Performance Evaluation of Ensemble based Recommendation System using Biased Matrix Factorization and Content Based Filtering

Jayesh Deogirikar
Student ID: 10514141

MSc Research Project
Data Analytics

Dublin Business School
06 Jan 2020

Supervisor: Terri Hoare
Declaration

I, Jayesh Vinayak Deogirikar, declare that this research is my original work and that it has never been presented to any institution or university for the award of Degree or Diploma. In addition, I have referenced correctly all literature and sources used in this work and this work is fully compliant with the Dublin Business School’s academic honesty policy.

Jayesh Vinayak Deogirikar
# Contents

1 Introduction .................................................. 9
   1.1 Background ............................................. 9
   1.2 Business Problem ...................................... 9
   1.3 Research Problem ...................................... 11
      1.3.1 Research Question ................................ 11
      1.3.2 Aim ............................................... 12
      1.3.3 Objective ......................................... 12
      1.3.4 Hypothesis ....................................... 12
   1.4 Scope .................................................... 12
   1.5 Limitations ............................................. 12
   1.6 Dissertation Roadmap ................................ 12

2 Literature Review ................................................. 13
   2.1 Introduction ........................................... 13
   2.2 Collaborative Filtering ................................ 13
      2.2.1 User Based Collaborative Filtering .............. 14
      2.2.2 Item Based Collaborative Filtering .............. 14
   2.3 Content Based Filtering ................................. 15
   2.4 Latent Matrix Factorization ............................. 15
      2.4.1 Basic and Biased Matrix Factorization ........... 15
      2.4.2 Single Value Decomposition ....................... 16
   2.5 Ensemble based approach ............................... 17

3 Methodology ..................................................... 18
   3.1 Tools Used ............................................. 18
   3.2 Business Understanding ................................ 19
   3.3 Data Understanding .................................... 19
   3.4 Data Preparation ........................................ 21
   3.5 Modelling ............................................... 21
      3.5.1 Item based collaborative filtering ............... 22
      3.5.2 Biased Matrix Factorization ....................... 26
      3.5.3 User profile Content based filtering ........... 30
   3.6 Ensemble based recommendation system ............... 34
      3.6.1 Weighted Ensemble models ......................... 35
      3.6.2 Evaluation ......................................... 36
      3.6.3 Deployment ......................................... 36

4 Analysis .......................................................... 37
   4.1 Exploratory Data Analysis .............................. 37
   4.2 Model Result ............................................ 40
      4.2.1 Using RapidMiner .................................. 41
      4.2.2 Using Spyder for python: ......................... 43
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>CRISP-DM model</td>
<td>18</td>
</tr>
<tr>
<td>3.2</td>
<td>Item Based Collaborative Filtering</td>
<td>22</td>
</tr>
<tr>
<td>3.3</td>
<td>Rating Matrix and User-item Rating Prediction</td>
<td>24</td>
</tr>
<tr>
<td>3.4</td>
<td>Transposed Rating Matrix</td>
<td>24</td>
</tr>
<tr>
<td>3.5</td>
<td>Normalized Ratings and Similarity with a movie</td>
<td>25</td>
</tr>
<tr>
<td>3.6</td>
<td>Items and users in latent factor space, (Kotu and Deshpande, 2019)</td>
<td>27</td>
</tr>
<tr>
<td>3.7</td>
<td>User profile Content based filtering</td>
<td>30</td>
</tr>
<tr>
<td>3.8</td>
<td>User profile Content based filtering</td>
<td>31</td>
</tr>
<tr>
<td>3.9</td>
<td>User profile from utility matrix and item profile</td>
<td>32</td>
</tr>
<tr>
<td>3.10</td>
<td>Rating Matrix R and Item Profile</td>
<td>33</td>
</tr>
<tr>
<td>3.11</td>
<td>User Profile U</td>
<td>33</td>
</tr>
<tr>
<td>3.12</td>
<td>Weighted Ensemble Model</td>
<td>34</td>
</tr>
<tr>
<td>4.1</td>
<td>Long Tail</td>
<td>37</td>
</tr>
<tr>
<td>4.2</td>
<td>Histogram</td>
<td>38</td>
</tr>
<tr>
<td>4.3</td>
<td>Top Rated Movies</td>
<td>39</td>
</tr>
<tr>
<td>4.4</td>
<td>Tree Map</td>
<td>40</td>
</tr>
<tr>
<td>5.1</td>
<td>Process of Item based collaborative filtering</td>
<td>46</td>
</tr>
<tr>
<td>5.2</td>
<td>Code snippet Item based collaborative filtering</td>
<td>47</td>
</tr>
<tr>
<td>5.3</td>
<td>Process of Biased Matrix Factorization</td>
<td>49</td>
</tr>
<tr>
<td>5.4</td>
<td>Code snippet for Biased Matrix Factorization</td>
<td>50</td>
</tr>
<tr>
<td>5.5</td>
<td>Process of User profile content-based filtering</td>
<td>52</td>
</tr>
<tr>
<td>5.6</td>
<td>Process of Item profile generation</td>
<td>52</td>
</tr>
<tr>
<td>5.7</td>
<td>Code snippet of User profile content-based filtering</td>
<td>53</td>
</tr>
<tr>
<td>5.8</td>
<td>Process of Ensemble Based recommender System without BMF</td>
<td>55</td>
</tr>
<tr>
<td>5.9</td>
<td>Process of Ensemble Based recommender System without BMF</td>
<td>56</td>
</tr>
<tr>
<td>5.10</td>
<td>Process of Ensemble Based recommender System</td>
<td>58</td>
</tr>
<tr>
<td>5.11</td>
<td>Code snippet of Ensemble Based recommender System</td>
<td>59</td>
</tr>
</tbody>
</table>
**List of Tables**

