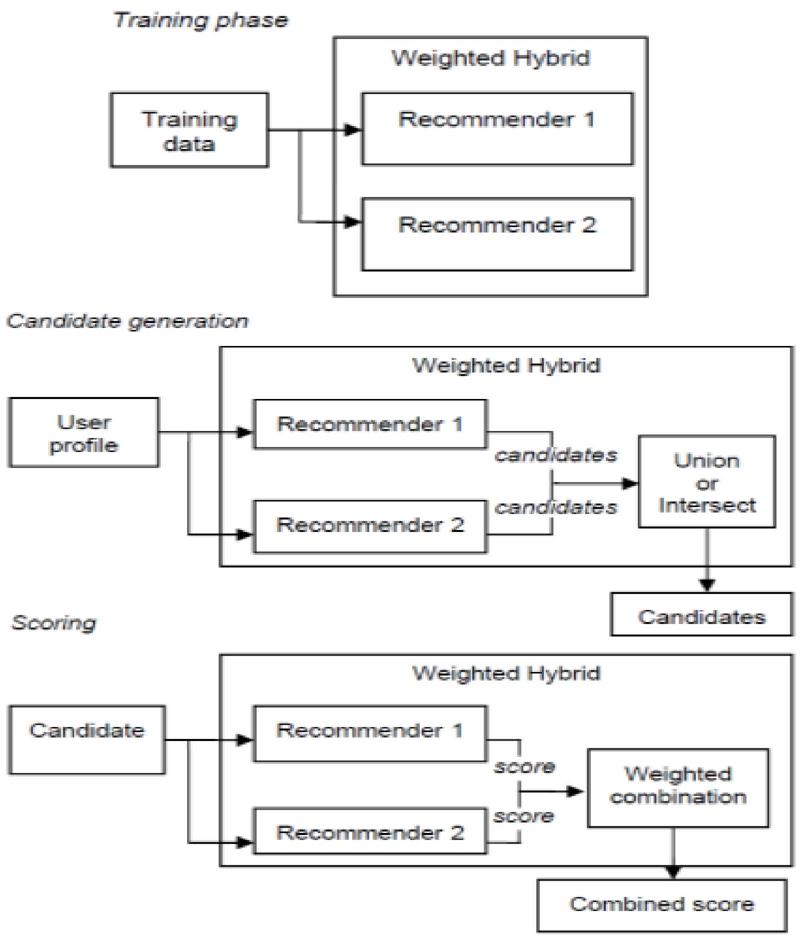


Figure 2.7 Weighted hybrid (Burke & Felfernig, 2011)

This weighted algorithm operates in the manner illustrated in Figure 2.7. During training phase, each recommender processes the training data. This phase is similar for almost all the hybrid scenarios. Then the recommenders jointly come up with candidate items for a test user. Content-based algorithms are able to make predictions on any item as they are not affected by the cold start problem, Collaborative Filtering techniques have to find the peer users or neighbours before they can predict the ratings for items. These candidates are therefore necessary to identify those items which can be considered.

This set of candidates is then rated jointly. The manner in which candidate sets are handled differs from hybrid to hybrid. The union or the intersection of these sets may be considered. If the intersection is considered, it is eminent that the only small set of candidate will be utilized between the candidate sets. On the other hand, if the union is considered, there is a possibility that a recommender may not be able to rate some of the given candidates. In this situation. It may be decided as to how the system should handle such. One possibility could be to rate such items neutral, which means they are neither liked nor disliked. Each candidate is then rated by the two recommendation components and a linear combination of their scores is computed and taken to be the items' rating. Then the items are ranked by their scores and the top items are recommended to the user.

The best weights for each component are determined empirically, for an example, it was found that 60/40 weighing semantic/collaborative produces better accuracy in systems (Mobasher, et al., 2005). The implicit assumption is that each recommendation component will uniformly perform across the product and user space. Although each component constitutes a fixed contribution to the score, it is still possible to have recommenders having different strengths in different parts of the product space. This leaves space for other hybrid combinations whereby the hybrid switches between its components depending on the context.

2.3.3.2 Mixed Hybrid

To explain the Mixed Hybrid, the document uses a Personalised TV (PTV) listing recommender for television shows (Smyth & Cotter, 2000). It also has two components consisting of Collaborative and Content-based recommender. However, because of the data sparsity of the ratings and the content space, recommenders struggle to produce a rating for any given show. Therefore, the components produce recommendations separately and they are combined just before presented to the user. These recommendations from different components are presented side by side in a combined list. The challenge posed by this type of recommendation is that, if lists are to be combined, how can rankings be integrated? The technique that could be used here is to merge based on predicted rating or on recommender confidence. This is illustrated on Figure 2.8.

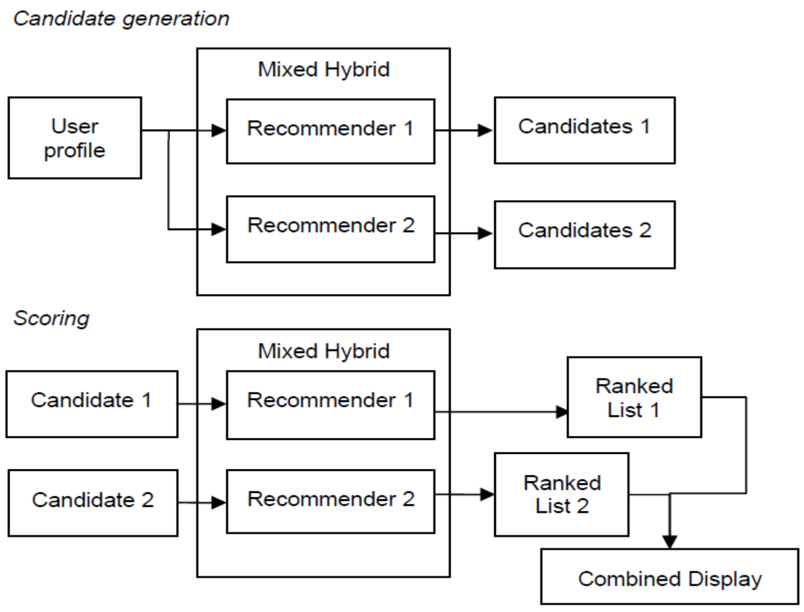


Figure 2.8 Mixed hybrid (Burke & Felfernig, 2011)

The evaluation of a mixed recommender poses some challenges when using retrospective data. Other types of hybrids types use users' actual rating to validate the rankings. With mixed strategy whereby results are presented side by side, it is difficult to monitor the improvement over the components without conducting an online user study.

2.3.3.3 Switching

To illustrate the switching combination, NewsDude (Pazzani & Billsus, 1998) Recommendation System for news stories will be used. It has a combination of three recommendation components; a Content-based nearest-neighbour recommender, a Collaborative Filtering recommender and another Content-based algorithm which uses a Naïve Bayes classifier. The recommenders are applied in order with the nearest neighbour technique being the first to be applied and if the results are still not that satisfactory, the Collaborative Filtering recommender is applied then Naïve Bayes is applied at the end.

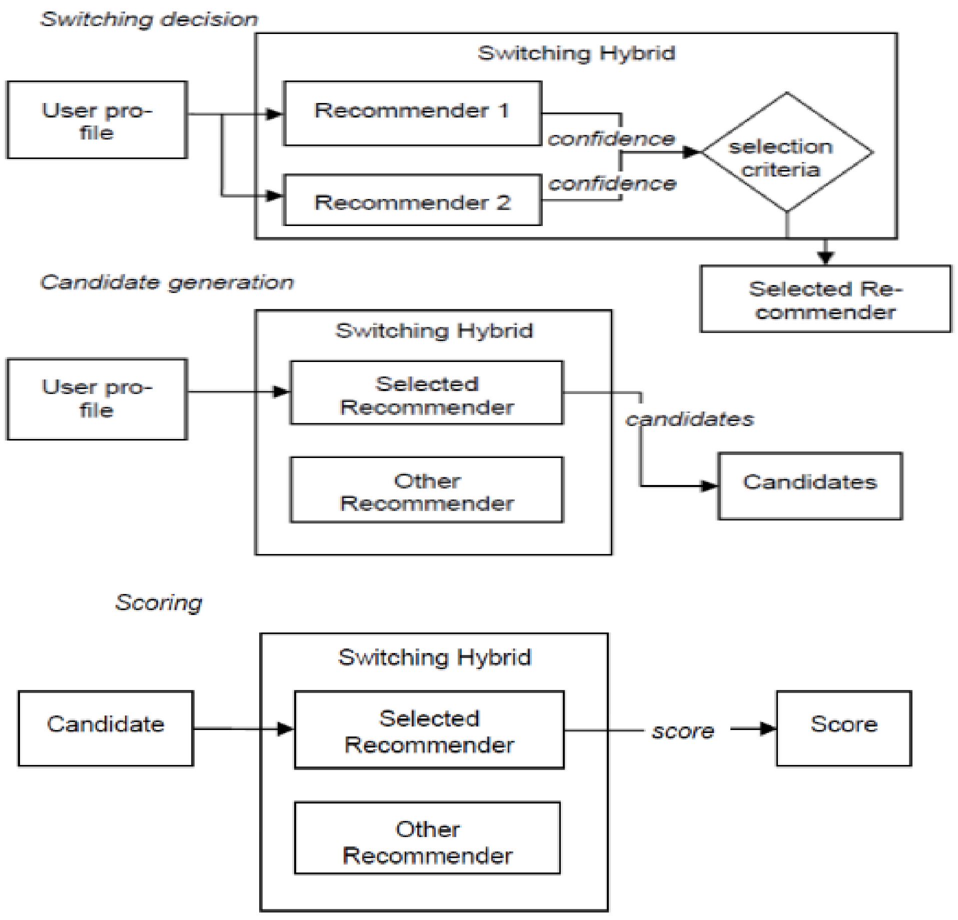


Figure 2.9 Switching hybrid (Burke & Felfernig, 2011)

In general, a switching hybrid selects a suitable recommendation technique among its constituent's components based on the recommendation situation. This combination takes into account the fact that components may be inconsistent with different types of people. Therefore, a different recommender is chosen for different profiles. There has to be a reliable criterion in which the switching decision will be based on. Confidence values encompassed in the recommendation components themselves have been largely employed by some other researchers (Setten, 2005). The other option is to use external criteria (Mobasher, et al., 2001). The issue of determining the appropriate confidence value for a recommendation is still an active research area and the developments can be observed from (Cheetham & Price, 2004). Figure 2.9 illustrates that initially the switching hybrid begins the recommendation process by selecting the appropriate component for the situation.

2.3.3.4 Feature Combination

Feature combination injects features of one source such as Content-based recommendation into an algorithm which was designed for other processing algorithm such as Collaborative Filtering. The idea is illustrated in Figure 2.10.

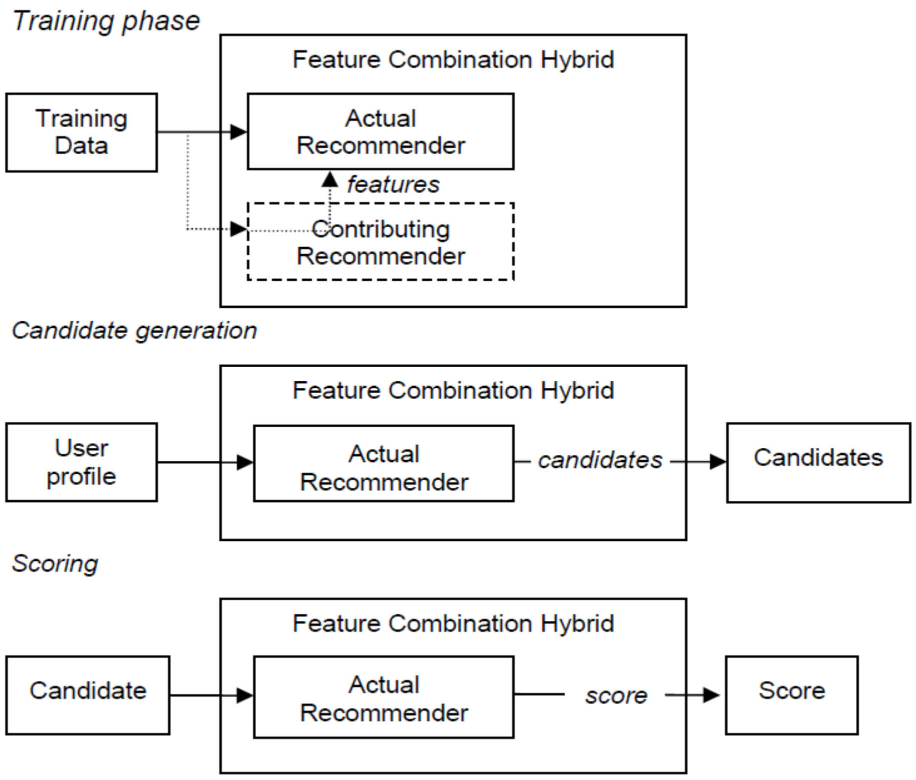


Figure 2.10 Feature combination hybrid (Burke & Felfernig, 2011)

From Figure 2.10, it can be noticed that there is also a virtual contributing recommender to compliment the component that actually makes the recommendation. The features which would have otherwise been processed by this recommender, are fed serially as an input to the actual recommender. The addition of new kinds of features ensures the expansion of the capabilities of a well-tuned and well-understood system (Basu, et al., 1998).

The feature content is a little different from the other types of hybrids in the sense that it is not composed of different components, there is only one component doing the recommendation task. Instead of making use of different components, it rather includes some of the recommendation logic from other techniques.

2.3.3.5 Feature Augmentation

Feature augmentation and feature combination are similar in a way, in that they all deal with attributes. The difference is that in feature augmentation new feature for each item is generated by using the recommendation logic of the contributing domain instead of using features depicted from the contributing recommender's domain. Association rule mining can be utilized in collaborative data to come up with new content features for Content-based recommendation (O'Sullivan, et al., 2004).

The difference is eminent in Figure 2.11 whereby the contributing recommender taps in on the data headed for the actual recommender and appends it with its own at each step. This is in contrast with feature combination where raw features are utilized. Feature augmentation is more applicable where there is a strong primary recommendation component and a desire to have additional knowledge sources. Practically the augmentation is done offline to make the approach attractive like in feature combination whereas on the other hand the existing recommendation algorithm is strengthened by adjusting its input.

Feature augmentation usually has the upper hand against feature combination because of the fact that it is challenging to create a feature combination for hybrid for all possible hybrids, therefore feature augmentation is deemed flexible. Another reason is that the primary recommender in feature combination hybrid is often confronted with the added dimensionality of the training data especially the collaborative rating data.

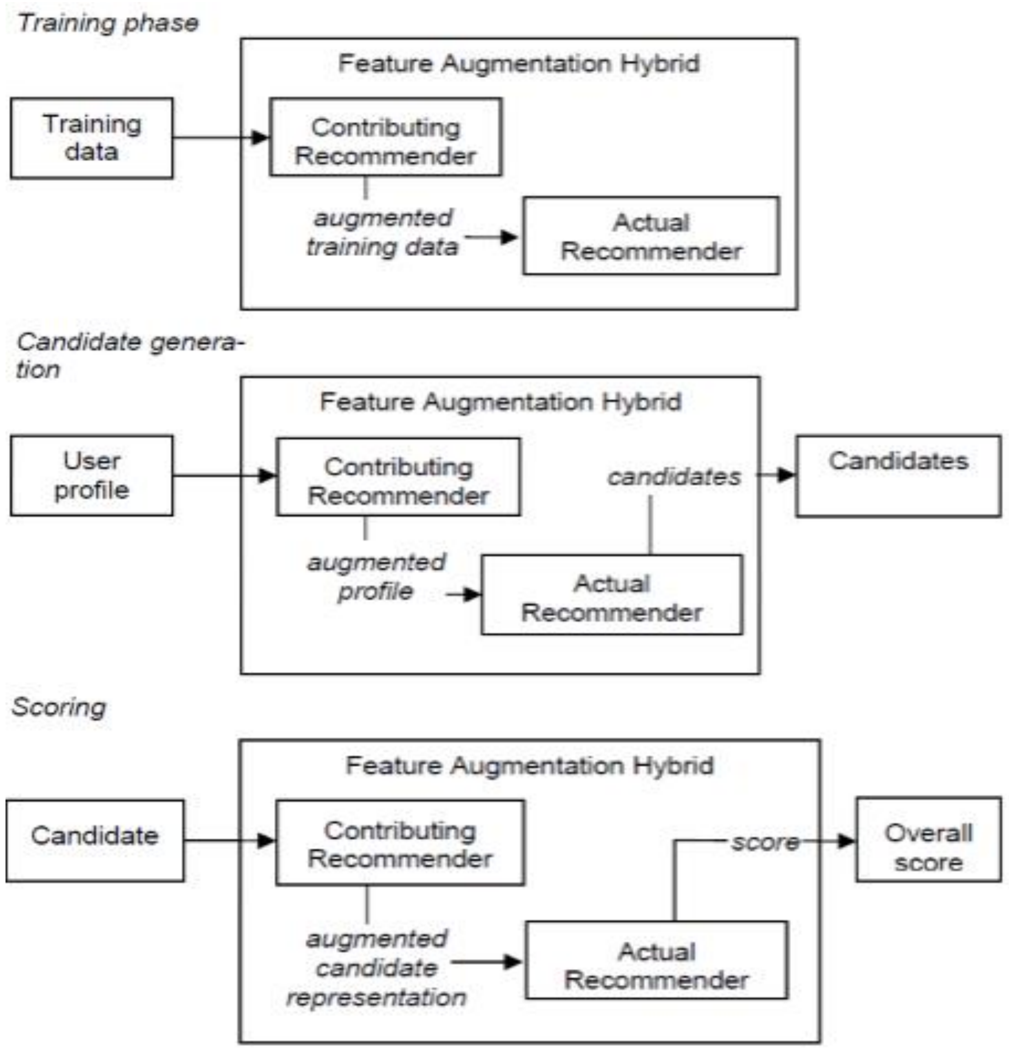


Figure 2.11 Feature Augmentation hybrid (Burke & Felfernig, 2011)

2.3.3.6 Cascade

The components in Cascade hybrid are in hierarchical fashion in such way that a decision made by a stronger recommender cannot be overturned by the weaker recommender, instead it can only refine it. Although in its order of dependence it might look similar to feature augmentation hybrid, its approach is to retain the core function of the recommendation as to predict ratings. The secondary component is used only to break the ties in the scoring of the primary one. Figure 2.12 illustrates a schematic depiction of this hybrid.

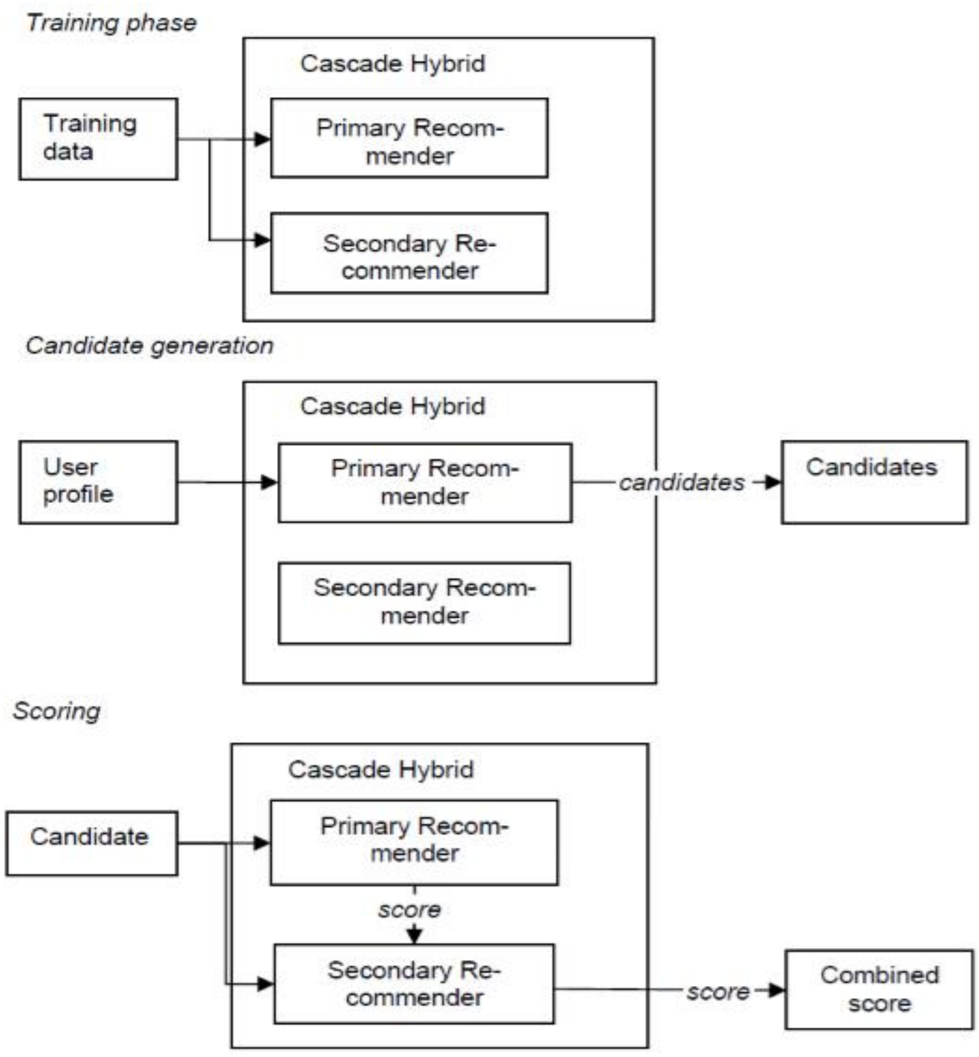


Figure 2.12 Cascade recommendation (Burke & Felfernig, 2011)

The probability of the ties in many recommendation techniques is minimal as they use real valued outputs. This results in secondary components in cascade to have minimal job to do. According to (Burke, 2007) the literature did not reveal any traces of the cascade type when the original hybrid survey was conducted. However EntreeC had the knowledge-based component which was already producing the integer valued score and ties were observed in set which made the cascade design to be a natural one (Burke, 2002).

Real-valued output of the recommendation algorithms is susceptible to inconsistency or uncertainty therefore this is correlated to the cascade hybrid. It is impractical to expect the precision up to the 32 of the floating point value. However, if the scoring of the algorithms is significantly imprecise, the cascade design can be employed to fine tune and refine the ties that might exists.

2.3.3.7 Meta-level

A Meta-level hybrid uses one model learned by one recommender as an input to another. An example is a Fab (Balabanovic & Shoham, 1997) document recommender which uses the same "collaborative through content" structure. It is partly similar to feature augmentation in the sense that the contributing recommender feeds the input to the actual recommender. The difference is that the contributing recommender completely replaces the original knowledge source with a learned model that the actual recommender utilizes in its processing. The actual recommender does not deal with raw profile data. This can be considered a change of basis in the recommendation space.

It is not necessarily feasible to implement a meta-level hybrid from any given pair of recommenders. There has to be some kind of a model produced by the contributing recommender to be utilized as the input to the actual recommender and not all recommendation logics are capable of that. Figure 2-13 schematically shows the hybrid.

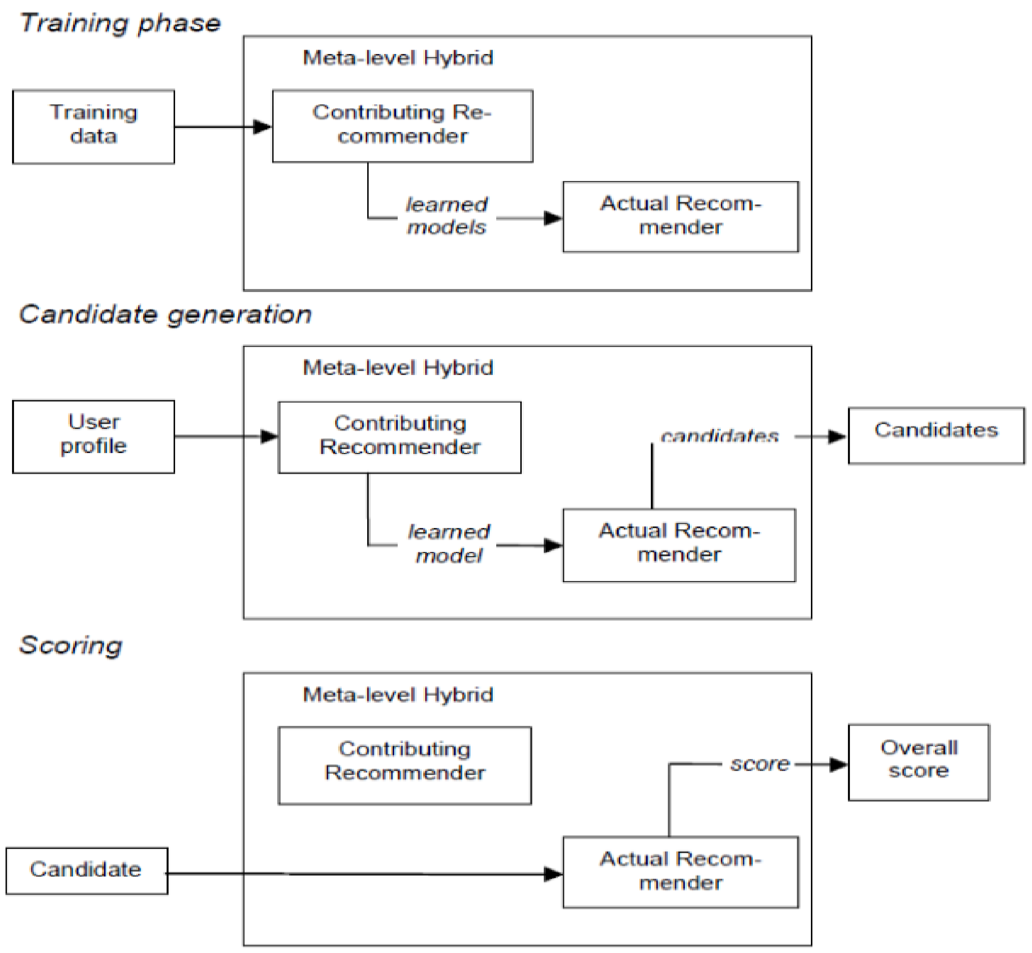


Figure 2.13 Meta-level hybrid (Burke & Felfernig, 2011)

2.4 NEIGHBORHOOD FORMATION

Neighbors simply means a group of likeminded users with a target user or a set of similar items with the items that have been already been identified as being preferred by the target user. Finding nearest neighbors, a variety of similarity methods have been researched, such as cosine similarity, which is widely used, the Pearson correlation (Resnick, et al., 1994), weight amplification and inverse user frequency, and default rating (Breese, et al., 1998), including probability-based approaches. According to the results of the selected similarity measure, particular users or items with highest similarity are identified as neighbors. The number of neighbors may be varied depending on the characteristics of the domains and the application. However, it also has significant impact on the quality of results from the CF (Sarwar, et al., 2000). The recommendation systems have to determine the size of the neighborhood in order to compute the prediction results effectively. That is, if the size of the neighborhood is too small, it becomes difficult to obtain accurate results, whereas the large size of neighborhood brings about the complexity of computation even though the results might be more accurate.

2.4.1 PREDICTION GENERATION

Once k neighbors are found, various methods can be used to combine the ratings of neighbors to compute a prediction value on unrated items for the target user. The preference rating of each neighbor is usually weighted by the similarity value which is computed when the neighbors are determined. The more a neighbor is similar to a target user or item, the more influence he or she has for calculating a prediction value. After predicting how a target user will like particular items which have not been rated yet by the target user, the Top-*N* item set, a set of ordered items with a higher predicted value, is identified and recommended. The target user can present feedback of whether the target user actually likes the recommend Top-*N* items or how much he/she prefers those items as scaled ratings. The feedback can be used for updating a similarity model in order to help the model reflect changes of preferences as well as a measurement of whether the recommendation system performs well or not. Some of the most commonly used similarity computation techniques in recommendation systems are correlation-based similarity, cosine-based similarity and adjusted cosine similarity.

2.5 GROUP RECOMMENDATION SYSTEMS

The first scientific publications regarding recommendation systems for groups date from the late nineties (McCarthy & Anagnost, 1998). From then, many researchers have already investigated how the current state-of-the-art recommendation algorithms can be adapted in order to generate group recommendations. In the literature, group recommendations have mostly been generated either by aggregating the users' individual recommendations into recommendations for the whole group (aggregating recommendations) or by aggregating the users' individual preference models into a preference model of the group (aggregating preferences) (Berkovsky & Freyne, 2010). In this dissertation, these strategies are referred to as grouping strategies. The first grouping strategy (aggregating recommendations) generates recommendations for each individual user using a general recommendation algorithm. Subsequently, the recommendation lists of all group members are aggregated into a group recommendation list which (hopefully) satisfies all group members. Different approaches to aggregate the recommendation lists have been proposed during the last decade. Most of them make a decision based on the algorithm's prediction score, i.e. a prediction of the user's rating score for the recommended item. The higher the prediction score is, the better the match between the user's preferences and the recommended item. Aggregating the users' individual recommendations into group recommendations has some advantages. For instance, the resulting recommendations can be directly linked to the individual recommendations, which makes them easy to explain based on the explanations of the traditional recommendation (Herlocker, et al., 2000). Conversely, the link between the group recommendations and the individual recommendations makes it less likely to identify unexpected, surprising items (O'Connor, et al., 2001).

The second grouping strategy (aggregating preferences) combines the users' preferences into group preferences. This way, the opinions and preferences of individual group members constitute a group preference model reflecting the interests of all members. In the literature, different approaches have been proposed to aggregate the members' preferences, but still no consensus exists about the optimal solution (Baltrunas, et al., 2010); (Masthoff, 2004). After aggregating the members' preferences, the group's preference model is treated as a pseudo user in order to produce recommendations for the group using a traditional recommendation algorithm. Compared to aggregating the individual recommendation lists, aggregating the users' preferences increases the chance of finding serendipitously valuable recommendations. On the other hand, aggregating the preferences may lead to group suggestions that lie outside the range of any individual recommendation list, which may be disorienting to the users and difficult to explain (Herlocker, et al., 2000).

2.5.1 GROUP AGGREGATION STRATEGIES/ RANK AGGREGATION FUNCTIONS

Various group modelling strategies for making recommendations have been proposed and tested to aggregate the individual group user's preferences into a recommendation for the group (Masthoff, 2011) evaluated ten strategies inspired from social choice theory. The strategies are discussed below:

2.5.1.1 Plurality Voting (also called 'first past the post').

Each voter votes for his or her most preferred alternative. The alternative with the most votes wins. When a sequence of alternatives needs to be selected, this method can be used repetitively: first, an election is held for the first place in the sequence, next for the second place, etc. In the example on table 2.1, John would like to vote for A, E, or I (all ratings of 10). Adam for B, D, F, or H, and Mary for A. Traditionally in Plurality voting, each individual has only one vote, so, John would have to decide whether to vote for A, E, or I. If John were aware of the preferences of the others, then it is likely that he would vote for A, as with Mary's vote this would secure a majority. In this scenario, with only three individuals and ten items, it is quite likely that a vote would end in a tie. If John were to vote for E or I, then all three individuals would vote for a different item, and there would be no winner. It would clearly be in John's interest to vote A. In this case, the television would decide on a choice for the group, and as the television would be aware of all individuals' preferences, it could easily accommodate strategic voting, to prevent ties. The interpretation of Plurality Voting in this context will therefore be that rather than giving individuals one vote, they are allowed to vote for all items that have the highest rating. In the example on table 2.1, this gives A two votes, and it becomes the start of the sequence. Next, John likes to vote for E or I, Adam for B, D, F, or H and Mary for E. With two votes E has most votes, and becomes second in the sequence (Masthoff, 2004).

Table 2.1: Plurality Voting Example (Masthoff, 2004)

	1	**2**	**3**	**4**	**6**	**7**	**8**	**10**	
John	A,E,I	E,I	I	I	H,J	J	G	C	Group List: AEI(I,D)HJ(BG)C
Adam	B,D,F,H	B,D,F,H	B,D,F,H	B,D,H	B,H	B	B	C	
Mary	A	E	F	D,I	H,J	J	B,G	C	
Group	A	E	F	D,I	H	J	B,G	C	

Instead of using the method respectively, each voter could vote for *x* alternatives (with *x* being the length of the sequence) (Masthoff, 2004).

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
John	AEI			AEIF	AEIFHJ		AEIFHJDG	
Adam	BDFH				BDFHJC		BDFHJCE	BDFHJCEG
Mary	A	AE	AEF	AEFDI		AEFDIHJ		AEFDIHJBG
Group	A	AE	AEF	F(AEFDI)	FHJ(AEDI)	J	FHJED(AI)	FHJEDG(AIB)

2.5.1.2 Utilitarian Strategy.

Utility values for each alternative (expressing the expected happiness) are used, instead of just using ranking information (as in plurality voting). This can be done in multiple ways: Additive. Ratings are added, and the larger the sum the earlier the alternative appears in the sequence. Note that the resulting group list will be exactly the same as when taking the average of individual ratings. For this reason (Masthoff, 2002) called this the "Average strategy". The strategy (often in a weighted form, where weights are attached to individual ratings) is used in multi-agent systems (Hogg & Jennings, 1999) and Collaborative filtering. This is also the strategy used in the INTRIGUE system (Ardissono, et al., 2003), with a weighting depending on the number of people in the subgroup and the subgroup's relevance (children and disabled had a higher relevance).

Table 2.2: Utilitarian Strategy Example (Masthoff, 2004)

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	
John	10	4	3	6	10	9	6	8	10	8	
Adam	1	9	8	9	7	9	6	9	3	8	Group List: (E,F)H(D,J)AI B G C
Mary	10	5	2	7	9	8	5	6	7	6	
Group	21	18	13	22	26	26	17	23	20	22	

Multiplicative. Instead of *adding* the utilities, they are multiplied, and the larger the product the earlier the alternative appears in the sequence.

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	
John	10	4	3	6	10	9	6	8	10	8	
Adam	1	9	8	9	7	9	6	9	3	8	Group List:
Mary	10	5	2	7	9	8	5	6	7	6	F E H J D I (B,G) A C
Group	100	1801	48	378	630	648	180	432	210	384	

2.5.1.3 Borda Count (Borda, 1781).

Points are awarded to each alternative according to its position in the individual's preference list: the alternative at the bottom of the list gets zero points, the next one up one point, etc. For instance, in the example below John has the lowest rating for C, and hence, C is awarded 0 points. A problem arises when an individual has multiple alternatives with the same rating. We have decided to distribute the points. So, for example, in Mary's list B and G share the place one up from the bottom and get (1+2)/2 = 1 1/2 points each. To obtain the group preference ordering, the points awarded for the individuals are added up (Masthoff, 2004).

Table 2.3: Borda Count Example (Masthoff, 2004)

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	
John	8	1	0	2½	8	6	2½	4½	8	4½	
Adam	0	7½	4½	7½	3	7½	2	7½	1	4½	Group List:
Mary	9	1½	0	5½	8	7	1½	3½	5½	3½	F E A (H,D) I J B G C
Group	17	10	4½	15½	19	20½	6	15½	14½	12½	

2.5.1.4 *Copeland Rule (Copeland 1951)*

This is a form of majority voting. It orders the alternatives according to the Copeland index: the number of times an alternative beats other alternatives minus the number of times it loses to other alternatives (Klamler, 2003). For instance, in the example A beats B as both John and Mary prefer it.

Table 2.4: Copeland Rule Example (Masthoff, 2004)

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	
A	0	-	-	-	0	-	-	-	0	-	
B	+	0	-	+	+	+	0	+	+	+	
C	+	+	0	+	+	+	+	+	+	+	Group List: E A F I D H J (B,J)C
D	+	-	-	0	+	+	-	0	0	-	
E	0	-	-	-	0	-	-	-	-	-	
F	+	-	-	-	+	0	-	-	-	-	
G	+	0	-	+	+	+	0	+	+	+	
H	+	-	-	0	+	+	-	0	+	-	
I	0	-	-	0	+	+	-	-	0	-	
J	+	-	-	+	+	+	-	+	+	0	
Index	+7	-6	-9	+1	+8	+5	-6	0	+3	-3	

Note that in the example the resulting group list is almost identical to the one resulting from repetitive plurality voting

2.5.1.5 Approval Voting.

Voters are allowed to vote for as many alternatives as they wish. This is intended to promote the election of moderate alternatives: alternatives that are not strongly disliked. This type of voting is used by several professional societies, like the IEEE (Masthoff, 2004). In the example below, we could assume that John, Mary, and Adam vote for all alternatives with a rating above a certain threshold. They could vote for all alternatives with a rating higher than 5, as this means voting for all alternatives they like at least a little bit.

Threshold 5.

Table 2.5: Approval Voting Example (Masthoff, 2004)

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	
John	1				1	1	1	1	1	1	
Adam		1	1	1	1	1	1	1		1	Group List:
Mary	1			1	1	1		1	1	1	(D, E, F, H, J)(G,A,I)(B,C)
Group	2	1	1	2	3	3	2	3	2	3	

Threshold 6.

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	
John	1				1	1		1	1	1	
Adam		1	1	1	1	1		1		1	Group List:
Mary	1			1	1	1			1		(E, F)(A,D,H,I,J)(B,C)G
Group	2	1	1	2	3	3	0	2	2	2	

2.5.1.6 Least Misery Strategy.

Make a new list of ratings with the minimum of the individual ratings. Items get selected based on their rating on that list, the higher the sooner. The idea behind this strategy is that a group is as happy as its least happy member. (O'Connor, et al., 2001) uses this strategy, assuming groups of people going to watch a movie together tend to be small and a small group to be as happy as its least happy member. A disadvantage is that a minority opinion can dictate the group: if everybody really wants to see something, but one person does not like it, then it will never be seen.

Table 2.6 Least Misery Strategy Example (Masthoff, 2004)

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	
John	10	4	3	6	10	9	6	8	10	8	
Adam	1	9	8	9	7	9	6	9	3	8	Group List: F,E,(H,J,D),G,B,I,C,A
Mary	10	5	2	7	9	8	5	6	7	6	
Group	1	4	2	6	7	8	5	6	3	6	

2.5.1.7 Most Pleasure Strategy.

Make a new list of ratings with the maximum of the individual ratings. Items get selected based on their rating on that list, the higher the sooner (Masthoff, 2004).

Table 2.7: Most Pleasure Strategy Example (Masthoff, 2004)

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	
John	10	4	3	6	10	9	6	8	10	8	
Adam	1	9	8	9	7	9	6	9	3	8	Group List: (A,E,I),(B,D,F,H),(C,J)G
Mary	10	5	2	7	9	8	5	6	7	6	

Group	10	9	8	9	10	9	6	9	10	8	

2.5.1.8 *Average without Misery Strategy.*

Make a new list of ratings with the average of the individual ratings, but without items that score below a certain threshold (say 4) for individuals (Masthoff, 2004).

Table 2.8: Average without Misery Strategy Example (Masthoff, 2004)

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	
John	10	4	3	6	10	9	6	8	10	8	Group List:
Adam	1	9	8	9	7	9	6	9	3	8	(E,F)H(D,J),B,G (threshold 4)
Mary	10	5	2	7	9	8	5	6	7	6	(E,F)H(D,J),I,B (threshold 5)
Group	-	18	-	22	26	26	17	23	-	22	

(McCarthy & Anagnost, 1998) uses a more complex version of this strategy. Their users rate all music stations, from +2 (really love this music) to -2 (really hate this music). These ratings are converted to positive numbers (by adding 2) and then squared to widen the gap between popular and less popular stations. An Average without Misery strategy is used to generate a group list. To avoid starvation and always picking the same station, a weighted random selection is made from the top m stations of the list (m being a system parameter).

2.5.1.9 *Fairness Strategy.*

Top items from all individuals are selected. When items are rated equally, the others' opinions are taken into account. The idea behind this strategy is that it is not so bad to watch something you hate, as long as you get to watch the things you really love as well. This strategy is often applied when people try to fairly divide a set of items: one person chooses first, then another, till everybody has made one choice. Next, everybody chooses a second item, often starting with the person who had to choose last on the previous round. It continues till all items have been used (Masthoff, 2004). In our example, if we assume John chooses first, then John would like A, E, or I. He could choose E because it causes the least misery to others and has the highest average. Next it is Adam's turn. Adam would like B, D, F, or H. He could choose F because it has the best ratings for the others. Mary would choose A (her highest rating). Next, Mary would like E, which has already been shown, and then F, which also has already been shown. Therefore, it makes sense to let Adam choose. He likes B, D, or H. He chooses H, as that has the best ratings for the others. Following this strategy, we could end up with a group list like: E, F, A, H, I, D, B, etc. The list would, of course, be different if we let Mary or Adam choose first. However, we would expect A to be within the first three items, as it is the item Mary prefers most.

2.5.1.10 Most Respected Person Strategy (Also called "Dictatorship").

The ratings of the most respected person are used --in our example assume that is Adam--, only taking the ratings of the others into account to choose between similarly rated items. The idea behind this strategy is that groups may be dominated by one person. For instance, some research shows that the television remote control is most often operated by the oldest male present. Similarly, adults may have more influence than children (could depend on the time of day, adults having more influence later in the day). Visitors may have more influence than inhabitants of the house. Special circumstances, like birthdays, illness, etc. can influence who is "the most respected" person on a particular moment. This strategy is used often in collaborative filtering under the name of “the nearest neighbour strategy”: only the preferences of the individual closest in taste to the outsider are used. A more sophisticated use of differences in social status would be to assign weights to the individuals' ratings. This has also been used in collaborative filtering and in the (Ardissono, et al., 2003), both of which use a weighted additive utilitarian strategy.

Table 2.9: Most Respected Person Strategy Example (Masthoff, 2004)

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	
John	10	4	3	6	10	9	6	8	10	8	
Adam	1	9	8	9	7	9	6	9	3	8	Group List: F H D B J C E G I A
Mary	10	5	2	7	9	8	5	6	7	6	
Group	1	9	8	9	7	9	6	9	3	8	

2.6 VIRTUAL GROUPS

(Cambridge dictionary, 2018) states that the term “virtual” is used to describe something that can be done or seen using computers or the internet instead of going to a place, meeting people in person. (Oxford English Dictionary, 2019) further clarifies that when talking about something virtual, it is not physically existing as such but made by software to appear to do so. On the other hand, (Merriam-Webster’s collegiate dictionary, 2014) iterates that the term “virtual” refers to an instance being on or simulated on a computer or computer network such as (a): occurring or existing primarily online or (b) of, relating to, or existing within a virtual reality.

A group can be defined as 'a small number of people with complementary skills who are committed to a common purpose, performance goals, and approach for which they hold themselves mutually accountable. Group sizes can differ with the minimum group members as two.

(Johnson, et al., 2002) defines virtual teams as groups of individuals who interact through various communication technologies to accomplish its common goals. The term ‘‘virtual team’’ is becoming more prevalent as teams move from being primarily ‘‘co-located,’’ where team members are located in one physical location, to ‘‘virtual,’’ where team members are geographically unrestricted (Lipnack & Stamps, 1997). These virtual teams rely on Internet technologies such as videoconferencing and chat rooms to interact and become functional.

(Levenson & Cohen, 2002) considers that interacting in virtual groups can assure a series of advantages consisting in the enhancement in creativity and innovation of products, the facilitation of learning, and the development of positive attitudes facing the tasks that need to be achieved. Virtual groups can also support knowledge sharing between members and sustain the application of obtained information and skills (Johnson, et al., 2002).

Virtual teams can offer flexibility, responsiveness, and diversity of perspectives in ways that differ from traditional groups. Despite these benefits, however, virtual teams encounter numerous challenges due to their dispersion and communication limitations, which can impede their effectiveness, or at least require great efforts to accommodate to the virtual environment and virtual partners (Walther, et al., 2005).

2.7 SOCIAL NETWORK SITES

(Boyd & Ellison, 2007) defines social network sites as web-based services that allow individuals to (1) construct a public or semi-public profile within a bounded system, (2) articulate a list of other users with whom they share a connection, and (3) view and traverse their list of connections and those made by others within the system. The nature and nomenclature of these connections may vary from site to site.

The global system of networked computers, servers, and routers known as the Internet has transformed many aspects of modern society and social interaction. The online distribution of goods and services, for instance, has influenced almost every industry and has radically transformed many. Alongside commerce-oriented technological development has been a rise in what has been termed "social media." One of the most significant developments connected to social media is the rise of social network sites (SNSs) such as Facebook, LinkedIn, MySpace, Cyworld, and Google Plus. Although sites of this nature first emerged around 1997, they rose to cultural significance as a phenomenon in 2003, when Friendster first attracted mass media attention. Less than a decade later, millions of people of all ages across the globe have joined SNSs (Anderson & Bernoff, 2010).

What makes social network sites unique is not that they allow individuals to meet strangers, but rather that they enable users to articulate and make visible their social networks. This can result in connections between individuals that would not otherwise be made, but that is often not the goal, and these meetings are frequently between "latent ties" (Haythornthwaite , 2011) who share some offline connection. On many of the large SNSs, participants are not necessarily "networking" or looking to meet new people; instead, they are primarily communicating with people who are already a part of their extended social network.

While SNSs have implemented a wide variety of technical features, their backbone consists of visible profiles that display an articulated list of Friends (Acquisti & Gross, 2006) who are also users of the system. Profiles are unique pages where one can "type oneself into being" (Sundén, 2005). After joining an SNS, an individual is asked to fill out forms containing a series of questions. The profile is generated using the answers to these questions, which typically include descriptors such as age, location, interests, and an "about me" section. Most sites also encourage users to upload a profile photo. Some sites allow users to enhance their profiles by adding multimedia content or modifying their profile's look and feel. Others, such as Facebook, allow users to add modules ("Applications") that enhance their profile.

2.8 RECOMMENDATION SYSTEMS EVALUATION APPROACHES

Recommendation system evaluation assess the effectiveness of recommendation algorithms. The importance of Evaluation of recommendation system depends on numbers of approaches or algorithms (Sridevi, et al., 2016). There are three primary types of evaluation or recommendation systems, corresponding to user studies, online evaluations and offline evaluations with historical data sets. The first two types involve users, although they are conducted in slightly different ways (Aggarwal, 2016).

2.8.1 User Studies

A user study is conducted by recruiting a set of test subject, and asking them to perform several tasks requiring an interaction with the recommendation system. While the subjects perform the tasks, we observe and record their behaviour, collecting any number of quantitative measurements, such as what portion of the task was completed, the accuracy of the task results, or the time taken to perform the task. In many cases we can ask qualitative questions, before, during, and after the task is completed. Such questions can collect data that is not directly observable, such as whether the subject enjoyed the user interface, or whether the user perceived the task as easy to complete. A typical example of such an experiment is to test the influence of a recommendation algorithm on the browsing behaviour of news stories. In this example, the subjects are asked to read a set of stories that are interesting to them, in some cases including related story recommendations and in some cases without recommendations. We can then check whether the recommendations are used, and whether people read different stories with and without recommendations. We can collect data such as how many times a recommendation was clicked, and even, in certain cases, track eye movement to see whether a subject looked at a recommendation. Finally, we can ask quantitative questions such as whether the subject thought the recommendations were relevant (Godin, et al., 2008).