<table>
<thead>
<tr>
<th>Table Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2.1</td>
<td>Result illustrating accuracy for Biased Matrix Factorization</td>
<td>41</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Result illustrating accuracy for Biased Matrix Factorization</td>
<td>41</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Result illustrating accuracy for User profile content-based filtering</td>
<td>42</td>
</tr>
<tr>
<td>4.2.4</td>
<td>Result illustrating accuracy for Ensemble based recommender without BMF</td>
<td>43</td>
</tr>
<tr>
<td>4.2.5</td>
<td>Result illustrating accuracy for Ensemble based recommender system</td>
<td>43</td>
</tr>
<tr>
<td>4.2.6</td>
<td>Result illustrating accuracy for Item based collaborative filtering using Python</td>
<td>44</td>
</tr>
<tr>
<td>4.2.7</td>
<td>Result illustrating accuracy for Biased Matrix Factorization using Python</td>
<td>44</td>
</tr>
<tr>
<td>4.2.8</td>
<td>Result illustrating accuracy for User profile content-based filtering using Python</td>
<td>44</td>
</tr>
<tr>
<td>4.2.9</td>
<td>Result illustrating accuracy for Ensemble based recommender without using BMF</td>
<td>45</td>
</tr>
<tr>
<td>4.2.10</td>
<td>Result illustrating accuracy for Ensemble based recommender without using BMF</td>
<td>45</td>
</tr>
<tr>
<td>5.2.1</td>
<td>Result Item based collaborative filtering</td>
<td>46</td>
</tr>
<tr>
<td>5.2.2</td>
<td>Result Item based collaborative filtering using python code</td>
<td>47</td>
</tr>
<tr>
<td>5.2.3</td>
<td>Top 10 recommendation</td>
<td>48</td>
</tr>
<tr>
<td>5.2.4</td>
<td>Result of Biased Matrix Factorization</td>
<td>48</td>
</tr>
<tr>
<td>5.2.5</td>
<td>Result Item based collaborative filtering using python code</td>
<td>49</td>
</tr>
<tr>
<td>5.2.6</td>
<td>Top 10 recommendation</td>
<td>51</td>
</tr>
<tr>
<td>5.2.7</td>
<td>Result of Biased Matrix Factorization</td>
<td>53</td>
</tr>
<tr>
<td>5.2.8</td>
<td>Result of Biased Matrix Factorization</td>
<td>53</td>
</tr>
<tr>
<td>5.2.9</td>
<td>Top 10 recommendation</td>
<td>54</td>
</tr>
<tr>
<td>5.2.10</td>
<td>Result of Biased Matrix Factorization</td>
<td>55</td>
</tr>
<tr>
<td>5.2.11</td>
<td>Result of Biased Matrix Factorization</td>
<td>55</td>
</tr>
<tr>
<td>5.2.12</td>
<td>Top 10 recommendation</td>
<td>57</td>
</tr>
<tr>
<td>5.2.13</td>
<td>Result of Biased Matrix Factorization</td>
<td>58</td>
</tr>
<tr>
<td>5.2.14</td>
<td>Result of Biased Matrix Factorization</td>
<td>58</td>
</tr>
<tr>
<td>5.2.15</td>
<td>Top 10 recommendation</td>
<td>60</td>
</tr>
</tbody>
</table>
List of Algorithms

- Biased Matrix Factorisation.
- Item based collaborative filtering.
- User profile content-based filtering.
- Ensemble based model without BMF.
- Ensemble based model with ItemKNN, User profile content-based filtering and BMF.
List of important abbreviations

• BMF - Biased Matrix Factorisation
• CF - Collaborative Filtering
• CRISP-DM - Cross-Industry Standard Process for Data Mining
• KNN - K Nearest Neighbours
• MAE - Mean Absolute Error
• RMSE - Root Mean Square Error
• RS - Recommendation System
• SVD - Single Value Decomposition
• TF-IDF - Term Frequency-Inverse Document frequency
Acknowledgments

I want to express my special thanks of gratitude to Ms. Terri Hoare, my teacher and research supervisor for her patient guidance from the beginning with constructive and valuable suggestions provided throughout the planning and improvement of my dissertation.