However, user study has some weaknesses. First, the cost of user study is very high. In one hand, a large number of testers must be recruited and on the other hand, testers must finish a large number of interacting tasks. Therefore, in common practice, the number of testers and the size of testing tasks should be controlled. Meanwhile, the quality of the data to be collected should be guaranteed in statistical significance. Besides, the distributions of testers should be considered, such as the distributions in hobbies and interests, sex ration, ages, activity levels, etc. should all be similar to those of the users in a real system. It’s also important that, when collecting data from the users’ tasks, the purposes of the testing should not be told to the testers before testing, in order to avoid the subjective tendency in users’ behaviours and answers, such as accepting more recommended information unintentionally (Chen & Liu, 2017).

2.8.2 Online Evaluations

In many realistic recommendation applications, the designer of the system wishes to influence the behaviour of users. We are therefore interested in measuring the change in user behaviour when interacting with different recommendation systems. For example, if users of one system follow the recommendations more often, or if some utility gathered from users of one system exceeds utility gathered from users of the other system, then we can conclude that one system is superior to the other, all else being equal. The real effect of the recommendation system depends on a variety of factors such as the user’s intent (e.g. how specific their information needs are, how much novelty vs. how much risk they are seeking), the user's context (e.g. what items they are already familiar with, how much they trust the system), and the interface through which the recommendations are presented. Thus, the experiment that provides the strongest evidence as to the true value of the system is an online evaluation, where the system is used by real users that perform real tasks. It is most trustworthy to compare a few systems online, obtaining a ranking of alternatives, rather than absolute numbers that are more difficult to interpret (Godin, et al., 2008).

It is worth noting that, there is some risk in online experiment. For example, if a recommendation system recommends too many unrelated items during online experiment, then the users' trust on the recommendation system will be reduced rapidly and will not care for the items being recommended after the real system is deployed finally. This is the worst and most unacceptable situation in commercial practices (Chen & Liu, 2017).

2.8.3 OFFLINE EVALUATIONS

An offline experiment is performed by using a pre-collected data set of users choosing or rating items. Using this data set we can try to simulate the behaviour of users that interact with a recommendation system. In doing so, we assume that the user behaviour when the data was collected will be similar enough to the user behaviour when the recommendation system is deployed, so that we can make reliable decisions based on the simulation. Offline experiments are attractive because they require no interaction with real users, and thus allow us to compare a wide range of candidate algorithms at a low cost. The downside of offline experiments is that they can answer a very narrow set of questions, typically questions about the prediction power of an algorithm. In particular, we must assume that users' behaviour when interacting with a system including the recommendation system chosen will be modelled well by the users' behaviour prior to that system's deployment. Thus we cannot directly measure the recommender's influence on user behaviour in this setting. Therefore, the goal of the offline experiments is to filter out inappropriate approaches, leaving a relatively small set of candidate algorithms to be tested by the costlier user studies or online experiments. A typical example of this process is when the parameters of the algorithms are tuned in an offline experiment, and then the algorithm with the best tuned parameters continues to the next phase (Godin, et al., 2008).

The advantage of offline analytics is that it doesn't need the interaction from real users, so it can be implemented at a low cost and can test and evaluate the performance of different kinds of recommendation algorithms quickly. But the disadvantages are that such experiments can usually be used in evaluating the prediction accuracy of the algorithms or Top N precision of recommendation, and can do little in the evaluation of serendipity or novelty and so on (Mobasher, et al., 2007).

2.9 ACCURACY METRICS IN RECOMMENDATION SYSTEMS

Several metrics have been proposed in order to evaluate the performance of the various models employed by a RS. The metrics allow the evaluation of the quality of the numeric prediction (Cremonesi, et al., 2008). The following accuracy metrics can be used to measure how much prediction is close to the true numerical rating expressed by the user.

2.9.1 MEAN ABSOLUTE ERROR

Mean Absolute Error (MAE) between ratings and predictions is a widely used metric (Xiaoyuan & Taghi, 2009). MAE is a measure of the deviation of recommendations from their true user-specified values. For each ratings-prediction pair $< p_{i,j}, r_{i,j} >$ this metric treats the absolute error between them, i.e., $\left|< p_{i,j} - r_{i,j} >\right|$ equally. The MAE is computed by first summing these absolute errors of the N corresponding ratings-prediction pairs and then computing the average. The lower the MAE, the more accurately the recommendation engine predicts the user ratings. (Xiaoyuan & Taghi, 2009) made use of equation 2.28 to calculate MAE.

$$MAE = \frac{\sum_{\{i,j\}} \left|p_{i,j} - r_{i,j}\right|}{N} \tag{2.28}$$

where N is the total number of ratings over all users, $p_{i,j}$ is the predicted rating for user i on item j and $r_{i,j}$ is the actual rating.

2.9.2 ROOT MEAN SQUARE ERROR

This is a statistical accuracy metric that is slightly different from Mean Absolute Error (Xiaoyuan & Taghi, 2009). Once rating-prediction difference is calculated, its power of 2 is taken. After summing them up and dividing them by the total number of rating-prediction pairs and taking square root of it, Root Mean Square Error can be found. (Xiaoyuan & Taghi, 2009) calculates RMSE as given in equation 2.29.

$$RMSE = \sqrt{\frac{1}{N}\sum_{\{i,j\}}(p_{i,j} - r_{i,j})^2} \qquad (2.29)$$

where N is the total number of ratings over all users, $p_{i,j}$ is the predicted rating for user *i* on item j, and $r_{i,j}$ is the actual rating.

2.9.3 PRECISION AND RECALL

Precision measures the number of correctly identified items as a percentage of the number of items identified. In other words, it measures how many of the items that the system identified were actually correct, regardless of whether it also failed to retrieve correct items (Chung, et al., 2018). The higher the Precision, the better the system is at ensuring that what has been identified is correct. (Gunawardana & Shani, 2009) formally defines precision and recall as shown in equation 2.30 and 2.31 respectively.

$$Precision = \frac{tp}{tp + fp} \qquad (2.30)$$

where *tp* represents true positives and *fp* represents false positives.

Recall measures the number of correctly identified items as a percentage of the total number of correct items. In other words, it measures how many of the items that should have been identified actually were identified, regardless of how many spurious identifications were made. The higher the Recall rate, the better the system is at not missing correct items (Chung, et al., 2018).

$$Recall = \frac{tp}{tp + fn} \qquad (2.31)$$

where *tp* represents true positives and *fp* represents false positives.

2.9.4 F-MEASURE

The F-measure is often used in conjunction with Precision and Recall, as a weighted average of the two. If the weight is set to 0.5 (which is usually the case), Precision and Recall are deemed equally important (Ye, et al., 2012). (Espindola & Ebecken, 2005) formally defines F-Measure as shown in equation 2.32 and 2.33.

$$F - measure = \frac{(\beta^2 + 1)\ P * R)}{(\beta^2 R) + P} \tag{2.32}$$

where β reflects the weighting of P vs. R. If P and R are to be given equal weights, then equation 2.33 can be used.

$$F_1 = \frac{P * R}{0.5 * (P + R)} \tag{2.33}$$

2.10 GAP FROM THE REVIEW OF LITERATURE

RSs emerged in the mid-90s, and earlier times, they depended solely on the overall ratings representing the preferences of items by users (Liu, et al., 2011). RS's aim is to suggest or recommend products or items that are available and the user might like or be interested in. The suggestions or recommendations are made using personalized information and filtering technologies (Zuva, et al., 2012).

Currently, a significant research has been made in the development of more complex methods in making recommendations. Researchers are striving to understand the user and items in a more in-depth fashion. Aside the keywords in the traditional profile features, a lot of profiling techniques that are based on data mining, structures that describe a user's interest or items features are applied to build the profiles for users and items. There are extensions over more difficult modeling techniques, such as machine learning, probabilistic models, regression trees etc (Asabere, 2013). Some other extensions focus on the simple aspect of Recommendation Systems for instance flexibility, privacy, effectiveness etc.

In this study, we extended the aspect of Recommendation System by integrating individual recommendations and group recommendations and also extended the recommendations for virtual groups based on the ratings similarities of users and similarity of items, in this case movies, making use of social network (Facebook) derived data. While there is a vast amount of research done on recommendation systems, there is not much on group recommendations and there is none on virtual group recommendations.

A lot of researchers developed RS's tailored to satisfy the individual needs. However, there has not been as much focus on group RSs, specifically group movie RSs. PolyLens (O'Connor, et al., 2001) recommends movies to groups of users. It is an extension to the MovieLens system, which is based on an individual's taste as inferred from ratings and collaborative filtering. But PolyLens focuses on increased group average.

TV program recommendations for multiple viewers, based on user profile merging, provides TV program users with a hybrid filtering technique to make recommendation to multiple viewers through merging user profiles, then merge individual user preferences on features, and finally recommends a sequence of programs and tries to make nobody in the group really unhappy (Avoid misery) (Yu, et al., 2006). (Yu, et al., 2006). Those, however, are not designed for virtual groups.

2.11 CHAPTER SUMMARY

This chapter has discussed the related theories in regard to Recommendation System, different filtering techniques including content based filtering, collaborative filtering and hybrid filtering techniques. The strength and weaknesses of each of these techniques were highlighted. Similarity computation methods were discussed. Virtual groups and social network sites were reviewed. The chapter further discussed evaluation approaches and accuracy metrics used in recommendation system. It also discusses what has been done previously by other researchers concerned with individual recommendations, group recommendations, individual movie recommendations and group movie recommendations.

The next chapter discussed the research methodology.

3 RESEARCH METHODOLOGY

3.1 INTRODUCTION

This chapter introduces the research methodology and highlights techniques that were used in the development, testing and evaluation of the virtual group movie recommendation systems prototype. The chapter presents the processes by which the objectives of the research were accomplished. Research methodology is a systematic way to solve a problem. It is a science of studying how research is to be carried out. Essentially, the procedures by which researchers go about their work of describing, explaining and predicting phenomena are called research methodology. It is also defined as the study of methods by which knowledge is gained. Its aim is to give the work plan of research (Rajasekar, et al., 2013). Research can be categorized into two basic techniques namely quantitative and qualitative research. This investigation pursued a qualitative approach which consisted of an extensive exploration of the literature on recommendation systems, development of a prototype with the ability to carry out individual and group recommendations, and finally the evaluation of the prototype.

There are three major filtering techniques in recommendation systems namely content based, collaborative and hybrid recommendation systems as the previous chapter discussed in detail. The chapter further clarified that content based filtering methods are based on a description of the item and profile of user's preferences, collaborative filtering methods are based on collecting and analysing a large amount of information on user's behaviours, activities or preferences and predicting what users will like based on their similarity to other users while hybrid approach combines collaborative filtering and content-based filtering to make recommendations (Belkin & Croft, 1992); (Ekstrand, et al., 2011); (Ardissono, et al., 2003).

The literature review chapter further stated that collaborative filtering techniques can further be split into memory-based and model-based. Predictions for memory based approach make use of the user database completely. Statistical methods are used by the system to find the like-minded set of users or neighbours who share similar interests with the active user. The implementation of a memory based system can either be item-item or user-user based (Akhil & Shelbi, 2017). Model-based collaborative filtering use the pure rating data to estimate or learn a model to make predictions. The model can be a data mining or machine learning algorithm as (Xiaoyuan & Taghi, 2009) states.

3.2 VIRTUAL GROUP MOVIE RECOMMENDATION SYSTEM

In this study, a model based matrix factorization algorithm of collaborative filtering technique was used. In this perspective, the algorithm was deployed because of its accuracy in making predictions and because of its ability to improve prediction performance (Koren, et al., 2009). In addition, this algorithm has been proved to be a better option to address the issues of data sparsity, over-fitting and convergence speed (Aleksandrova, 2017).

The general approach of the virtual Group Movie Recommendation system followed these steps:

1. Prediction of movie ratings using matrix factorization
2. Standard ranking of movies above pre-set threshold value (3.0)
3. Recommendations of 3 movies to individuals
4. Plurality check
5. Generation of virtual groups
6. Standard ranking to generate group recommendation list
7. Recommendation of movies to a virtual group, together with a list of group members

The approach followed by the virtual group movie recommendation system is illustrated by figure 3.1.

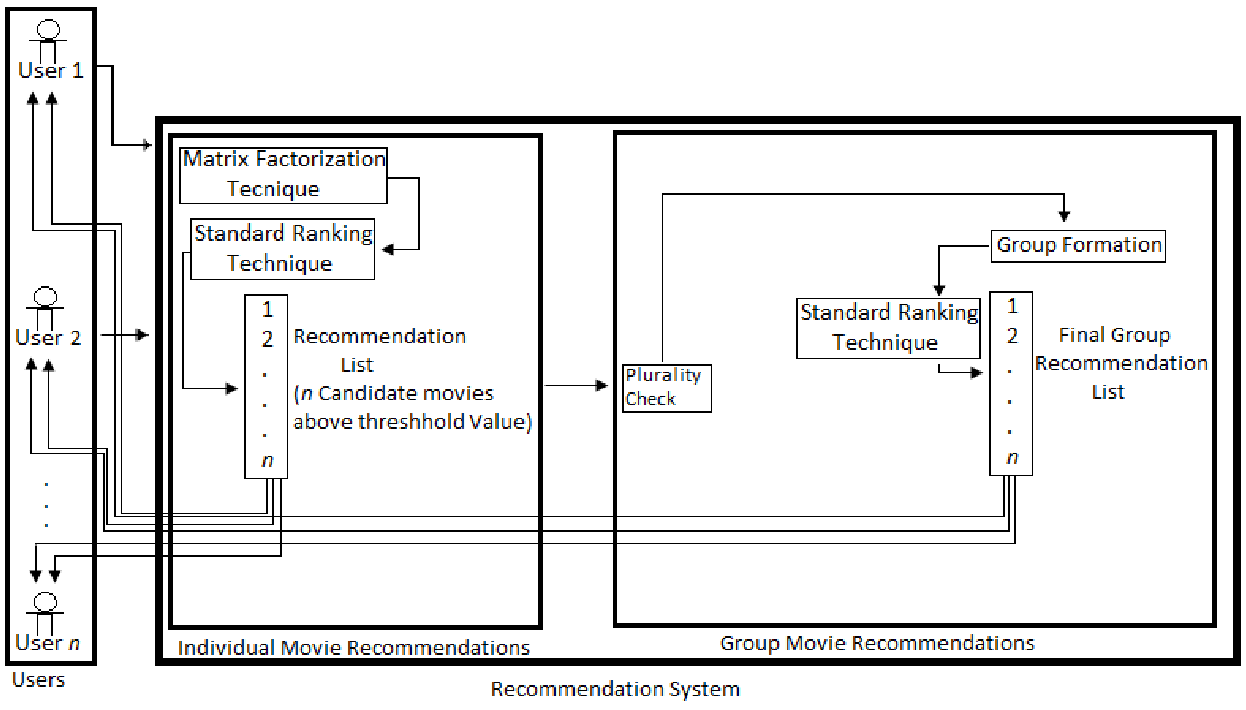

User 1
User 2
User *n*
Users
Matrix Factorization Tecnique
Standard Ranking Technique
1 2 . . . *n*
Recommendation List (*n* Candidate movies above threshhold Value)
Individual Movie Recommendations
Plurality Check
Group Formation
Standard Ranking Technique
1 2 . . . *n*
Final Group Recommendation List
Group Movie Recommendations
Recommendation System

Figure 3.1 Virtual Group Movie Recommendation System Architecture derived from (Thai-Nghe, et al., 2017); (Masthoff, 2004); (Dara, et al., 2019).

As illustrated by figure 3.1, the recommendation system predicts ratings for all unrated movies using matrix factorization algorithm of collaborative filtering technique. The system then uses standard ranking technique to determine top *n* predicted movies, after which the top *n* predicted movies that are above the pre-set threshold are sent to each user. Based on common movies recommendations with predicted rating above the pre-set threshold, the systems goes through the plurality check, generates virtual groups, undergoes standard ranking technique, recommends top *n* movies to the generated virtual groups and sends movie recommendations to each group member. The list is sent together with the list of group members. The individual movie recommendations part of figure 3.1, which focuses on the matrix factorization technique, standard ranking and movie recommendations to individual users is derived from (Thai-Nghe, et al., 2017), while the group movie recommendation part which is all about plurality check, formation of groups, standard ranking and movie recommendations for formed virtual groups is derived from (Masthoff, 2004) and (Dara, et al., 2019).

3.2.1 PREDICTION OF MOVIE RATINGS USING MATRIX FACTORIZATION

Matrix factorization approximates matrix X by the product of two smaller matrices W and H, i.e. $X \approx WH^T$. In the context of recommendation systems the matrix X is the partially observed ratings matrix, $W \in \mathbb{R}^{U \times K}$ is a matrix where each row i is a vector containing the K features describing the item i. Let $w_{uk}$and h_{ik}be the elements of W and H, respectively, then the rating given by a user u to an item i was predicted using equation 3.1.

$$\hat{r}_{ui} = \sum_{k=1}^{K} w_{uk} h_{ik} = (WH^T)_{u,i} \tag{3.1}$$

where W and H are the model parameters and can be learned by optimizing a given criterion using stochastic gradient descent.

3.2.2 STANDARD RANKING OF MOVIES ABOVE PRE-SET THRESHOLD VALUE

This is a commonly used approach for ranking the items in RSs. In this approach, the predicted rating of movies was ranked from highest to lowest. In our case, all the predicted movies above the pre-set threshold value, which was 3, were ranked in descending order, which means the highly predicted item comes first in the list and the lowest predicted at the bottom of the list. This is to ensure accuracy of the recommended items. This was achieved by utilizing equation 3.2 the same way it was used by (Taurshia & Kanmani, 2013).

$$rank_{standard}(i) = R * (u, i)^{-1} \tag{3.2}$$

where $R * (u, i)$is the predicted rating. The power of -1 indicates that the items with the highest predicted are recommended to user. This approach increases the accuracy in recommendation system (Liang, et al., 2018).

3.2.3 Recommendation of 3 Movies To Individuals

After movie rating predictions were computed using matrix factorization technique, and the standard ranking of movies to ensure accuracy was carried out, each user was sent a list of 3 recommended movies stating with the highest predicted to the lowest predicted rating (descending). Figure 3.2 demonstrates an example of the steps carried out in generating the final movie recommendation list for each user.

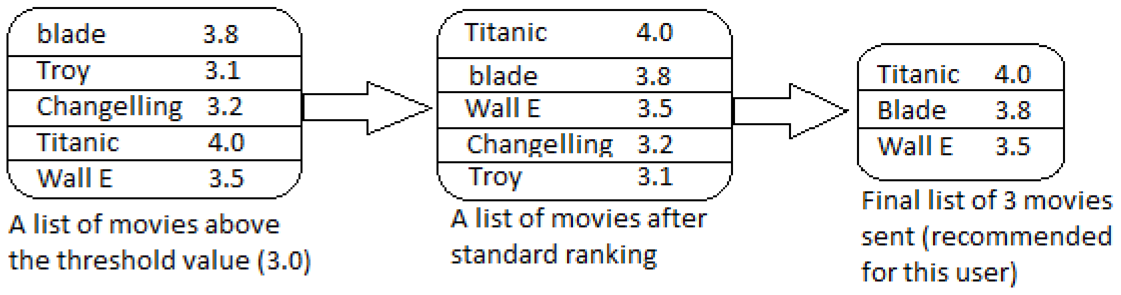

blade 3.8
Troy 3.1
Changelling 3.2
Titanic 4.0
Wall E 3.5
A list of movies above the threshold value (3.0)
Titanic 4.0
blade 3.8
Wall E 3.5
Changelling 3.2
Troy 3.1
A list of movies after standard ranking
Titanic 4.0
Blade 3.8
Wall E 3.5
Final list of 3 movies sent (recommended for this user)

Figure 3.2 Individual Movie Recommendation List derived from (Liang, et al., 2018).

3.2.4 Generation Of Virtual Groups

After the individual recommendations were made for each user, the recommendation system identifies 4 movies with the highest predicted rating (above the threshold 3) by common users. 4 virtual groups were then formed out of these users based on the predictions above the threshold.

3.2.5 Plurality Check

The literature review chapter of this dissertation discussed 10 group aggregation strategies/ rank aggregation functions. In this work, amongst all the strategies discussed, the plurality voting aggregation strategy was picked. The assumption is that the movie that is recommended to more people has an advantage of being watched by these people as a virtual group. The implication is that the virtual group movie recommendation system went through the plurality check to pick the 4 movies that have the highest number of voting's (recommendations). All these movies were picked amongst the movies the users have not seen, and that have a prediction above the threshold. Table 3.1 shows an example of how the strategy was applied.

Table 3.1: Plurality Voting Strategy example derived from (Masthoff, 2004)

Movie	**User 1**	**User 2**	**User 3**	**User 4**	**User 5**	**User 6**	**Group Average**
Blade	4.0		4.8		5.0		13.8
Troy			4.5	3.3			7,8
Changelling	3.5	3.2		3.1	3.0		12.8
Titanic		4.5	3.5		3.2		11.2
Wall E				3.0		5.0	8.0

In the case of the example stated on table 3.1, it is evident that the movie Blade has the highest group average of 13.8. But the plurality voting strategy would recommend the movie Changelling even though it has a group average of only 12.8. This would be done because the movie Changelling has the most number of voting's (recommendations). It is worth noting that all the movies that go through this stage of plurality check are all predicted to be above the threshold.

3.2.6 Standard Ranking To Generate Group Recommendation List

On completion of the plurality check, the recommendation system undergoes the standard ranking process for group recommendations the same way it does with individual recommendations. Just like with individual recommendations list, this is done so as to increase accuracy. The list is displayed to virtual group members in descending order, starting with the movie with the highest group average to the lowest.

3.2.7 Virtual Groups Movie Recommendations, Together With A List Of Group Members

When both the plurality check and the standard ranking processes are complete, the recommendation list reaches its final stage where it sends the final list to each virtual group member. The final group movie recommendation list is sent to all the virtual group members together with the names/user ID’s of all the group members.

3.3 EXPERIMENTAL PREDICTIVE ACCURACY METRICS

Another essential part of this investigation was to evaluate the developed prototype so as to determine its effectiveness and accuracy in generating predictions and making recommendations. Chapter 2 discussed several evaluation metrics used in recommendation systems. These metrics may be selected depending on the goal that the researcher wishes to achieve. To measure the predictive performance of the system, in order to obtain the error, the system can encounter during the implementation, the Mean Absolute Error and Root Mean Squared Error were calculated. These are two of the most common metrics used to measure accuracy for continuous variables.

3.3.1 MAE

Mean Absolute Error (MAE): was used to measures the average magnitude of the errors in a set of predictions, without considering their direction. As MAE measures accuracy for continuous variables, it was used to determine the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight. Equation 3.3 illustrates how MAE was calculated.

$$MAE = \frac{1}{n}\sum_{j=1}^{n}\left|y_j - \hat{y}_j\right| \tag{3.3}$$

where $\hat{y}$ is the predicted rating, y is the actual rating and n is number of occurrences/instances (amount of ratings)

3.3.2 RMSE

The RMSE is a quadratic scoring rule which measures the average magnitude of the error. Expressing the formula in words, the difference between forecast and corresponding observed values are each squared and then averaged over the sample. Finally, the square root of the average is taken. Since the errors are squared before they are averaged, the RMSE gives a relatively high weight to large errors. This means the RMSE is most useful when large errors are particularly undesirable. Equation 3.4 was used for calculating the RMSE.

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(y_j - \hat{y}_j\right)^2} \tag{3.4}$$

where $\hat{y}$ is the predicted rating, y is the actual rating is n = number of occurrences/instances (the amount of ratings)

Both the MAE and RMSE can range from 0 to ∞ where ∞ is the maximum error depending on the rating scale of the measured application. They are negatively-oriented scores, which mean lower values are better.

In this investigation, both the MAE and the RMSE were used together to diagnose the variation in the errors in a set of predictions. The RMSE will always be larger or equal to the MAE; the greater difference between them, the greater the variance in the individual errors in the sample. If the RMSE=MAE, then all the errors are of the same magnitude

3.4 DATASETS

A publicly available dataset based on Group Recommender Systems Enhanced by Social Elements, constructed by Lara Quijano from the Group of Artificial Intelligence Applications (GIGA) obtained from (http://gaia.fdi.ucm.es/research/happymovie/download). The dataset consists of a sample of 58 users and 50 movies selected from the MovieLens dataset. Datasets were in form of two separate files, movies dataset and ratings dataset, in notepad files.

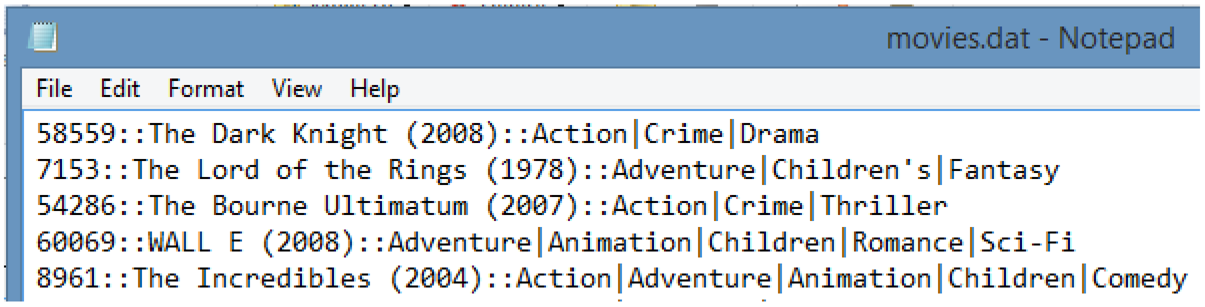
3.4.1 MOVIES DATASET

The movies file contains fifty movies that users had to rate movies he or she may have seen. This set of movies consists of a sample of all the different genres that existed and movie types, so that with fifty movies a general idea of what types of movies a given user liked could be formed. The dataset had movie_id, movie_name, and genre attributes. Table 3.2 gives detailed descriptions of the attributes in the movies dataset while figure 3.3 shows an overview of the movies dataset Microsoft excel file. Only the first ten elements of the movies dataset and its attributes are displayed.

Table 3.2: Movies dataset attribute description table

Attribute	**Description**

MovieLens Id Number	A unique number identifying each movie in the dataset. Each and every single movie has a special movie_id for identification. movie_ids used here are similar to those used by grouplens in movielens
MovieLens Name	This attribute holds the title of the movie. Similar to movie id, the movie names used in the dataset are identical to movie names used by grouplens in movielens
Year	This attribute holds a specific year a movie was released
genre	A movie genre attribute refers to a motion-picture category based on similarities either in the narrative elements or in emotional response to the film. The genres contained in the dataset are Action, Adventure, Animation, Children, Comedy, Crime, Documentary, Drama, Fantasy, FilmNoir, Horror, Musical, Mystery, Romance, SciFi, Thriller, War and Western



```
58559::The Dark Knight (2008)::Action|Crime|Drama
7153::The Lord of the Rings (1978)::Adventure|Children's|Fantasy
54286::The Bourne Ultimatum (2007)::Action|Crime|Thriller
60069::WALL E (2008)::Adventure|Animation|Children|Romance|Sci-Fi
8961::The Incredibles (2004)::Action|Adventure|Animation|Children|Comedy
```

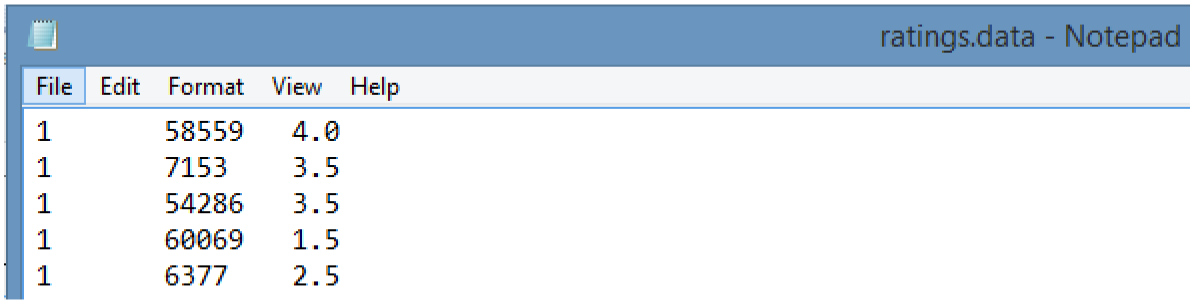
Figure 3.3 Overview of movies dataset notepad file

3.4.2 Ratings Dataset

The ratings file had ratings ranging from 0.5 rating as the minimum possible rating and 5.0 as the highest possible rating. The file had user_id, movie_id and rating attributes. Table 3.3 gives detailed descriptions of the attributes in the ratings dataset. Figure 3.4 shows an overview of the ratings dataset after the data. Only the first ten elements of the dataset are displayed here.

Table 3.3: Ratings dataset attribute description table

Attribute	**Description**
user_id	A unique number identifying user in the dataset. Each and every single movie has a special user_id for identification. The dataset consists of 58 users. As a result, user_ids begin at 1 to 58.
movie_id	A unique number identifying movies in the dataset. Each and every single movie has a special movie_id for identification. movie_ids used here are similar to those used by grouplens in movielens
rating	Rating attribute holds the ratings that 58 users gave to the 50 movies that we presented them ratings range from 0.5 which is awarded for movies the user least liked to 5.0 for movies the user considered flawless in every department and really left a long lasting impression on them.



```
1       58559   4.0
1       7153    3.5
1       54286   3.5
1       60069   1.5
1       6377    2.5
```

Figure 3.4 Overview of ratings notepad dataset

3.5 DATA SCRUBBING AND PRE-PROCESSING

When conducting this investigation, the author understood that data quality is important and without accurate, good-quality data, a significant amount of time, effort and resources would be wasted trying to develop a recommendation system. As a result, it was ensured that only the most accurate and relevant data was entered and used in the datasets. Data scrubbing also referred to as data cleansing, may be described as the identification of errors within a dataset, and the removal or correction of those errors. This process involves insuring that your data is correct, consistent and usable by identifying and removing or correcting any errors or corruptions in the data. After an establishment of which attributes to use, the irrelevant data was then left out and the movies and ratings datasets were loaded into the data frame, using python, in

Jupiter notebook integrated development environment. In order to obtain meaningful data from the datasets, movies and ratings datasets were merged into one data frame using movie_id attribute.

After the data was cleaned, the author then pre-processed the data so as to obtain information from it as the raw data is non-comprehensive. By so doing, the author added visual aspect to the data, making it easier and quicker to understand. This process included opening movies data set (movies.dat) and ratings dataset (ratings.data), removing attributes that are not relevant to the study, removing or correcting errors from movies.dat and ratings.data and saving the changes on both movies.dat and ratings.data files.

After finalizing the pre-processing, the datasets contained 58 users and 50 movies with 1696 given ratings out of possible 2900 ratings. This implies that 58.5% ratings were given and the ratings expected were in a format 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 5.5 and 5.0. Figure 3.5 gives a general overview of the data after pre-processing.

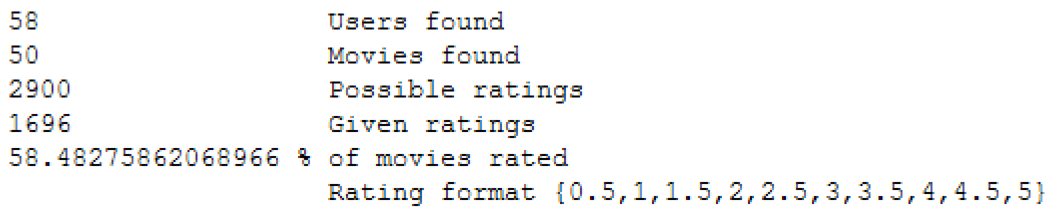

```
58                   Users found
50                   Movies found
2900                 Possible ratings
1696                 Given ratings
58.48275862068966 % of movies rated
                     Rating format {0.5,1,1.5,2,2.5,3,3.5,4,4.5,5}
```

Figure 3.5 Results after pre-processing data

3.6 CHAPTER SUMMARY

In this chapter, the methodology that was used to carry out the investigation was introduced. The steps followed by the virtual group movie recommendation were illustrated. The chapter also discussed how the matrix factorization was deployed to make predictions. An overview of the ratings and movies datasets, the structure of datasets, and how data was prepared to run the experiments was illustrated. The chapter further discussed how Mean Absolute Error and Root Mean Squared Error were used so as to evaluate the prototype for accuracy in making predictions. The author demonstrated the technique that was used to generate groups and make group recommendations.

The next chapter discusses experiments, results and interpretation of those results.

4 EXPERIMENTS, RESULTS AND INTERPRETATION

4.1 INTRODUCTION

The sole purpose of this chapter is to present and interpret the experimental results of the prototype. The chapter also discusses all the basic steps that were taken to evaluate the prototype. In addition, the exploration of the happy movie dataset is done in this chapter. The programming environment used in the design of the prototype is also discussed briefly. The results presented here will be based primarily on both the individual recommendations and group recommendations. This chapter will therefore:

- Discuss the programming environment used in building the prototype
- Explore datasets
- Present a graphical visualization of datasets used for running tests
- Carry out basic calculations pertaining to ratings found in the datasets (Sparsity, density and parsity)
- Deploy the matrix factorization technique and make predictions
- Simulate the prototype by making individual movie recommendations and analyse the results obtained from individual movie recommendations
- Simulate formation of a virtual group
- Simulate virtual group movie recommendations and analyse the results obtained from virtual group movie recommendation simulation
- Evaluate the accuracy of the virtual group movie recommendation system prototype

4.2 PROGRAMMING ENVIRONMENT

Because it has several technical advantages over other programming languages, and because its practical application covers several industries including data science and machine learning, the prototype was designed and developed using Python programming language, and Jupyter notebook as the integrated development environment (IDE). To summarize the technical advantages that make Python a powerful programming language, often preferred over other programming languages, we can say Python is free and constantly updated, it that can be used in multiple domains, calculations processing does not require too much time and its syntax is intuitive allowing for complex quantitative computations. In addition, Python has a rich ecosystem for scientific inquiry in the form of many proven, and popular open-source packages including the ones used in this project which are explained below:

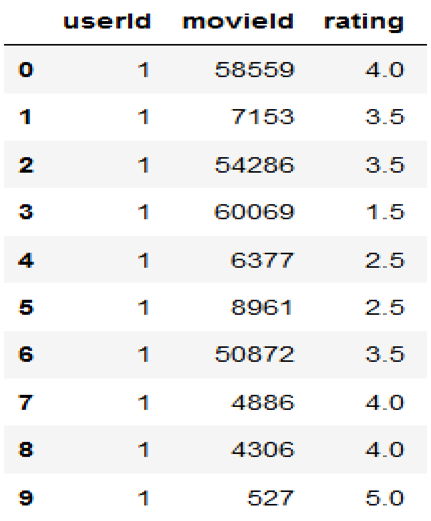
- NumPy is important to perform scientific computing with Python. It encompasses an assortment of high-level mathematical functions to operate on multi-dimensional arrays and matrices.
- SciPy works in association with NumPy arrays and offers effective routines for numerical integration and up-gradation.
- Pandas also developed on top of NumPy, delivers data structures and operations to change numerical tables and time series.
- Matplotlib is a 2D plotting library. It offers data visualizations in the form of histograms, power spectra, bar charts, and scatterplots with minimal coding lines.
- Scikit-learn acts as a machine learning library that leads to classification, regression, and clustering algorithms that involve support vector machines, logistic regression, naive Bayes, random forests, and gradient boosting.
- Seaborn is a python data visualization library based on Matplotlib. It provides high level interface for drawing attractive and informative statistical graphs.
- OS provides portable way of using operating system dependent functionality.

4.3 DATASETS EXPLORATION

In order to familiarize ourselves with the data we were working on, the datasets were explored and the results also analysed. This was done so as to get a quick and simple view of the relevant features of our datasets. The exploration involved exploring the ratings dataset, movies dataset and the merged data frame of both datasets. This stage also included importing all the necessary python libraries including Pandas, Seaborn, NumPy, OS and Matplotlib. Furthermore, it was on this stage that the movies and ratings datasets were read.

4.3.1 RATINGS DATASET

The prototype was fed with ratings dataset. Figure 4.1 shows the results in python, Jupyter notebook integrated development environment. It is noteworthy that only the first 10 elements of the dataset are displayed here. The output shows the user id, movie id and the rating rendered by the user for a specific movie.

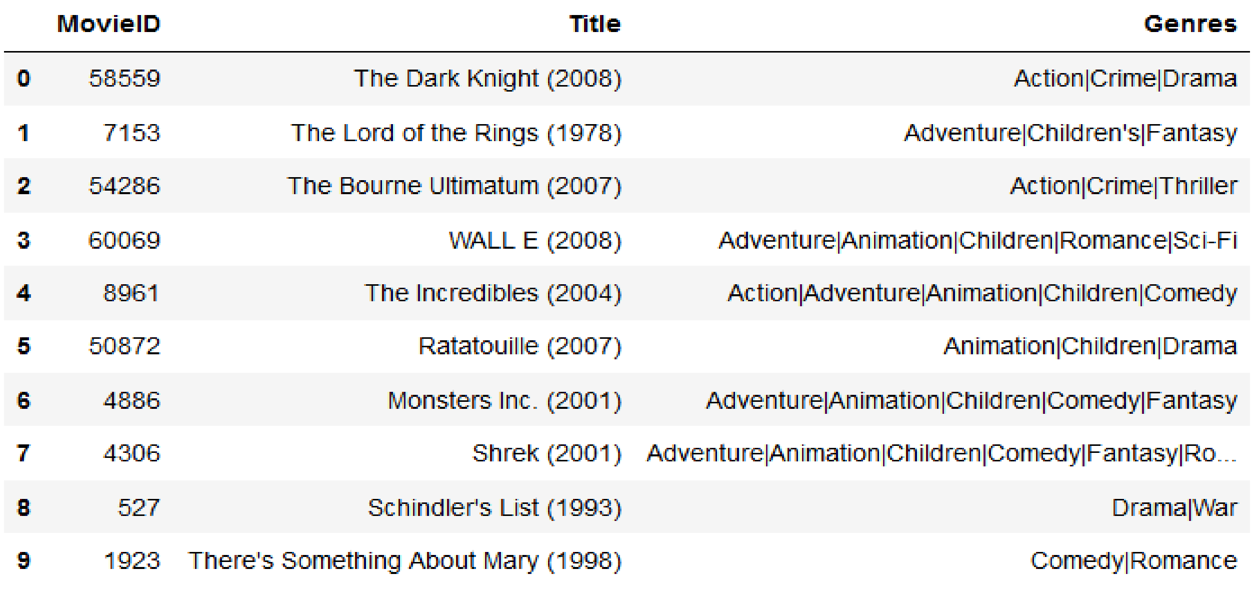


	userId	movieId	rating
0	1	58559	4.0
1	1	7153	3.5
2	1	54286	3.5
3	1	60069	1.5
4	1	6377	2.5
5	1	8961	2.5
6	1	50872	3.5
7	1	4886	4.0
8	1	4306	4.0
9	1	527	5.0

Figure 4.1 Jupyter notebook ratings file overview

4.3.2 MOVIES DATASET

Similarly to the ratings dataset, the movies dataset was also loaded into the prototype and its contents were read. Figure 4.2 shows the first 10 elements of the movies dataset. The attributes displayed are the movie id, title and genres.

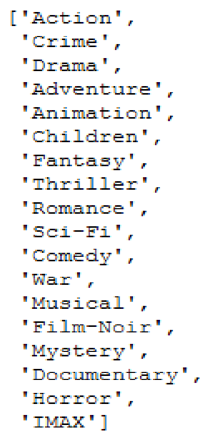


	MovieID	Title	Genres
0	58559	The Dark Knight (2008)	Action|Crime|Drama
1	7153	The Lord of the Rings (1978)	Adventure|Children's|Fantasy
2	54286	The Bourne Ultimatum (2007)	Action|Crime|Thriller
3	60069	WALL E (2008)	Adventure|Animation|Children|Romance|Sci-Fi
4	8961	The Incredibles (2004)	Action|Adventure|Animation|Children|Comedy
5	50872	Ratatouille (2007)	Animation|Children|Drama
6	4886	Monsters Inc. (2001)	Adventure|Animation|Children|Comedy|Fantasy
7	4306	Shrek (2001)	Adventure|Animation|Children|Comedy|Fantasy|Ro...
8	527	Schindler's List (1993)	Drama|War
9	1923	There's Something About Mary (1998)	Comedy|Romance

Figure 4.2 Movies file overview

4.3.2.1 Movies Genre Extraction

Movies genre attributes were further extracted from the movies dataset. Figure 4.3 shows the genres found in the dataset. As shown by the figure, eighteen different genres were found.

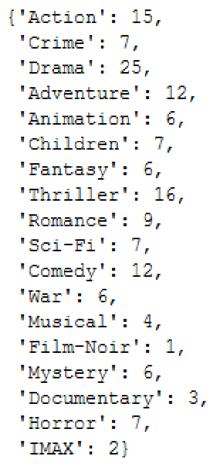


```
['Action',
 'Crime',
 'Drama',
 'Adventure',
 'Animation',
 'Children',
 'Fantasy',
 'Thriller',
 'Romance',
 'Sci-Fi',
 'Comedy',
 'War',
 'Musical',
 'Film-Noir',
 'Mystery',
 'Documentary',
 'Horror',
 'IMAX']
```

Figure 4.3 List of genres found in the dataset

4.3.2.2 Genre Count

Amongst the eighteen movie genres found, we calculated count per genre with the results displayed by figure 4.4. It can be established that most movies (twenty-five) contain drama while very few (only one) movies contain film-noir genre.



```
{'Action': 15,
 'Crime': 7,
 'Drama': 25,
 'Adventure': 12,
 'Animation': 6,
 'Children': 7,
 'Fantasy': 6,
 'Thriller': 16,
 'Romance': 9,
 'Sci-Fi': 7,
 'Comedy': 12,
 'War': 6,
 'Musical': 4,
 'Film-Noir': 1,
 'Mystery': 6,
 'Documentary': 3,
 'Horror': 7,
 'IMAX': 2}
```

Figure 4.4 Count of movies per genre

4.3.2.3 Genre versus Movies Graphical View

A movies versus genres graph was then plotted using the count of each movie genre as displayed by figure 4.4. The histogram on figure 4.5 shows a graphical view of each genre against the total count of that genre. The histogram basically displays the data in figure 4.4 but in a graphical manner so as to have a clearer overview of the number of movies per genre.

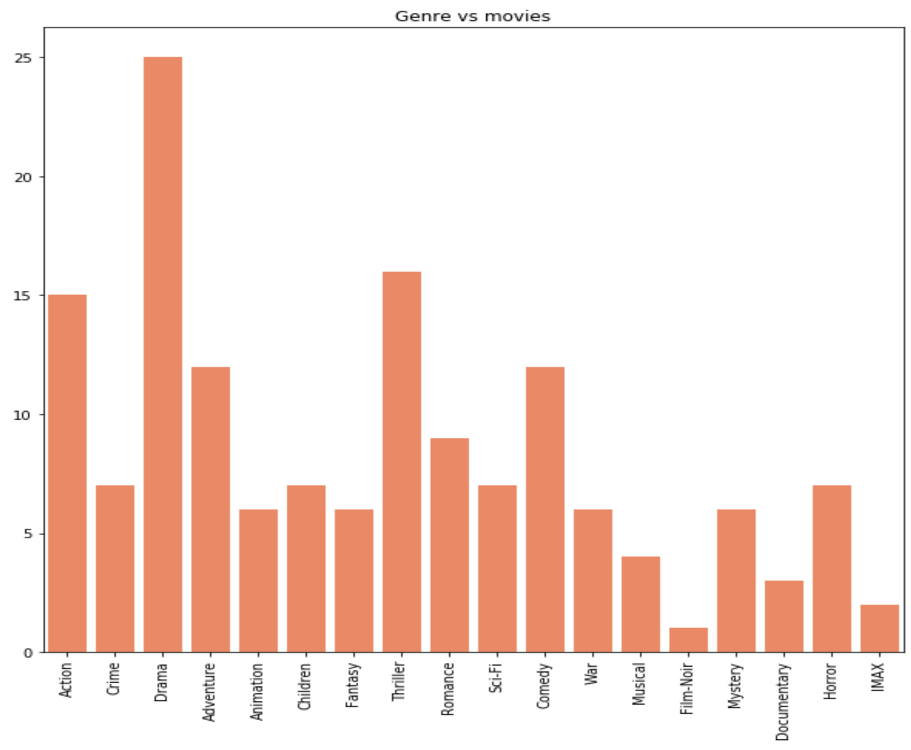
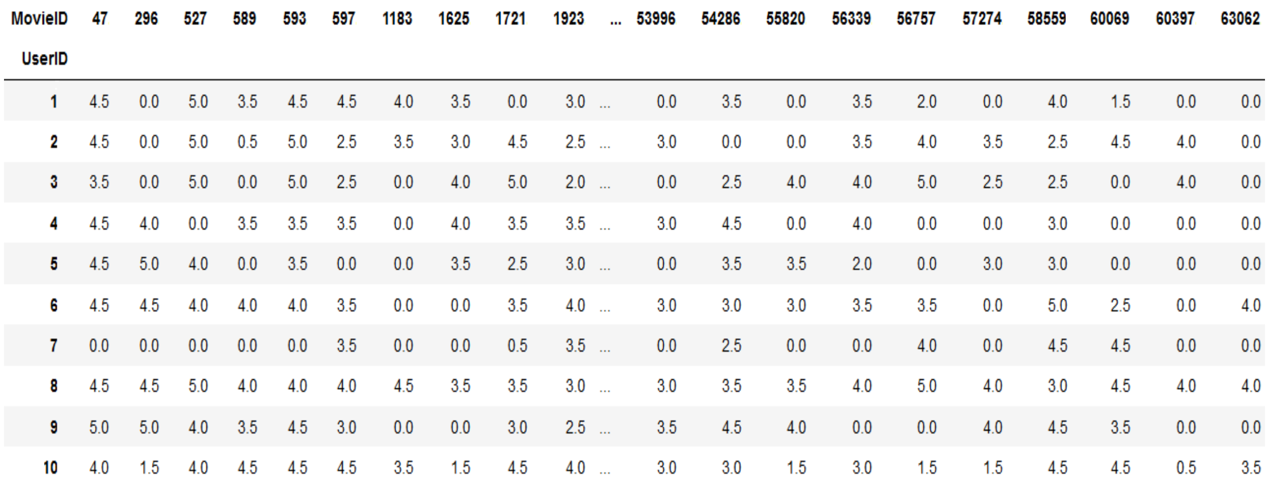


Figure 4.5 Genre versus movie histogram

4.3.3 Merged Dataset

After both the movies.dat and rating.data datasets were loaded into the prototype, the files were merged into one data frame which gives a more meaningful information as demonstrates by figure 4.3. The X-axis of the data frame shows the movie ids while the Y-axis shows the user ids. Inside the data frame are the ratings given by specific users for specific movies. As shown, the ratings range from 0.0 to 5.0. In the data frame, 0.0 implies that user has not rated (watched) a specific movie. In essence, the ratings given begin from 0.5 to 5.0. Figure 4.6 displays only the first 10 users out of 58 and only 20 movies out of 50.



MovieID	47	296	527	589	593	597	1183	1625	1721	1923	...	53996	54286	55820	56339	56757	57274	58559	60069	60397	63062
UserID																					
1	4.5	0.0	5.0	3.5	4.5	4.5	4.0	3.5	0.0	3.0	...	0.0	3.5	0.0	3.5	2.0	0.0	4.0	1.5	0.0	0.0
2	4.5	0.0	5.0	0.5	5.0	2.5	3.5	3.0	4.5	2.5	...	3.0	0.0	0.0	3.5	4.0	3.5	2.5	4.5	4.0	0.0
3	3.5	0.0	5.0	0.0	5.0	2.5	0.0	4.0	5.0	2.0	...	0.0	2.5	4.0	4.0	5.0	2.5	2.5	0.0	4.0	0.0
4	4.5	4.0	0.0	3.5	3.5	3.5	0.0	4.0	3.5	3.5	...	3.0	4.5	0.0	4.0	0.0	0.0	3.0	0.0	0.0	0.0
5	4.5	5.0	4.0	0.0	3.5	0.0	0.0	3.5	2.5	3.0	...	0.0	3.5	3.5	2.0	0.0	3.0	3.0	0.0	0.0	0.0
6	4.5	4.5	4.0	4.0	4.0	3.5	0.0	0.0	3.5	4.0	...	3.0	3.0	3.0	3.5	3.5	0.0	5.0	2.5	0.0	4.0
7	0.0	0.0	0.0	0.0	0.0	3.5	0.0	0.0	0.5	3.5	...	0.0	2.5	0.0	0.0	4.0	0.0	4.5	4.5	0.0	0.0
8	4.5	4.5	5.0	4.0	4.0	4.0	4.5	3.5	3.5	3.0	...	3.0	3.5	3.5	4.0	5.0	4.0	3.0	4.5	4.0	4.0
9	5.0	5.0	4.0	3.5	4.5	3.0	0.0	0.0	3.0	2.5	...	3.5	4.5	4.0	0.0	0.0	4.0	4.5	3.5	0.0	0.0
10	4.0	1.5	4.0	4.5	4.5	4.5	3.5	1.5	4.5	4.0	...	3.0	3.0	1.5	3.0	1.5	1.5	4.5	4.5	0.5	3.5

Figure 4.6 Movies and ratings merged data frame

4.4 DATASET VISUALIZATION

Data visualization is viewed by many disciplines as a modern equivalent of visual communication. It involves the creation and study of the visual representation of data. To communicate information clearly and efficiently, data visualization uses statistical graphics, plots, information graphics and other tools. Numerical data may be encoded using dots, lines, or bars, to visually communicate a quantitative message (Friendly, 2009).

So as to identify patterns and trends in the dataset, various graphs were applied using Matplotlib and Seaborn libraries. The graphs were meant to give a clearer overview of the HappyMovie ratings and movies datasets.

4.4.1 RATING COUNT HISTOGRAM

The histogram shown in figure 4.7 shows the specific rating given against the total number of occurrences for that rating. The figure clearly states that the 4.0 rating has the highest occurrences been given 350 time, followed by 3.5 rating that was given 300 times, etc. with the rating of 1.0 with the lowest number of occurrences.

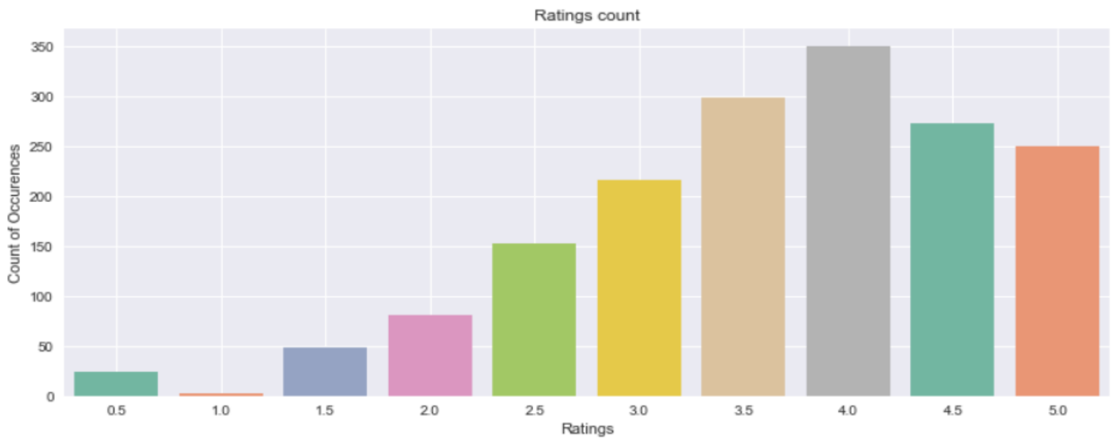


Figure 4.7 Ratings count histogram

4.4.2 RATED MOVIES DISTRIBUTION GRAPH

On chart of a 'normal' distribution, showing the classic 'bell curve' shape, shown on figure 4.8, the mean (or average) is the vertical line at the centre, and the vertical lines to either side represent intervals of one, two and three sigma. The percentage of data points that would lie within each segment of that distribution are shown. From the normal distribution plot we can infer that the mean (average) of movies rated is thirty, the least number of movies rated is nine and the highest number of ratings given by a single user is forty-nine out of fifty movies.

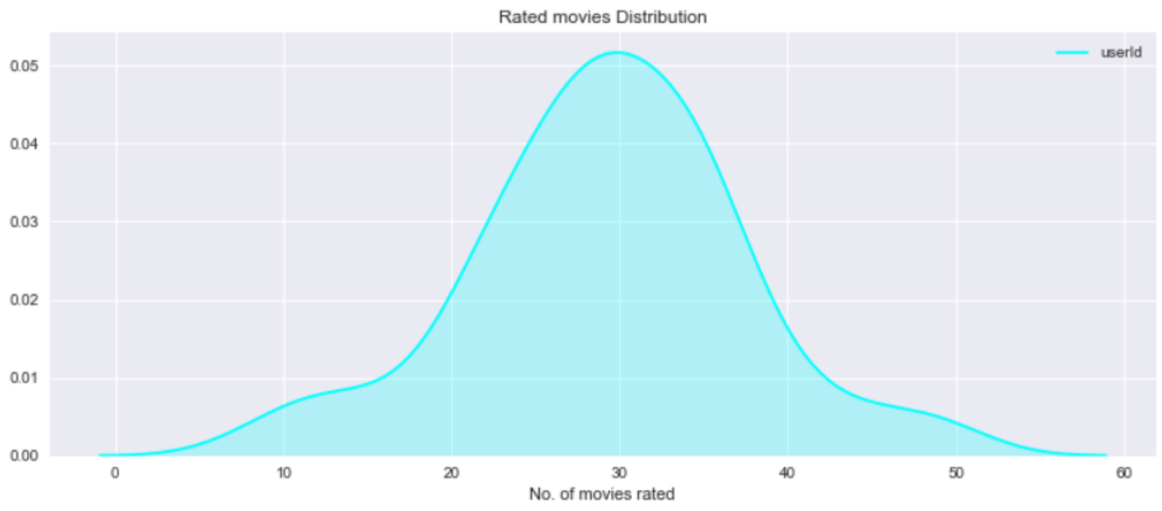


Figure 4.8 Rated movie distribution plot

4.4.3 Distribution Of A Number Of Times A Movie Has Been Rated Graph

Similarly to figure 4.8, figure 4.9 below shows the distribution of the number of times a movie has been rated with the average as thirty-five.

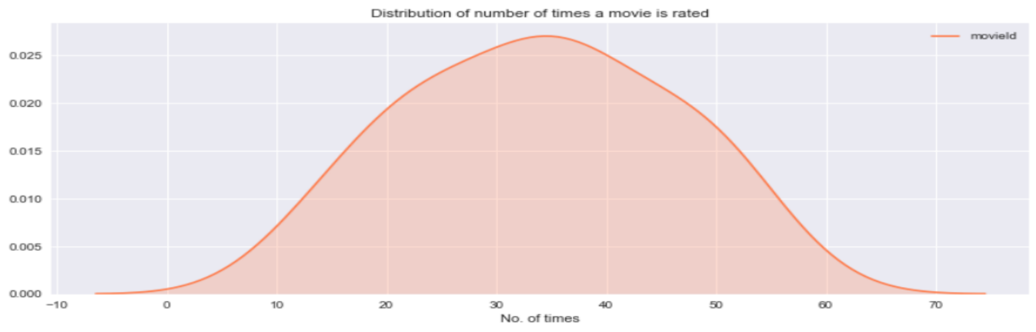


Figure 4.9 Distribution of the number of times a movie has been rated plot

4.4.4 Rating Distribution Violin

Violin graphs are another way to visualize data distribution. They show a lot of information about the data. They are essentially called violin plots because they look a lot like the body of a violin. These plots show several descriptive statistics including the median and the interquartile range. Figure 4.10 shows a rating distribution violin created from the ratings dataset. The white dot represents the median. The thick grey bar in the centre represents the interquartile range and the thin grey line represents the 95% confidence interval. On each side of the grey line is a kernel density estimation to show the distribution shape of the data. Wider sections of the violin plot represent a higher probability that members of the population will take on the given value; the skinnier sections represent a lower probability.

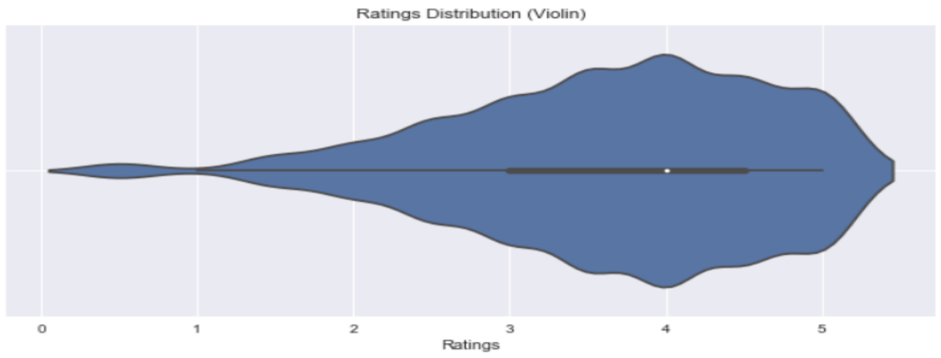


Figure 4.10 Rating distribution violin plot

4.4.5 RATING DISTRIBUTION KERNEL DENSITY ESTIMATION (KDE)

Density Plots visualizes the distribution of data over a continuous interval or time period. They are a variation of a Histogram that uses kernel smoothing to plot values, allowing for smoother distributions by smoothing out the noise. The peaks of a Density Plot help display where values are concentrated over the interval. Compared to histograms, kernel density estimation plots are better at determining the distribution shape because they're not affected by the number of bins used (each bar used in a typical histogram). The kernel most often used is a Gaussian (which produces a Gaussian bell curve at each data point). From HappyMovie ratings dataset, the ratings kde is represented by figure 4.11.

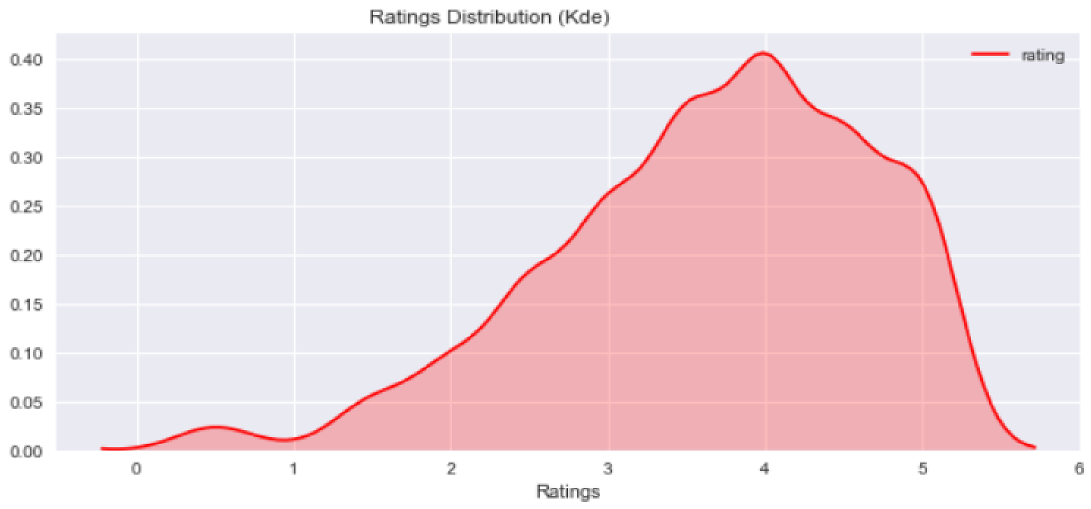
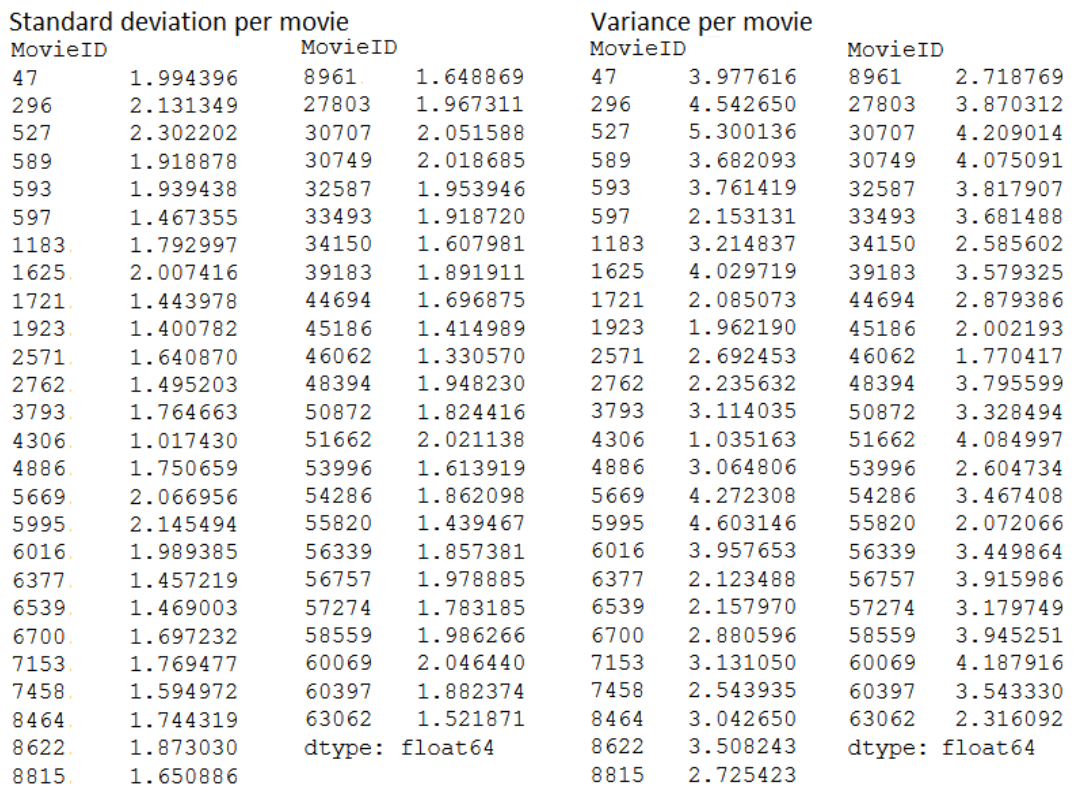


Figure 4.11 Gaussian ratings kernel distribution estimation (kde) plot

4.5 BASIC CALCULATIONS

4.5.1 STANDARD DEVIATION AND VARIANCE PER MOVIE

In addition to visualizing data in form of plots, the variance and standard deviation per movie were calculated and the results obtained are shown by figure 4.12.



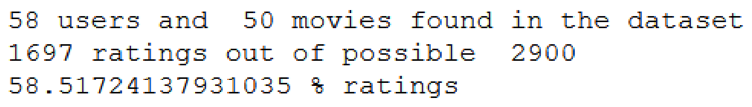
```
Standard deviation per movie
MovieID                      MovieID
47       1.994396            8961     1.648869
296      2.131349            27803    1.967311
527      2.302202            30707    2.051588
589      1.918878            30749    2.018685
593      1.939438            32587    1.953946
597      1.467355            33493    1.918720
1183     1.792997            34150    1.607981
1625     2.007416            39183    1.891911
1721     1.443978            44694    1.696875
1923     1.400782            45186    1.414989
2571     1.640870            46062    1.330570
2762     1.495203            48394    1.948230
3793     1.764663            50872    1.824416
4306     1.017430            51662    2.021138
4886     1.750659            53996    1.613919
5669     2.066956            54286    1.862098
5995     2.145494            55820    1.439467
6016     1.989385            56339    1.857381
6377     1.457219            56757    1.978885
6539     1.469003            57274    1.783185
6700     1.697232            58559    1.986266
7153     1.769477            60069    2.046440
7458     1.594972            60397    1.882374
8464     1.744319            63062    1.521871
8622     1.873030            dtype: float64
8815     1.650886
```

```
Variance per movie
MovieID                      MovieID
47       3.977616            8961     2.718769
296      4.542650            27803    3.870312
527      5.300136            30707    4.209014
589      3.682093            30749    4.075091
593      3.761419            32587    3.817907
597      2.153131            33493    3.681488
1183     3.214837            34150    2.585602
1625     4.029719            39183    3.579325
1721     2.085073            44694    2.879386
1923     1.962190            45186    2.002193
2571     2.692453            46062    1.770417
2762     2.235632            48394    3.795599
3793     3.114035            50872    3.328494
4306     1.035163            51662    4.084997
4886     3.064806            53996    2.604734
5669     4.272308            54286    3.467408
5995     4.603146            55820    2.072066
6016     3.957653            56339    3.449864
6377     2.123488            56757    3.915986
6539     2.157970            57274    3.179749
6700     2.880596            58559    3.945251
7153     3.131050            60069    4.187916
7458     2.543935            60397    3.543330
8464     3.042650            63062    2.316092
8622     3.508243            dtype: float64
8815     2.725423
```

Figure 4.12 Standard deviation and variance per movie

4.5.2 RATINGS VERSUS USERS

Figure 4.13 shows calculations and the output derived from the dataset. From the figure, it can be gathered that the dataset contains fifty-eight users, fifty movies, one thousand six hundred and ninety-seven ratings out of the possible two thousand nine hundred ratings and finally the percentage of ratings as 58.52%.

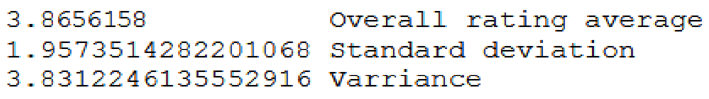


```
58 users and  50 movies found in the dataset
1697 ratings out of possible  2900
58.51724137931035 % ratings
```

Figure 4.13 Users, movies, ratings count and rating percentage

4.5.3 OVERALL RATINGS AVERAGE, STANDARD DEVIATION AND VARIANCE

Other important aspects to consider before making recommendations were ratings average, standard deviation and the variance. As displayed on figure 4.14, the overall ratings average was found to be 3.87, the standard deviation as 1.96 while the variance was found to be 3.83.

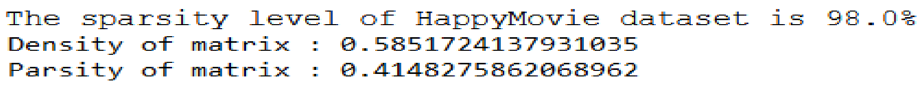


```
3.8656158          Overall rating average
1.9573514282201068 Standard deviation
3.8312246135552916 Varriance
```

Figure 4.14 Overall average, standard deviation and variance

4.5.4 Sparsity, Density And Parsity Of The Rating Matrix

As demonstrated by the output in figure 4.15, the density, sparsity and parsity of the ratings matrix were also calculated and the sparsity was found to be 98.0%, the density of the matrix 0.585 while the parsity of the matrix was found to be 0.415.

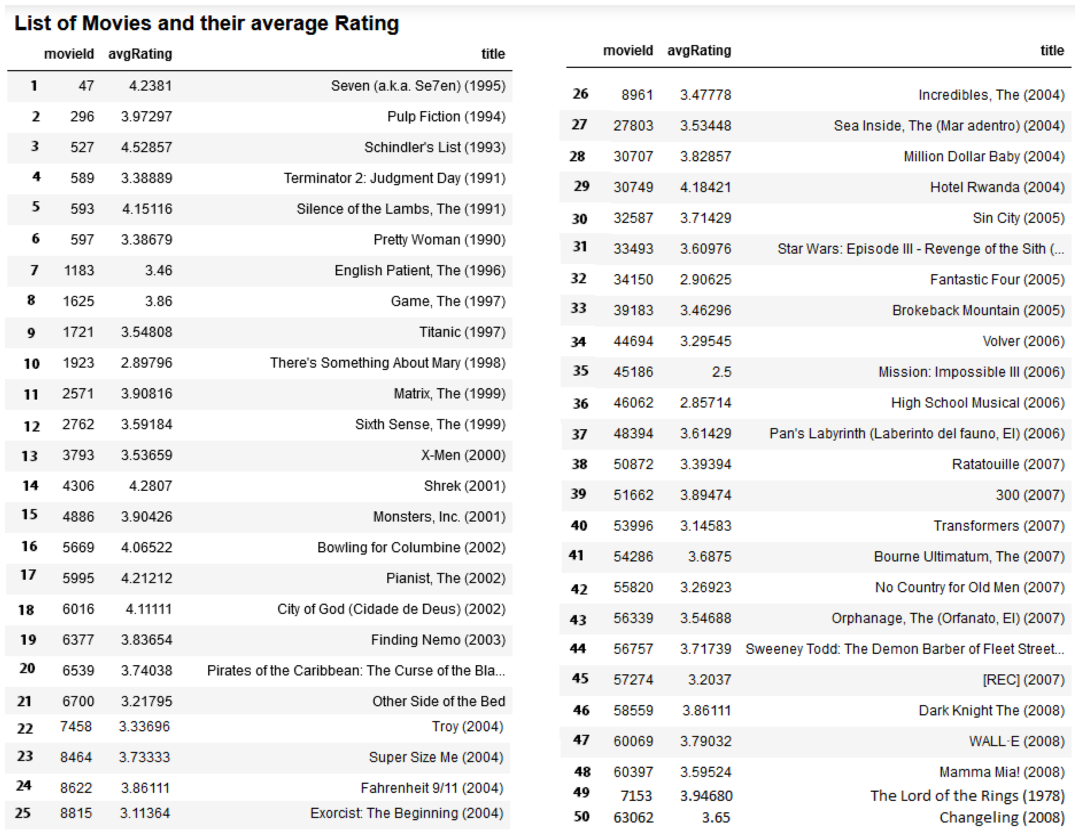


```
The sparsity level of HappyMovie dataset is 98.0%
Density of matrix : 0.585172413793103 5
Parsity of matrix : 0.4148275862068962
```

Figure 4.15 Sparsity, density and parsity level calculation

4.5.5 Movies And Their Average Rating

In addition to the sparsity, density and parsity calculations, an average rating for every movie was calculated with the results shown in figure 4.16. The output shows the movie id, average rating for every movie, movie title and the year of release.



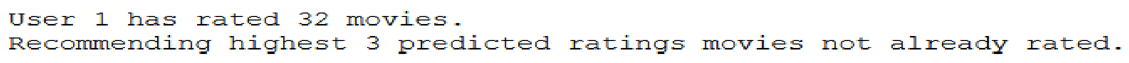
List of Movies and their average Rating

	movieId	avgRating	title
1	47	4.2381	Seven (a.k.a. Se7en) (1995)
2	296	3.97297	Pulp Fiction (1994)
3	527	4.52857	Schindler's List (1993)
4	589	3.38889	Terminator 2: Judgment Day (1991)
5	593	4.15116	Silence of the Lambs, The (1991)
6	597	3.38679	Pretty Woman (1990)
7	1183	3.46	English Patient, The (1996)
8	1625	3.86	Game, The (1997)
9	1721	3.54808	Titanic (1997)
10	1923	2.89796	There's Something About Mary (1998)
11	2571	3.90816	Matrix, The (1999)
12	2762	3.59184	Sixth Sense, The (1999)
13	3793	3.53659	X-Men (2000)
14	4306	4.2807	Shrek (2001)
15	4886	3.90426	Monsters, Inc. (2001)
16	5669	4.06522	Bowling for Columbine (2002)
17	5995	4.21212	Pianist, The (2002)
18	6016	4.11111	City of God (Cidade de Deus) (2002)
19	6377	3.83654	Finding Nemo (2003)
20	6539	3.74038	Pirates of the Caribbean: The Curse of the Bla...
21	6700	3.21795	Other Side of the Bed
22	7458	3.33696	Troy (2004)
23	8464	3.73333	Super Size Me (2004)
24	8622	3.86111	Fahrenheit 9/11 (2004)
25	8815	3.11364	Exorcist: The Beginning (2004)
26	8961	3.47778	Incredibles, The (2004)
27	27803	3.53448	Sea Inside, The (Mar adentro) (2004)
28	30707	3.82857	Million Dollar Baby (2004)
29	30749	4.18421	Hotel Rwanda (2004)
30	32587	3.71429	Sin City (2005)
31	33493	3.60976	Star Wars: Episode III - Revenge of the Sith (...
32	34150	2.90625	Fantastic Four (2005)
33	39183	3.46296	Brokeback Mountain (2005)
34	44694	3.29545	Volver (2006)
35	45186	2.5	Mission: Impossible III (2006)
36	46062	2.85714	High School Musical (2006)
37	48394	3.61429	Pan's Labyrinth (Laberinto del fauno, El) (2006)
38	50872	3.39394	Ratatouille (2007)
39	51662	3.89474	300 (2007)
40	53996	3.14583	Transformers (2007)
41	54286	3.6875	Bourne Ultimatum, The (2007)
42	55820	3.26923	No Country for Old Men (2007)
43	56339	3.54688	Orphanage, The (Orfanato, El) (2007)
44	56757	3.71739	Sweeney Todd: The Demon Barber of Fleet Street...
45	57274	3.2037	[REC] (2007)
46	58559	3.86111	Dark Knight The (2008)
47	60069	3.79032	WALL·E (2008)
48	60397	3.59524	Mamma Mia! (2008)
49	7153	3.94680	The Lord of the Rings (1978)
50	63062	3.65	Changeling (2008)

Figure 4.16 List of movies and their average ratings

4.6 INDIVIDUAL RECOMMENDATIONS AND ANALYSIS

The information obtained from the data frame showed in figure 4.17 was then used to make recommendations for individual users. User 1 was used as the test user as shown by figure 4.19. From the figure, it can be gathered that out of the fifty movies in the dataset, user 1 has already watched (rated) thirty-two movies. That leaves user 1 with only eighteen possible movies for recommendations.

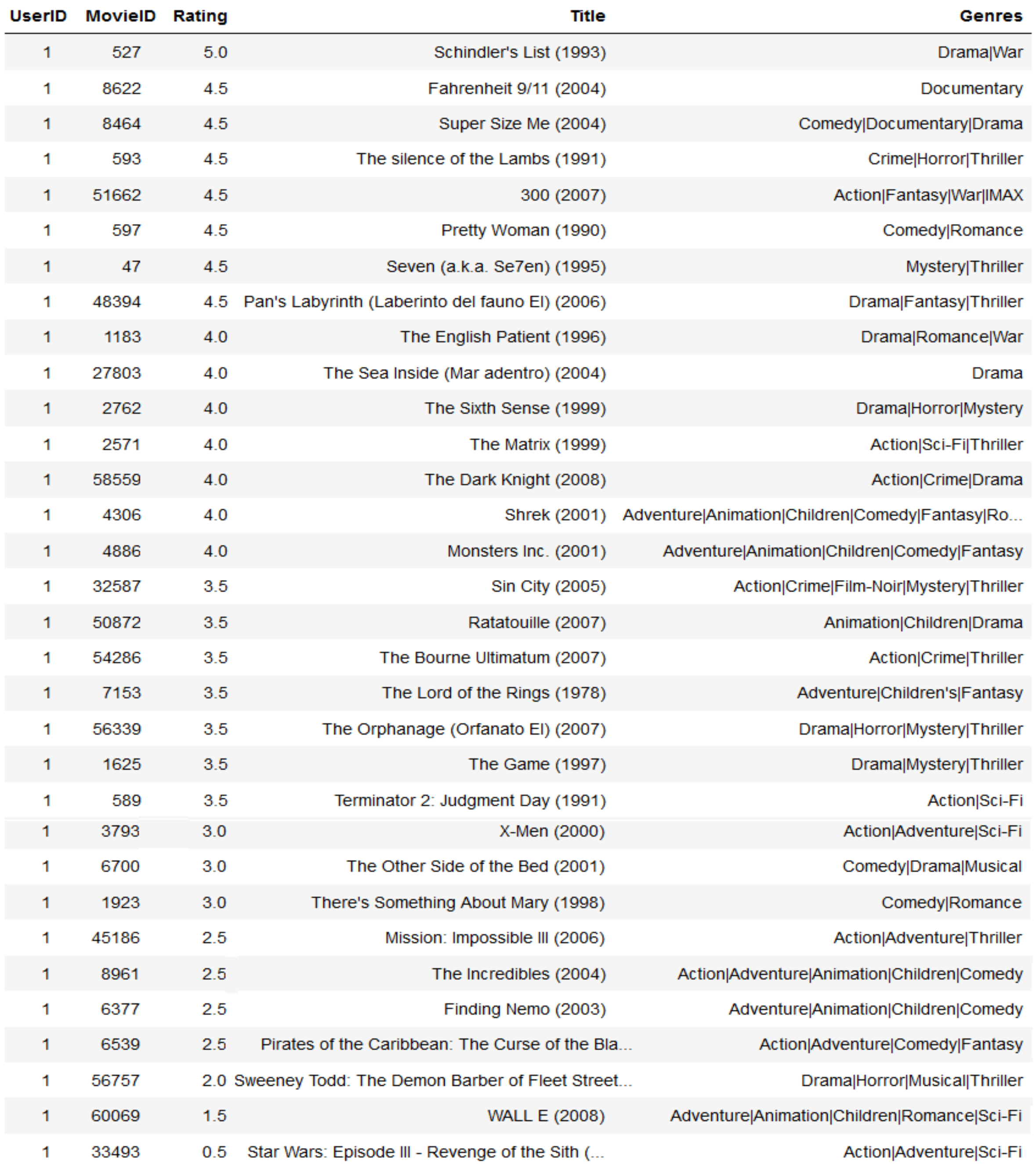


```
User 1 has rated 32 movies.
Recommending highest 3 predicted ratings movies not already rated.
```

Figure 4.17 Selection of test user (user 1)

Already rated for test user (User 1)

The entire list of all the thirty-two movies rated by user 1 was viewed with the results shown by figure 4.18. The figure shows movie ids, rating allocated, title of the movie and the genre of each movie rated.

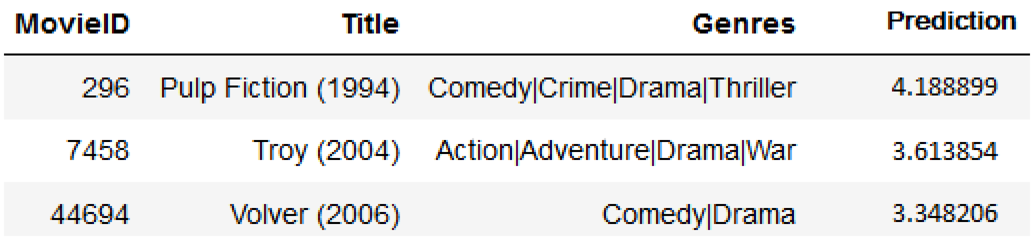


UserID	MovieID	Rating	Title	Genres
1	527	5.0	Schindler's List (1993)	Drama|War
1	8622	4.5	Fahrenheit 9/11 (2004)	Documentary
1	8464	4.5	Super Size Me (2004)	Comedy|Documentary|Drama
1	593	4.5	The silence of the Lambs (1991)	Crime|Horror|Thriller
1	51662	4.5	300 (2007)	Action|Fantasy|War|IMAX
1	597	4.5	Pretty Woman (1990)	Comedy|Romance
1	47	4.5	Seven (a.k.a. Se7en) (1995)	Mystery|Thriller
1	48394	4.5	Pan's Labyrinth (Laberinto del fauno El) (2006)	Drama|Fantasy|Thriller
1	1183	4.0	The English Patient (1996)	Drama|Romance|War
1	27803	4.0	The Sea Inside (Mar adentro) (2004)	Drama
1	2762	4.0	The Sixth Sense (1999)	Drama|Horror|Mystery
1	2571	4.0	The Matrix (1999)	Action|Sci-Fi|Thriller
1	58559	4.0	The Dark Knight (2008)	Action|Crime|Drama
1	4306	4.0	Shrek (2001)	Adventure|Animation|Children|Comedy|Fantasy|Ro...
1	4886	4.0	Monsters Inc. (2001)	Adventure|Animation|Children|Comedy|Fantasy
1	32587	3.5	Sin City (2005)	Action|Crime|Film-Noir|Mystery|Thriller
1	50872	3.5	Ratatouille (2007)	Animation|Children|Drama
1	54286	3.5	The Bourne Ultimatum (2007)	Action|Crime|Thriller
1	7153	3.5	The Lord of the Rings (1978)	Adventure|Children's|Fantasy
1	56339	3.5	The Orphanage (Orfanato El) (2007)	Drama|Horror|Mystery|Thriller
1	1625	3.5	The Game (1997)	Drama|Mystery|Thriller
1	589	3.5	Terminator 2: Judgment Day (1991)	Action|Sci-Fi
1	3793	3.0	X-Men (2000)	Action|Adventure|Sci-Fi
1	6700	3.0	The Other Side of the Bed (2001)	Comedy|Drama|Musical
1	1923	3.0	There's Something About Mary (1998)	Comedy|Romance
1	45186	2.5	Mission: Impossible III (2006)	Action|Adventure|Thriller
1	8961	2.5	The Incredibles (2004)	Action|Adventure|Animation|Children|Comedy
1	6377	2.5	Finding Nemo (2003)	Adventure|Animation|Children|Comedy
1	6539	2.5	Pirates of the Caribbean: The Curse of the Bla...	Action|Adventure|Comedy|Fantasy
1	56757	2.0	Sweeney Todd: The Demon Barber of Fleet Street...	Drama|Horror|Musical|Thriller
1	60069	1.5	WALL E (2008)	Adventure|Animation|Children|Romance|Sci-Fi
1	33493	0.5	Star Wars: Episode III - Revenge of the Sith (...	Action|Adventure|Sci-Fi

Figure 4.18 All movies rated by test user (user 1) and the ratings allocated

Movie recommendation user 1

Ultimately the prototype goes through the entire data frame and recommends the highest three predicated ratings for user 1. All the three recommendations are of movies that user 1 has not rated (watched). The recommendations are displayed in descending order, starting with the highest predicted rating as shown by figure 4.19. The figure shows movie ids, title and genres of the top three movie recommendations for test user 1.

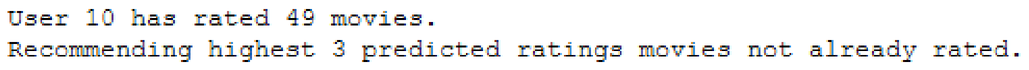


MovieID	Title	Genres	Prediction
296	Pulp Fiction (1994)	Comedy|Crime|Drama|Thriller	4.188899
7458	Troy (2004)	Action|Adventure|Drama|War	3.613854
44694	Volver (2006)	Comedy|Drama	3.348206

Figure 4.19 Test user (user 1) predictions and recommendations

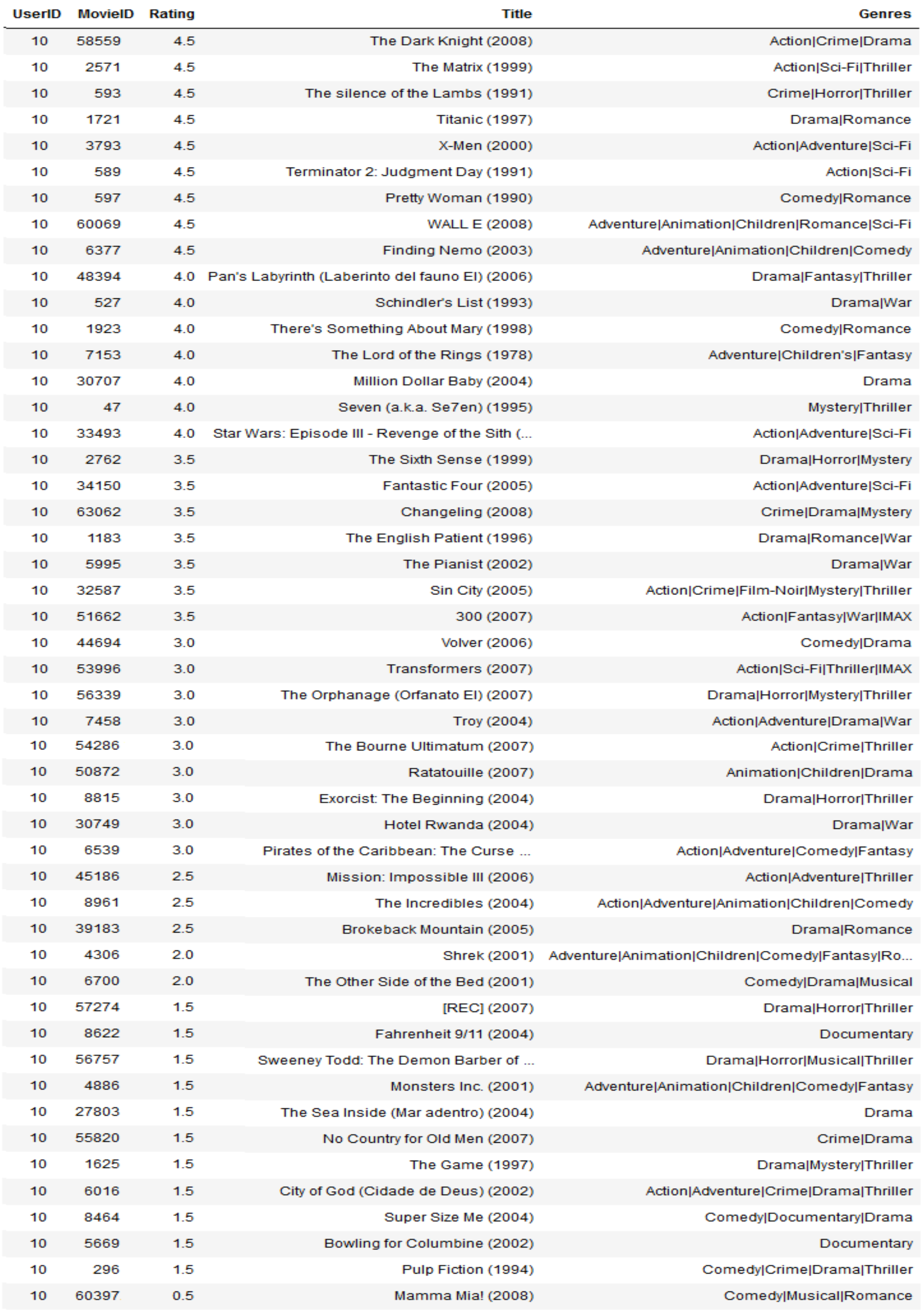
Second test user (user 10)

While user 1 has rated thirty-two movies out of fifty, leaving user 1 with eighteen possible recommendations, user 10 has already rated forty-nine out of fifty movies, leaving user 10 with only one possible recommendation. In this case, the prototype by default recommends that movies regardless of the predicted rating since it is the only movie user 10 has not rated (watched) as shown in figure 4.22. In figure 4.20 the prototype is searching the dataset for user 10 details and the prototype gives a feedback of the number of movies rated by user 10. In addition to this, the recommendation function is called. Figure 4.21 shows all the movies rated by user 10. User id, movie id, movie title, year of release and genres are shown.



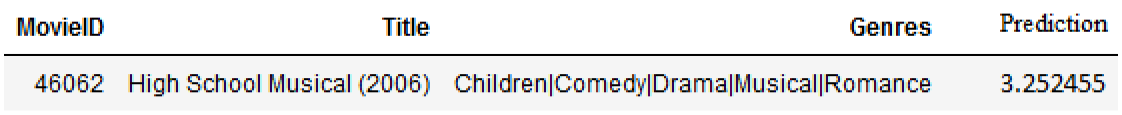
```
User 10 has rated 49 movies.
Recommending highest 3 predicted ratings movies not already rated.
```

Figure 4.20 Fetching user details and displaying the total number of movies rated by user 10



UserID	MovieID	Rating	Title	Genres
10	58559	4.5	The Dark Knight (2008)	Action|Crime|Drama
10	2571	4.5	The Matrix (1999)	Action|Sci-Fi|Thriller
10	593	4.5	The silence of the Lambs (1991)	Crime|Horror|Thriller
10	1721	4.5	Titanic (1997)	Drama|Romance
10	3793	4.5	X-Men (2000)	Action|Adventure|Sci-Fi
10	589	4.5	Terminator 2: Judgment Day (1991)	Action|Sci-Fi
10	597	4.5	Pretty Woman (1990)	Comedy|Romance
10	60069	4.5	WALL E (2008)	Adventure|Animation|Children|Romance|Sci-Fi
10	6377	4.5	Finding Nemo (2003)	Adventure|Animation|Children|Comedy
10	48394	4.0	Pan's Labyrinth (Laberinto del fauno El) (2006)	Drama|Fantasy|Thriller
10	527	4.0	Schindler's List (1993)	Drama|War
10	1923	4.0	There's Something About Mary (1998)	Comedy|Romance
10	7153	4.0	The Lord of the Rings (1978)	Adventure|Children's|Fantasy
10	30707	4.0	Million Dollar Baby (2004)	Drama
10	47	4.0	Seven (a.k.a. Se7en) (1995)	Mystery|Thriller
10	33493	4.0	Star Wars: Episode III - Revenge of the Sith (...	Action|Adventure|Sci-Fi
10	2762	3.5	The Sixth Sense (1999)	Drama|Horror|Mystery
10	34150	3.5	Fantastic Four (2005)	Action|Adventure|Sci-Fi
10	63062	3.5	Changeling (2008)	Crime|Drama|Mystery
10	1183	3.5	The English Patient (1996)	Drama|Romance|War
10	5995	3.5	The Pianist (2002)	Drama|War
10	32587	3.5	Sin City (2005)	Action|Crime|Film-Noir|Mystery|Thriller
10	51662	3.5	300 (2007)	Action|Fantasy|War|IMAX
10	44694	3.0	Volver (2006)	Comedy|Drama
10	53996	3.0	Transformers (2007)	Action|Sci-Fi|Thriller|IMAX
10	56339	3.0	The Orphanage (Orfanato El) (2007)	Drama|Horror|Mystery|Thriller
10	7458	3.0	Troy (2004)	Action|Adventure|Drama|War
10	54286	3.0	The Bourne Ultimatum (2007)	Action|Crime|Thriller
10	50872	3.0	Ratatouille (2007)	Animation|Children|Drama
10	8815	3.0	Exorcist: The Beginning (2004)	Drama|Horror|Thriller
10	30749	3.0	Hotel Rwanda (2004)	Drama|War
10	6539	3.0	Pirates of the Caribbean: The Curse ...	Action|Adventure|Comedy|Fantasy
10	45186	2.5	Mission: Impossible III (2006)	Action|Adventure|Thriller
10	8961	2.5	The Incredibles (2004)	Action|Adventure|Animation|Children|Comedy
10	39183	2.5	Brokeback Mountain (2005)	Drama|Romance
10	4306	2.0	Shrek (2001)	Adventure|Animation|Children|Comedy|Fantasy|Ro...
10	6700	2.0	The Other Side of the Bed (2001)	Comedy|Drama|Musical
10	57274	1.5	[REC] (2007)	Drama|Horror|Thriller
10	8622	1.5	Fahrenheit 9/11 (2004)	Documentary
10	56757	1.5	Sweeney Todd: The Demon Barber of ...	Drama|Horror|Musical|Thriller
10	4886	1.5	Monsters Inc. (2001)	Adventure|Animation|Children|Comedy|Fantasy
10	27803	1.5	The Sea Inside (Mar adentro) (2004)	Drama
10	55820	1.5	No Country for Old Men (2007)	Crime|Drama
10	1625	1.5	The Game (1997)	Drama|Mystery|Thriller
10	6016	1.5	City of God (Cidade de Deus) (2002)	Action|Adventure|Crime|Drama|Thriller
10	8464	1.5	Super Size Me (2004)	Comedy|Documentary|Drama
10	5669	1.5	Bowling for Columbine (2002)	Documentary
10	296	1.5	Pulp Fiction (1994)	Comedy|Crime|Drama|Thriller
10	60397	0.5	Mamma Mia! (2008)	Comedy|Musical|Romance

Figure 4.21 Ratings for all movies watched and rated by user 10

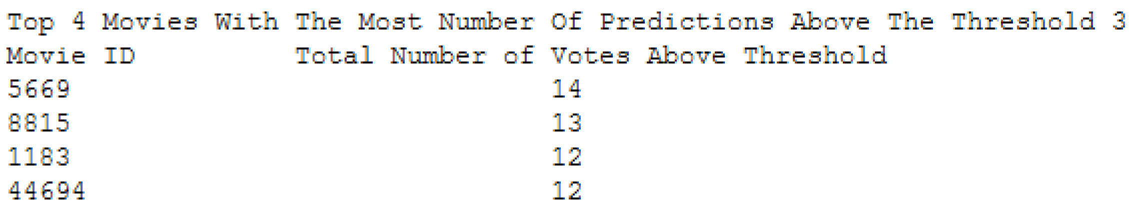


MovieID	Title	Genres	Prediction
46062	High School Musical (2006)	Children|Comedy|Drama|Musical|Romance	3.252455

Figure 4.22 Movie recommendations for user 10

4.7 VIRTUAL GROUP FORMATION

After all the predications and recommendations are made for each individual in the system, the system then went through the plurality check to determine 4 movies with the most recommendations/predictions that are above the predefined threshold which is 3 in our case. The system then displays the movie ID and the total number of votes above the threshold. The list is displayed in discerning order as displayed on figure 4.23, starting with the movie that has the most motes to the movie with the least votes that are above the threshold. 4 virtual groups were then formed.

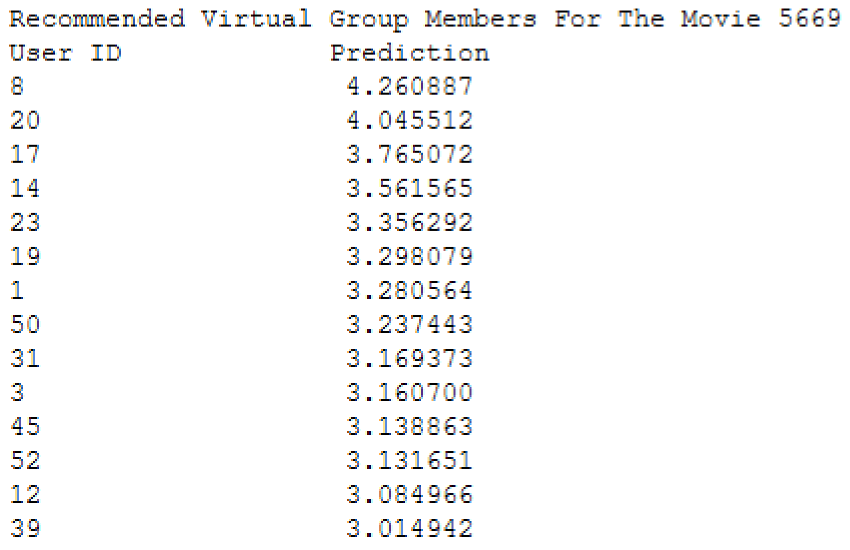


```
Top 4 Movies With The Most Number Of Predictions Above The Threshold 3
Movie ID          Total Number of Votes Above Threshold
5669                             14
8815                             13
1183                             12
44694                            12
```

Figure 4.23 Movies with the greatest number of predictions above threshold

4.8 VIRTUAL GROUP MOVIE RECOMMENDATIONS AND ANALYSIS

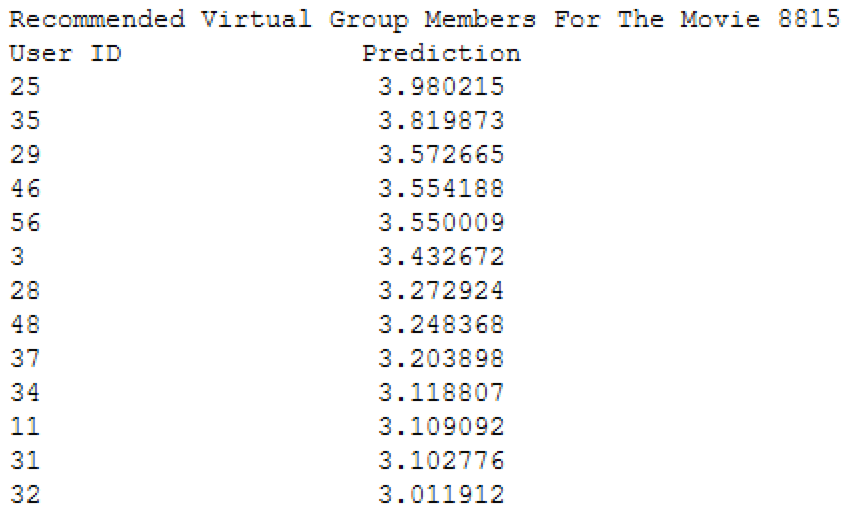
From the 4 virtual groups formed, group movie recommendations were made. The first virtual group was formed of 14 members and the movie 5669 was recommended to this group. As figure 4.24 shows, the list is displayed in descending order starting with the highest prediction of user 8 which is 4.260887 to the last prediction of user 39 with the prediction of 3.014942.



```
Recommended Virtual Group Members For The Movie 5669
User ID              Prediction
8                     4.260887
20                    4.045512
17                    3.765072
14                    3.561565
23                    3.356292
19                    3.298079
1                     3.280564
50                    3.237443
31                    3.169373
3                     3.160700
45                    3.138863
52                    3.131651
12                    3.084966
39                    3.014942
```

Figure 4.24 Recommended virtual group members for the 5669

The second virtual group was formed of 13 members and the movie 5815 was recommended to this group. Just like with the movie 5669 on figure 4.24, figure 4.25 shows, the list is displayed in descending order starting with the highest prediction of user 25 which is 3.980215 to the last prediction of user 32 with the prediction of 3.011912.



```
Recommended Virtual Group Members For The Movie 8815
User ID                Prediction
25                      3.980215
35                      3.819873
29                      3.572665
46                      3.554188
56                      3.550009
3                       3.432672
28                      3.272924
48                      3.248368
37                      3.203898
34                      3.118807
11                      3.109092
31                      3.102776
32                      3.011912
```

Figure 4.25 Recommended virtual group members for the movie 8815

The third virtual group was formed of 12 members and the movie 1183 was recommended to this group. Figure 4.26 shows, the list displayed in descending order starting with the highest prediction of user 31 which is 4.132451 to the last prediction of user 3 with the prediction of 3.103615.

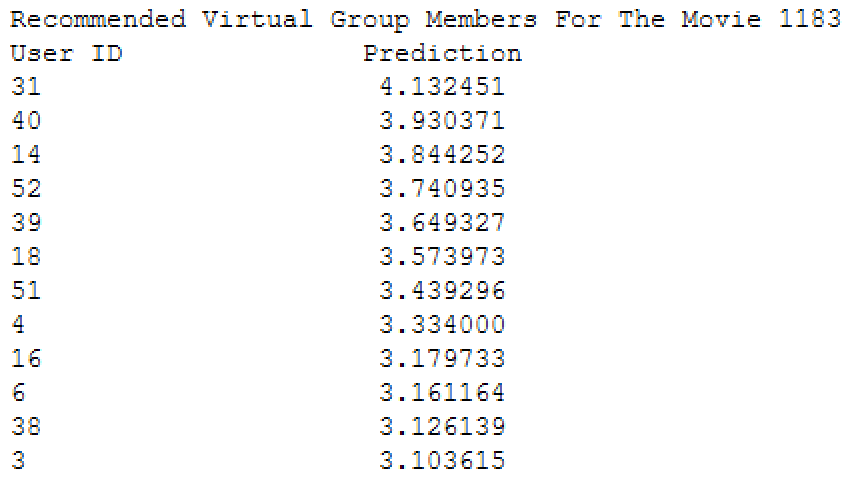

```
Recommended Virtual Group Members For The Movie 1183
User ID                 Prediction
31                        4.132451
40                        3.930371
14                        3.844252
52                        3.740935
39                        3.649327
18                        3.573973
51                        3.439296
4                         3.334000
16                        3.179733
6                         3.161164
38                        3.126139
3                         3.103615
```

Figure 4.26 Recommended virtual group members for the 1183

The fourth virtual group was formed of 12 members as well, and the movie 44694 was recommended to this group. Figure 4.27 shows, the list displayed in descending order starting with the highest prediction of user 37 which is 4.696413 to the last prediction of user 20 with the prediction of 3.018649.

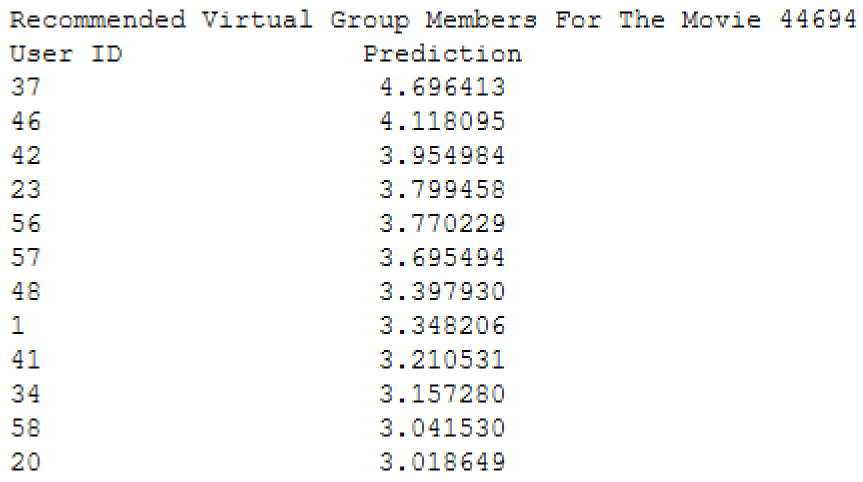
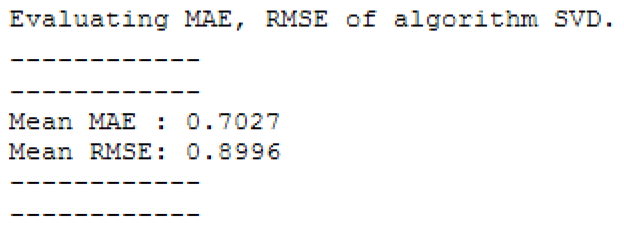

```
Recommended Virtual Group Members For The Movie 44694
User ID                 Prediction
37                        4.696413
46                        4.118095
42                        3.954984
23                        3.799458
56                        3.770229
57                        3.695494
48                        3.397930
1                         3.348206
41                        3.210531
34                        3.157280
58                        3.041530
20                        3.018649
```

Figure 4.27 Recommended virtual group members for the 44694

4.9 EVALUATION OF THE PROTOTYPE

Another crucial stage of the prototype was to evaluate its accuracy by comparing the predicted ratings directly with the actual ratings given by the users. To fulfil this purpose, mean absolute error (MAE) and root mean squared error (RMSE) were deployed with the results shown in figure 4.28. (Sarif & Ziad, 2016) Proposed a movie recommendation system using movielens dataset and achieved 0.709531 MAE and 0.905520 RMSE using FunkSVD while achieving 0.717344 MAE and 0.9200979 RMSE using item-based collaborative filtering. (Rahul & Om, 2016) Proposed a movie recommendation system via K-Means PSO-

FCM technique that achieved 0.7547 as the MAE. (Kleeman, et al., 2018) proposed a system that achieved 0.82 MAE and 1.08 RMSE using Fast Maximum Margin Matrix Factorization, 0.80 MAE and 1.05 RMSE using Iterative SVD and 0.72 MAE and 0.95 RMSE using Repeated Matrix. Our prototype showed a better performance over the all these with 0.7027 MAE and 0.8996 RMSE as figure 4.30 illustrates.



```
Evaluating MAE, RMSE of algorithm SVD.
------------
------------
Mean MAE : 0.7027
Mean RMSE: 0.8996
------------
------------
```

Figure 4.28 MAE and RMSE results of the prototype

4.10 Overall Analysis Of The Results

The proposed collaborative filtering virtual group movie recommendation system using social network information has demonstrated a good 0.70 MAE and 0.89 RMSE. The algorithm has been explored and evaluated comprehensively. The findings depicted that the prototype fulfils its objectives and performs better than other recommendation systems considered for comparison.

4.11 Chapter Summary

This chapter presented the experimental results of the application of collaborative filtering using the matrix factorization technique. The chapter initially discusses the programming environment deployed in the study, python libraries used, overview of the dataset, data exploration and some basic calculations carried out. Thereafter, individual movie recommendations were made, individual movie recommendation results discussed and the results evaluated using mean absolute error (MAE) and root mean squared error (RMSE). We further tested the prototype using 2 test users (User 1 and User 10) and found appropriate recommendations for them in the process.

Finally, the chapter discussed how the virtual group was formed and movies were recommended to this virtual group. An analysis of the results obtained from group movie recommendations is given. It also discusses how the prototype went through the plurality check to ensure that movies with the most number of predictions above the threshold were recommended to respective groups.

The next chapter concludes the study and chips in on future work.

5 CONCLUSION AND FUTURE WORK

5.1 INTRODUCTION

This final chapter of the research project provides conclusion, summary of the study, limitations, evaluation of whether the objective was met and future recommendations in this area of study. Chapter one of the dissertation stated the primary research question as "How can a group movie recommendation system be developed using social network information?" In order to answer this primary question, the following were identified as secondary questions: "What group movie recommendation algorithms have been used in literature?", "How can the system be developed in a way that it meets the needs of individuals and groups?" and "How will the effectiveness and efficiency of the developed system be measured?"

So as to address the primary questions, the following were stated as the objectives of the research project:" Investigate group recommendation systems algorithms in literature", "Propose a new algorithm that can optimize the group movie recommendation systems using social network data", "Develop a prototype for a group movie recommendation system" and "Evaluate the prototype the effectiveness and efficiency of the developed prototype."

This chapter evaluates whether this objectives of the research project has been met and if the research questions have been answered.

5.2 SUMMARY OF THE STUDY

An extensive literature review, both individual and group recommendations, was carried out in the second chapter of the dissertation. Different filtering techniques, content based, collaborative and hybrid were explored together with their strength and limitations. The study further highlighted challenges faced by each of the filtering techniques and how to address those challenges. In so doing, focus was given to memory based and model based filtering techniques. In addition, the study went deeper on the matrix factorization, of model based filtering. Online, offline and user studies evaluation approaches were amongst the items discussed in the study. Various accuracy metrics in recommendation systems were also discussed.

After this exploration, a decision was made to use model based, matrix factorization technique of collaborative filtering because of its strengths, specifically its proven reasonably good recommendations.

5.3 LIMITATIONS, RECOMMENDATIONS AND FUTURE WORK

It is evident that the prototype produced some desirable results. However, the proposed recommendation system presents some limitations as well. Below are the limitations presented by the prototype, recommendations and future work:

1- The dataset contained only fifty-eight users and fifty movies. This resulted in the prototype recommending a movie that has a low predicted rating because it was the only movie that user has not rated. Future research could utilize a bigger dataset of movies so as to have a broader selection of movies. This on its own would enable the prototype to recommend movies with higher predicted ratings for individuals and ultimately an increased satisfaction for virtual group members.

2- The system generates four virtual groups. Future research could focus on generating recommendations for varied number of groups of varied sizes each.

5.4 CONCLUSION

This research project successfully managed to cover all the major areas of recommendation systems. These areas include filtering techniques, strengths and limitations of each filtering technique, similarity computation in recommendation system, evaluation approaches, accuracy metrics and prediction computation. A prototype was then developed to generate predictions using python, Jupyter notebook integrated development environment (IDE) because of its technical advantages over other programming languages and its rich ecosystem for scientific inquiry in the form of many proven, and popular open-source packages like SciPy, NumPy, Matplotlib and Seaborn. HappyMovie dataset, consisting of fifty-eight users (Facebook friends) and fifty movies was used. Individual movie predictions and recommendations were made, after which four virtual were formed based on the plurality voting algorithm. The accuracy of the prototype in making predictions was evaluated using mean absolute error (MAE) and root mean squared error (RMSE) which achieved a good mean MAE of 0.7027 and mean RMSE of 0.8996 as shown by figure 4.28 in chapter 4. This performance measurement reveals that the methods applied yielded a very satisfactory result. The system was also able to recommend only movies with predictions above the pre-set threshold, 3. Movies above the threshold were recommended not only to individuals, but also to the virtual groups that the prototype generated. Plurality check was also achieved as shown by table 3.1 which is also in chapter 4 of the dissertation. Standard ranking technique was applied both on movies recommended to individuals and to virtual groups to display the movies in descending order.

6 REFERENCES

O'Sullivan, D. et al., 2004. Improving the Quality of the Personalized Electronic Program Guide. *User-Modeling and User-Adapted Interaction.*

Abhyankar, A., 2011. Social Networking Sites. *College Research Journal,* pp. 18-21.

Acquisti, A. & Gross, R., 2006. *Imagined communities: Awareness, information sharing, and privacy on the Facebook.* Cambridge, Robinson College, p. 36– 58.

Adomavicius, G. & Tuzhilin, A., 2005. Toward the Next Generation of Recommender. *A Survey of the State-of-the-Art and Possible Extensions,* pp. 734-749.

Adomavicius, G. & Tuzhilin, A., 2005. Towards The Next Generation of Recommender System: A Survey of The State-of-The-Art And Possible Extensions. *IEEE Transactions On Knowledge And Data Engineering,* 17(6), pp. 734-749.

Aggarwal, C. C., 2016. *Recommender Systems.* New York: Springer.

Akhil, V. & Shelbi, J., 2017. A SURVEY OF RECOMMENDER SYSTEM TYPES AND ITS CLASSIFICATION. *International Journal of Advanced Research in Computer Science,* 8(9), pp. 486-491.

Al-Barznji, K. & Atanassov, A., 2018. Comparison of Memory Based Filtering Techniques For Generating Recommendations On Large Data. *Engineering And Automation.*

Al-Barznjl, K. & Atanassov, A., 2017. *Collaborative Filtering Techniques For Generating Recommendations On Big Data.* Sofia, Bulgaria, s.n.

Al-Bashiri, H., Abdulgabber, M. A., Romli, A. & Kahtan, H., 2018. *An Improved Memory-Based Collaborative Filtering Method Based on The TOPSIS.* s.l., s.n.

Aleksandrova, M., 2017. *Matrix Factorization And Contrast AnalysisTechniques For Recommendation.* s.l., s.n.

Anand, S. S. & Mobasher, B., 2003. *Intelligent Techniques for Web Personalization.* s.l., Springer , pp. 1-36.

Anderson , j. & Bernoff, J., 2010. *Millenials will make online sharing in networks a lifelong habit.* Washington DC: Pew Research Center.

Ansgar, K. et al., 2015. Ethics of Personalized Information Filtering. In: *IAA: Incentive-Based Anonymous Authentication Scheme in Smart Grids* . s.l.:s.n., pp. 123-132.

Ardissono, L. et al., 2003. Intrigue: Personalized recommendation of tourist attractions for desktop and hand held devices. *APPLIED ARTIFICIAL INTELLIGENCE: Special Issue on Artificial Intelligence for Cultural Heritage and Digital,* 17(8), pp. 687-714.

Ardissono, L. et al., 2003. Intrigue: Personalized recommendation of tourist attractions for desktop and hand held devices. *Applied Artificial Intelligence,* pp. 687-714.

Asabere, N. Y., 2013. Towards a Viewpoint of Context-Aware Recommender Systems (CARS) and Services. *International Journal of Computer Science and Telecommunications*, pp. 19-29.

Bahadorpour, M., Neysiani, B. S. & Shahraki, M. N., 2017. Determining Optimal Number of Neighbors In Item-Based kNN CollaborativeFiltering Algorithm For Learning Preferences of New Users. *Journal of Telecommunications,* 9(3), pp. 163-167.

Balabanovic, M. & Shoham, Y., 1997. Fab: Content-Based, Collaborative Recommendation. *Communications of the ACM,* Volume 40, pp. 66-72.

Baltrunas, L., Makcinskas, T. & Ricci, F., 2010. *Group recommendations with rank aggregation and collaborative filtering.* Barcelona, USA ©2010.

Baltrunas, L., Makcinskas, T. & Ricci, F., 2010. *Group recommendations with rank aggregation and collaborative filtering..* New York, NY, USA (2010), ACM, p. 119–126.

Basu, C., Hirsh, H. & Cohen, W., 1998. Recommendation as Classification: Using Social and Content-Based Information in Recommendation. *Proceedings of AAAI-98.*

Belkin, N. J. & Croft, W. B., 1992. *Information Filtering And Information Retrieval:Two Sides of The Same coin?.* s.l., s.n., pp. 29-38.

Berkovsky, S. & Freyne, J., 2010. *Group-based recipe recommendations: analysis of data aggregation strategies.* New York, NY, USA, ACM, p. 111–118.

Berry, M. W., Dumais, S. T. & O'Brien, G. W., 1995. Using linear algebra for intelligent information retrieval. In: *SIAM Review.* s.l.:s.n., pp. 573-595.

Bobadilla, J., Ortega, F., Hernando, A. & Gutiérrez, A., 2013. *Recommender systems survey.* s.l., s.n., pp. 109-132.

Boyd, D. M. & Ellison, N. B., 2007. Social Network Sites: Definition, History, and Scholarship. *Journal of Computer-Mediated Communication,* pp. 210-230.

Boyd, D. M. & Ellison, N. B., 2008. Journal of Computer-Mediated Communication. *Social Network Sites: Definition, History, and Scholarship,* 1(1), pp. 210-230.

Brand, M. E., 2003. *Incremental Singular Value Decomposition of Incomplete Data.* s.l., s.n.

Breese, J., Heckerma, D. & Kadie, C., 1998. *Empirical analysis of predictive algorithms for collaborative filtering.* San Francisco, CA, Morgan Kaufmann Publishers Inc, p. 43–52.

Breese, J., Heckerman, D. & Kadie, C., 1998 . *Empirical analysis of predictive algorithms for collaborative filtering.* Madison, Wisconsin, Morgan Kaufmann Publishers Inc..

Breese, J. S., Heckerman, D. & Kadie, C., 1998. *Empirical Analysis of Predictive Algorithms for Collaborative Filtering.* s.l., s.n.

Brusilovsky, P., Kobsa, A. & Nejdl, W., 2007. *The Adaptive Web - Methods and Strategies of Web Personalization*, Berlin/Heidelberg: Springer .

Burke, C., 2007. Hybrid Web Recommender Systems. In: *The Adaptive Web* . Berlin, Heidelberg: Springer, pp. 377-408.

Burke, R., 1997. Knowledge-based recommender systems. *Encyclopedia of Library and Information Science,* 1(1), pp. 1-21.

Burke, R., 2002. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction,* p. 331–370.

Burke, R., 2007. The Adaptive Web. In: S. B. Heidelberg, ed. *Hybrid web recommender systems..* Berlin: Springer Link, pp. 377-408.

Burke, R. & Felfernig, A., 2011. Recommender Systems: An Overview. *Ai Magazine* , pp. 13-18.

Cambridge dictionary, 2018. Cambridge: © Cambridge University Press 2019.

Castro-Herrera, C., Cleland-Huang, J. & Mobasher , B., 2009. *A recommender system for dynamically evolving online forums.* New York, ACM, pp. 213-216.

Cheetham, W. & Price, J., 2004. *Measures of Solution Accuracy in Case-Based Reasoning Systems.* Madrid, Spain, ECCBR 2004, pp. 106-118.

Chen, M. & Liu, P., 2017. Performance Evaluation of Recommender Systems. *International Journal of Performability Engineering,* pp. 1246-1256.

Chung, Y., Kim, N.-r., Park, C.-y. & Lee, J.-H., 2018. Improved Neighborhood Search For Collaborative Filtering. *International Journal Of Fuzzy Logic And Intelligent Systems,* 18(1), pp. 29-40.

Cremonesi, P., Turrin, R., Lentini, E. & Matteucci, M., 2008. *An EvaluationMethodologyforCollaborativeRecommenderSystems.* Florence, IEEE, pp. 224-231.

Dara, S., Chowdary, R. & Kumar, C., 2019. A Survey on Group Recommender Systems. *Journal of Intelligent Information Systems,* pp. 1-25.

Deerwester, S. et al., 1990. Indexing by latent semantic analysis.. *Journal of the American Society for information Science,* 41(6), pp. 391-407.

Domingos, P. & Pazzani, M., 1997. On the Optimality of the Simple Bayesian Classifier under Zero-One Loss. *Machine Learning,* p. 103–130.

Dooms, S., De Pessemier, T. & Martens, L., 2013. *MovieTweetings: a Movie Rating Dataset Collected From Twitter.* [Online]
Available at: https://www.researchgate.net/publication/258446476_MovieTweetings_a_Movie_Rating_Dataset_Collected_From_Twitter
[Accessed 1 April 2018].

Do, T., Phung, M. & Nguyen, V., 2010. *Model-based approach for Collaborative Filtering.* Ho Chi Minh city, Vietnam, Academic Network of Loc Nguyen, pp. 217-228.

Ekstrand, M. D., Riedl, J. T. & Konstan, J. A., 2011. *Collaborative Filtering Recommender Systems.* Boston, USA: now Publishers Inc..

Ekstrand, M. D., Riedl, J. T. & Konstan, J. A., 2011. Collaborative Filtering Recommender Systems.. *Found. Trends Hum.-Comput. Interact,* 4(2), pp. 81-173.

Espindola, R. P. & Ebecken, N. F., 2005. On Extending F-Measure And G-Mean Metrics To Multi-Class Problems. *WIT Transactions On Information And Communication Technology,* pp. 1743-3517.

Fleder, D. M. & Hosanagar, K., 2009. Blockbuster Culture's Next Rise or Fall: The Impact of Recommender Systems on Sales Diversity. *Management Science,* pp. 697-712.

Friendly, M., 2009. *Milestones in the history of thematic cartography, statistical graphics, and data visualization,* Truman: s.n.

Funk , S., 2006. *Netflix.* [Online]
Available at: http://sifter.org/˜simon/journal/20061211.html

Funk, S., 2006. [Online]
Available at: http://sifter.org/simon/journal/20061211.html

Godin, R., Zaier, Z. & Faucher, L., 2008. *Evaluating Recommender Systems.* s.l., s.n., pp. 1-42.

Gorrell, G., 2006. *Generalized Hebbian algorithm for incremental singular value decomposition in natural language processing..* s.l.:EACL.

Gunawardana, A. & Shani, G., 2009. A Survey of Accuracy Evaluation Mtrics of Recommendation Tasks. *Journal of Machine Learning Research,* pp. 2935-2962.

Guo, Z., 2017. Curbing Crowdturfing in Online Social Networks. *Fine-Grained Recommendation Mechanism to Curb Astroturfing in Crowdsourcing Systems,* Volume: 5(1), pp. 15529 - 15541.

Hameed, A. M., Jordan, O. A. & Ramachandram, S., 2012. Collaborative Filtering Based Recommendation System: A survey. *International Journal on Computer Science and Engineering (IJCSE),* pp. 859-876.

Han, J. & Kamber, M., 2006. *Data Mining: Concepts and Techniques.* 2nd Edition ed. Francisco, CA: Morgan Kaufmann Publishers.

Hastie, T., Tibshirani, T. & Friedman, R., 2009. The elements of statistical learning. In: *Unsupervised learning.* s.l.:Springer.

Haythornthwaite , C., 2011. Social networks and Internet connectivity effects. *Full Terms & Conditions of access and use can be found at https://www.tandfonline.com/action/journalInformation?journalCode=rics20 Information, Community & Society* , pp. 125-147.

Herlocker, J. L., Konstan, J. A. & Riedl, J., 2000. *Explaining collaborative filtering recommendations.* New York, NY, USA, ACM, p. 241–250.

Hogg, M. A. & Jennings, N. R., 1999. *Social Order in Multiagent Systems.* s.l.:Springer.

Isinkaye, F. O., Folajimi, Y. O. & Ojokoh, B. A., 2015. Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal,* Volume 16, pp. 261-273.

Jadhav, S. D. & Channe, H. P., 2016. Efficient Recommendation System Using Decision Tree Classifier And Collaborative Filtering. *International Research Journal of Engineering and Technology,* 3(8), pp. 2114-2118.

Jameson, A. & Smyth, B., 2017. Recommendation to Groups. In: S. Nature, ed. *The Adaptive Web.* Dublin: © 2017 Springer International Publishing AG, pp. 596-627.

Jannach, D., Zanker, M., Felfernig, A. & Friedrich, G., 2011. *Recommender Systems, An Introduction.* s.l.:Cambridge University Press.

Jeyasekar, A., Akshay, K. & Karan, 2016. Collaborative Filtering Using Euclidean Distance In Recommendation Engine. *Indian Journal of Science And Technology,* 9(37).

Johnson, S. D. et al., 2002. Team development and group processes of virtual learning teams. *Computers & Education,* p. 379–393.

Jung , Y. G., KANG, M. S. & HEO, J., 2014. Clustering performance comparison using K-means and expectation maximization algorithms. *Biotechnology & Biotechnological Equipment,* pp. 44-48.

Kardan, A. A. & Ebrahimi, M., 2013. A novel approach to hybrid recommendation systems based on association rules mining for content recommendation in asynchronous discussion groups. *Information Sciences,* pp. 93-110.

Kavzoglu, T. & Mather, P. M., 2003. The use of backpropagating artificial neural networks in land cover classification.. *International journal of remote sensing,* pp. 4907-4938.

Keenan, T., 2019. *Upwork Global Inc.* [Online]
Available at: https://www.upwork.com/hiring/data/what-is-content-based-filtering/

Keenan, T., 2019. *Upwork Global Inc.* [Online]
Available at: https://www.upwork.com/hiring/data/how-collaborative-filtering-works/

Kim, B., Park, C., Kim, S. & Kim, J., 2006. A new approach for combining content-based and collaborative filters. *Intell Inf Syst* , 27(1), p. 79–91.

Kim, H.-N., Ji, A.-T., Ha, I. & Jo, G.-S., 2010. Collaborative Filtering Based On Collaborative Tagging For Enhancing The Quality of Recommendation. *Electronic Commerce Research And Applications,* 9(1), pp. 73-83.

Kim, J. K., Kim, K. H., Oh, Y. H. & Ryu, Y. U., 2010. A group recommendation system for online communities. *International Journal of Information Management: The Journal for Information Professionals,* 30(3).

Klamler, C., 2003. *A comparison of the Dodgson method and the Copeland rule,* s.l.: s.n.

Kleeman, A., Hendersen, N. & Denuit, S., 2018. *Matrix factorization for collaborative prediction.* [Online]
Available at: http://cs229.stanford.edu/proj2006/KleemanDenuitHenderson-MatrixFactorizationForCollaborativePrediction.pdf

Kohavi, R., 1995. *A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection.* California, USA, IJCAI, pp. 1-7.

Koren, Y., Bell , R. & Volinsky, C., 2009. *MATRIX FACTORIZATION TECHNIQUES FOR RECOMMENDER SYSTEMS,* s.l.: IEEE Computer Society.

Kurucz, M., Bencz´ur, A. A. & Csalog´any, A., 2007. Methods for large scale SVD with missing values.. *KDD Cup and Workshop*.

KURUCZ, M., BENCZ, A. A. & CSALOG´ANY, K., 2007. Methods for large scale SVD with missing values. *KDD Cup and Workshop*.

Lee, J., Sun, M. & Lebanon, G., 2012. *A Comparative Study of Collaborative Filtering Algorithms,* s.l.: arXiv:1205.3193v1 [cs.IR].

Levenson, A. R. & Cohen, S. G., 2002. *Meeting the performance challenge:Calculating RIO for virtual Teams,* Los Angeles: Center for Effective Organizations.

Liang, J., Hu, J., Dong, S. & Honavar, V., 2018. *Top-N-Rank: A Scalable List-Wise Ranking Method for Recommender Systems.* s.l., s.n.

Lieberman, H., van Dyke, N. & Vivacqua, A., 1999. Let's browse: a collaborative browsing agent. *Knowledge-Based Systems,* 12(8), pp. 427-431.

Lipnack, J. & Stamps, S., 1997. *Virtual Teams: Reaching Across Space, Time, and Organizations with Technology.* New Jersey: John Wiley & Sons, Inc.

Liu, L., Mehandjiev, N. & Xu, I., 2011. *Using Contextual Information for Service Recommendation.* Hawaii, s.n.

Liu, R. D., Chou, C. Y., Chung, C. Y. & Liao, H. Y., 2018. Recommender system based on social influence and the virtual house bandwagon effect in virtual worlds. *KYBERNETES,* 47(3), pp. 587-604.

Lops, P., de, M. G. & Semeraro, G., 2010. Recommender Systems Handbook. In: *Content-based Recommender Systems: State of the Art and Trends.* Boston: Springer, pp. 73-105.

Lops, P., Gemmis, M. & Semeraro, G., 2010. Recommender Systems Handbook. In: S. Nature, ed. *Content-based Recommender Systems: State of the Art and Trends.* s.l.:© 2017 Springer International Publishing AG, pp. 73-105.

Lü, L. et al., 2012. Recommender systems. *Physics Reports,* pp. 1-49.

Madhukar, M., 2014. Challenges & Limitation in Recommender Systems. *International Journal of Latest Trends in Engineering and Technology (IJLTET),* pp. 138-142.

Martínez, A. B. et al., 2010. A hybrid content-based and item-based collaborative filtering approach to recommend TV programs enhanced with singular value decomposition. *Information Sciences,* 180(22), pp. 4290-4311.

Masthoff, J., 2002. *Modeling a group of television viewers.* s.l., s.n., pp. 34-42.

Masthoff, J., 2004. Group Modeling: Selecting a Sequence of Television Items to Suit a Group of Viewers. *User Modeling and User-Adapted Interaction.*

Masthoff, J., 2004. Group modeling: Selecting a sequence of television items to suit a group of viewers.. In: *User Modeling and User-Adapted Interaction.* s.l.:s.n., p. 37–85.

Masthoff, J., 2004. Group Recommender Systems:Combining individual models. *User Modeling and User-Adapted Interaction,* 14(1), p. 37–85.

Masthoff, J., 2011. Group recommender systems: Combining individual models. In: *Recommender Systems Handbook.* USA: Springer US, p. 677–702.

Mccallum, A. & Nigam, K., 1998. *A Comparison of Event Models for Naive Bayes Text Classification.* s.l., s.n., pp. 41-48.

McCarthy, J. F. & Anagnost, T. D., 1998. *MUSICFX: An Arbiter of Group Preferences for.* Washington, Association for Computing Machinery. Copyright © 2018 ACM, Inc..

McCarthy, J. F. & Anagnost, T. D., 1998. *Musicfx: an arbiter of group preferences for computer supported collaborative workouts.* New York, NY, USA , ACM, p. 363–372.

McCarthy, J. F. & Anagnost, T. D., 1998. *MusicFX: an arbiter of group preferences for computer supported collaborative workouts.* Seattle, Washington, USA , s.n., pp. 363-372.

Merriam-Webster’s collegiate dictionary, 2014. Springfield: MA: Merriam-Webster Incorporated.

Miller, B. N., Konstan, J. A. & Riedl, J., 2004. PocketLens: Toward a personal recommender system.. *ACM Transactions on Information Systems,* 22(3), pp. 437- 476.

Mobasher, B., Burke, R., Bhaumik, R. & Williams, C., 2007. Toward Trustworthy Recommender Systems:An Analysis of Attack Models and Algorithm Robustness. *ACM Transactions on Internet Technology,* pp. 23-38.

Mobasher, B., Burke, R., Runa, B. & Chad, W., 2005. *Effective attack models for shilling item-based collaborative filtering systems.* Chicago, Illinois, USA, s.n.

Mobasher, B., Dai, H., Luo, T. & Nakagawa, M., 2001. *Improving the Effectiveness of Collaborative Filtering on Anonymous Web Usage Data.* s.l.:s.n.

Montaner, M., López, B. & Rosa, J., 2003. A Taxonomy of Recommender Agents on the Internet. *Artificial Intelligence Review,* pp. 285-330.

Mooney, R. J. & Roy, L., 2000. *Content-based book recommending using learning for text categorization.* San Antonio, Texas, USA, ACM , pp. 195-204.

Mustafa, N., Osman, A., Ahmed, A. & Abdullah, A., 2017. *Collaborative filtering: Techniques and applications.* s.l., DOI: 10.1109/ICCCCEE.2017.7867668.

Nagpal, D., Kaur, S., Gujral, S. & Singh, A., 2015. *FR: A Recommender for Finding Faculty Based On CF Technique.* s.l., s.n.

Najafi, S. & Salam, Z., 2016. *Evaluating Prediction Accuracy for for Collaborative Filtering Algorithms in Recommender Systems,* Stockholm, Sweden: CSC, KTH.

Nakamura, A. & Abe, N., 1998. *Collaborative filtering using weighted majority prediction algorithms..* San Francisco, CA, USA, Morgan Kaufmann Publishers Inc., pp. 395-403.

O’Connor, M., Cosley, D., Konstan, J. A. & Riedl, J., 2001. *Polylens: a recommender system for groups of users.* Norwell, MA, USA , ECSCW’01, p. 199–218.

O’Connor, M., Cosley, D., Konstan, J. A. & Riedl, J., 2001. *PolyLens: A Recommender System for Groups of Users.* Dordrecht: Springer.

O’Connor, M., Cosley, D., Konstan, J. & Riedl, J., 2001. *PolyLens: A Recommender System for Groups of Users.* Bonn, © 2001 Kluwer Academic Publishers, pp. 199-218.

Oxford English Dictionary, 2019. Oxford: Oxford University Press.

Park, D. C. et al., 1991. Electric Load Forecasting Using Artificial Neural Network. *IEEE Transactions on Power Systems,* 6(2), pp. 442-449.

Park, S.-T. & Chu, W., 2009. *Pairwise Preference Regression for Cold-start Recommendation.* New York, ACM, pp. 21-28.

Pazzani, M., 2000. *A Framework for Collaborative, Content-Based and Demographic Filtering.* Netherlands, Kluwer Academic Publishers, p. 393–408.

Pazzani, M. & Billsus, D., n.d. Content-Based Recommendation Systems. In: S. Nature, ed. *The Adaptive Web.* New Brunswick: © 2017 Springer International Publishing AG, pp. 325-341.

Pazzani, M. J. & Billsus, D., 1998. *Learning Collaborative Information Filters.* San Francisco, CA, USA, Morgan Kaufmann Publishers Inc, pp. 46-54.

Pera, M. & Ng, K. Y., 2012. A group recommender for movies based on content similarity and popularity. *Information Processing & Management,* 49(3), pp. 673-687.

Porcel, C., Tejeda-Lorente, A., Martínez, M. A. & Herrera-Viedma, E., 2012. A hybrid recommender system for the selective dissemination of research resources in a Technology Transfer Office. *Information Sciences,* pp. 1-19.

Rahul, K. & Om, P. V., 2016. A collaborative recommender system enhanced with particle swarm optimization technique. *Multimedia tools and applications,* pp. 9255-9239.

Rajasekar, S., Philominathan, P. & Chinnathambi, v., 2013. *RESEARCH METHODOLOGY.* [Online] Available at: https://arxiv.org/pdf/physics/0601009.pdf

Rajput, A. et al., 2011. J48 and JRIP rules for e-governance data.. *International Journal of Computer Science and Security (IJCSS),* p. 201.

Resnick, P. et al., 1994. *GroupLens: An Open Architecture for Collaborative Filtering.* North Carolina, ACM , pp. 175-186 .

Resnick, P. et al., 1994. *GroupLens: An Open Architecture for Collaborative Filtering of Netnews.* Chapel Hill, USA ©1994.

Ricardo, Y. B. & Berthier, N. R., 1999. *Modwern Information Retrieval.* NY, New York, USA: ACM Press.

Ricci, F., 2011. Information Technology & Tourism. *Mobile Recommender Systems,* 12(3), pp. 205-231.

Ricci, F., Rokach, L. & Shapira, B., 2010. Introduction to Recommender Systems Handbook. In: S. Nature, ed. *Recommender Systems Handbook.* Bozen-Bolzano: © 2017 Springer International Publishing AG, pp. 1-35.

Ricci, F., Rokach, L. & Shapira, B., 2011. *Introduction to Recommender Systems Handbook.* Boston: Springer.

Ricci, F., Rokach, L. & Shapira, B., 2015. Recommender Systems Handbook . In: S. Nature, ed. *Introduction and Challenges.* Bolzano: Springer International Publishing, pp. 1-34.

Robin, B., 2002. Hybrid Recommender Systems: Survey and Experiments. In: S. Nature, ed. *User Modeling and User-Adapted Interaction.* Fullerton: © 2017 Springer International Publishing AG, p. 331–370.

Rocchio, J., 1971. *Relevance feedback,* s.l.: s.n.

Russek, E., Kronmal, R. A. & Fisher, L. D., 1983. The Effect of Assuming Independence in Applying Bayes Theorem to Risk Estimation and Classification in Diagnosis. *Comput Biomed Res,* pp. 537-552.

Salehi, M., 2013. An effective recommendation based on user behaviour: a hybrid of sequential pattern of user and attributes of product. *International Journal of Business Information Systems ,* 14(1), pp. 481-494.

Salter, J. & Antonopoulos, N., 2006. CinemaScreen recommender agent: combining collaborative and content-based filtering. *CinemaScreen recommender agent: combining collaborative and content-based filtering,* pp. 35-41.

Salton, G., 1989. *Automatic text processing: the transformation, analysis, and retrieval of information by computer.* Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Sanger, T. D., 1989. Optimal unsupervised learning in a single-layer linear feedforward neural network.. In: *Neural Networks.* s.l.:s.n., pp. 459-473.

Saptono, R., 2010. *User-Item Based Collaborative Filtering For Improved Recommendation.* s.l., s.n.

Sarif, N. & Ziad, S., 2016. *Evaluating prediction accuracy for collaborative filtering algorithms in recommender systems,* s.l.: KTH computer science and communication.

Sarwar, B., Karypis, G., Konstan, A. J. & Riedl, J., 2002. *Incremental SVD-based algorithms for highly scaleable recommender systems.* s.l., s.n.

Sarwar, B., Karypis, G., Konstan, J. A. & Riedl, J., 2000. *Application of dimensionality reduction in recommender system.* [Online]
[Accessed 2019].

Sarwar, B., Karypis, G., Konstan, J. & Riedl, J., 2001. *Item-Based Collaborative Filtering Recommendation Algorithms.* Hong Kong, ACM, pp. 285-295.

Schafer, B. J., Frankowski, D., Herlocker, J. & Sen, S., 2007. Collaborative Filtering Recommender Systems. In: *The Adaptive Web.* Berlin Heidelberg: Springer, p. 291 – 324.

Schwab, I., Kobsa, A. & Koychev, I., 2001. *Learning User Interests through Positive Examples Using Content Analysis and Collaborative Filtering.* s.l., s.n.

Sebastiani, F., 2002. Machine Learning in Automated Text Categorization. *ACM Computing Surveys,* pp. 1-47.

Semeraro, G., Lops, P. & Degemmis, M., 2005. *WordNet-based user profiles for neighborhood formation in hybrid recommender systems.* Rio de Janeiro, IEEE , pp. 6-11.

Serrano-Guerrero, J. et al., 2011. A google wave-based fuzzy recommender system to disseminate information in University Digital Libraries 2.0. *Information Sciences,* p. 1503–1516.

Setten, M. V., 2005. *Supporting People in Finding Information: Hybrid Recommender Systems and Goal-Based Structuring.* s.l.:s.n.

Shani, G. & Gunawardana, A., 2011. Evaluating Recommendation Systems. In: *Recommender Systems Handbook.* Boston: Springer, pp. 257-297.

Shani, G., Heckerman, D. & Brafman, R. I., 2005. An MDP-based Recommender System. *Journal of Machine Learning Research,* pp. 1265-1295.

Shepperd, M. & Kadoda, G., 2001. *Comparing software prediction techniques using simulation.* s.l., s.n., pp. 1014-1022.

Shinde, U. & Shedge, R., 2013. Comparative Analysis of Collaborative Filtering Technique. *IOSR Journal of Computer Engineering (IOSR-JCE),* pp. 77-82.

Shrkhorshidi, A. S., Aghabozorgi, S. & Wah, T. Y., 2015. *A Comparison Study on Similarity And Dissimilarity Measure in Clastering Continuous Data.* s.l., s.n.

Smyth, B. & Cotter, P., 2000. A Personalized television listings service. *Communications of the ACM,* pp. 107-111.

Sridevi, M., Rajeshwara, R. & Varaprasad, M., 2016. A Survey on Recommender System. *(IJCSIS) International Journal of Computer Science and Information Security,* pp. 265-272.

Sundén, J., 2005. *Material Virtualities: Approaching Online Textual Embodiment,* New York: Peter Lang Publishing.

Su, X. & Khoshgoftaar, M. T., 2009. A Survey of Collaborative Filtering Techniques. *Advances in Artificial Intelligence,* pp. 1-20.

Su, X. & Khoshgoftaar, T., 2009. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence,* 2009(4).

Su, X. & Khoshgoftaar, T. M., 2009. A Survey of Collaborative Filtering Technique. *Advances in Artificial Intelligence.*

Taurshia, A. A. & Kanmani, D. S., 2013. Recommender System And Ranking Techniques: A Survey. *International Journal Of Engineering Research and Applications,* 3(1), pp. 491-493.

Thai-Nghe, N., Nhut-Tu, M. & Huu-Hoa, N., 2017. An Approach For Multi-Relational Data Context In Recommender Systems. In: *Intelligent Information And Database Systems.* s.l.:s.n., pp. 709-720.

Ting, K. M., 2011. Precision and Recall. In: C. Sammut & G. Webb, eds. *Encyclopedia of Machine Learning.* 2010 Edition ed. Boston: Springer, pp. 33-45.

Torres, R. D., 2004. *Combining Collaborative and Content-based Filtering to Recommend Research Paper.* s.l., Porto Alegre.

Tuan, D., 2019. *findoutyourfavorite.* [Online] Available at: http://findoutyourfavorite.blogspot.com/2012/04/content-based-filtering.html

Ungar, H. L. & Foster, D. P., 1998. *Clustering Methods for Collaborative Filtering.* s.l., Madison.

Universidad Complutense de Madrid, S., 2009. *Group of Artificial Intelligence Applications.* [Online] Available at: http://gaia.fdi.ucm.es/research/happymovie/download [Accessed 20 04 2018].

Walther, . J. B., Bunz, U. & Bazarova , N. N., 2005. *The Rules of Virtual Groups.* Hawaii, IEEE, pp. 1-10.

Wikimedia Foundation, I., 2018. *Wikipedia.* [Online] Available at: https://en.wikipedia.org/wiki/Recommender_system [Accessed 23 October 2018].

Williams, C., Mobasher, B. & Burke, R., 2007. Service Oriented Computing and Applications. *Defending recommender systems: detection of profile injection attacks,* 1(3), p. 157–170.

Witten, I. & Bell, T. C., 1991. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory* , pp. 1085-1094.

Xiaoyuan, S. & Taghi, M., 2009. A Survey of Collaborative Filtering Techniques. *Advances in Artificial Intelligence*, 9 February, Volume 2009, pp. 1-20.

Ye, N., Chai, K. M. A. & Chieu, H. L., 2012. *Optimizing F-Measure : A Tale of Two Approaches.* Edinburgh, Scotland, s.n.

Yeung, C. H., 2015. Journal of Statistical Mechanics Theory and Experiment. *Do recommender systems benefit users?,* 1(1), pp. 1-32.

Yu, Z., Zhou, X., Hao, Y. & Gu, J., 2006. TV program recommendation for multiple viewers. *User Model User-Adap Inter* , 1(1), p. 63–82.

Zhang, W. & Feng, G., 2011. An Improvement to Naive bayes for Text Classification. *Procedia Engineering,* Volume 15, pp. 2160-2164.

Zheng, M., Min, F., Zhang, H.-R. & Chen, W.-B., 2016. *Fast Recommendations With the M-Distance.* s.l., IEEE.

Zhongqi, L. et al., 2015. *Content-based collaborative filtering for news topic recommendation.* Austin, Texas, AAAI Press ©2015.

Zuva, T., Ojo, S. O., Ngwira, A. M. & Zuva, K., 2012. A Survey of Recommender Systems Techniques, Challenges and Evaluation Metrics. *International Journal of Emerging Technology and Advanced Engineering,* pp. 382-386.