I would like to specially thanks to RapidMiner, Google Developers tutorial forum, and Python documentation to provide free access to valuable machine learning materials.

Last but not least, I would like to thank my family and friends for their support and encouragement throughout my course.
Abstract

Boosting product sales is the primary objective of any Recommender System and it is achieved by providing users a personalized recommendation from the world of information density and product overload. The suggestions given by RS support users in their decision-making process such as which item to buy from thousands of items, which music to listen to from millions of soundtracks or which news feed to read first. Recommender System can be implemented by using Collaborative or Content-based filtering techniques and each of them has its own merits and demerits. Aiming at this concern, this research proposed an ensemble model combining Item-KNN, Biased Matrix factorization and user profile content-based filtering. The result for comparative analysis on MovieLens data shows that the ensemble model outperforms each model when implemented individually and ensemble model without BMF giving an accuracy score of 0.818 stars as RMSE and 0.629 stars as MAE.
1 Introduction

1.1 Background

“Many receive advice, only the wise profit from it”– Harper Lee

Nowadays, machine learning has become essential driving force for almost all business, and everyone can experience growing impact of it on product and applications. Amongst all techniques of machine learning, recommendation systems provide most productive utilities in everyday experience and it predicts user preference for an item. In a nutshell, the purpose of any enterprise is to connect a customer with product which customer wishes for and recommender systems influences this connection by introducing relevant products to customer for consumption. Whether it is new friend recommendation, what to watch next in streaming services, which article to read from news feed or which product to by next from e-commerce website the role of recommendation systems is very crucial in today’s digital world (Kotu and Deshpande, 2019).

Internet penetration and evolution of smartphones increased importance of Web in electronic and business transaction. Feedback given by user about liking or disliking for product acted as important catalyst in evolution of Recommender Systems. Feedback can be explicit such as rating given in numeric scale such as 0 to 5 rating or can be implicit feedback in which user mouse clicks for products can be consider as user endorsement for that product. All this explicit or implicit feedback stored by merchants effortlessly and sole purpose of recommendation systems are to utilize this stored feedback data and infer user interests (Agarwal, 2016).

1.2 Business Problem

Online retail is moving forward with blazing pace not only in just metro areas but also in semi urban areas of world. Before purchasing any small product, customer is exploring numerous channels to finalise buying decision. Online retail has become vital for every retailer in marketplace due to aggressive cut-throat competition to grab pool of customers. Every retailer is finding innovative way to provide excellent customer service with bridging gap with help of online technologies which leads to gain trust of customers.

The brick and mortar store can hold only number of items due to space constraint, same way an enterprise with few dozens of products has some limited number of choices to customers. In such cases it is very likely to
recommend customer top selling products or selling product with Market Basket Analysis technique also known as Association analysis where enterprise recommends exactly same choices of products to all its user. With expansion of online business, user can have millions of product options to consume from and in such scenario most popular products serve instant pick for user and log tail products can serve purpose of some obscure choices keeping novelty alive in recommendation system (Kotu and Deshpande, 2019).

Recommendation systems have the capability to change how user interact with website while purchasing items which make possible to companies to maximize their Return on investment depending upon information, they collect for every customers items consumption or preference (Underwood, 2019). Below are some reasons why merchants should need to exploit recommender systems in their existing website:

- Increase volume of sales: Primary aim of recommender system is to increase conversion rate as user will tend to buy more recommended products which rather than just browsing website (Ricci et al., 2011).

- Selling more distinct items: As recommender system uses leverage long tail concept due to which user able to see more obscure items recommended along with most popular items (ibid.).

- Increase user trust and satisfaction: Recommender system takes user feedback for item, so user feels privilege that merchant is considering his opinion which builds trust and customer satisfaction (ibid.).

- Improving customer retention: Recommendation system has ability to continuously calibrate system with users’ preferences, so it becomes very unlikely that user will go to other competitor merchants for purchasing (Underwood, 2019).

Recommendation Systems are broadly classified as below:

- Collaborative Filtering: This type of recommender system uses rating matrix as past user-item interaction. It works on assumption that similar user has similar likings. It uses rating given by similar user for provide prediction for unknown user-item interaction. Collaborative filtering further divided into Neighbourhood methods and Latent factor models (Kotu and Deshpande, 2019).

- Content-based filtering: Item rating profile is used in addition to user-item profile which used in collaborative filtering for prediction.
profile is attributes data about item such as cast played in movie, genre, director or released year and so on. Together item profile and user-item rating content-based filtering can predict user-item preference by two methods known as User profile methods or Supervised learning models (Kotu and Deshpande, 2019).

- Hybrid/Ensemble Recommenders: Collaborative filtering as well as Content based filtering has its own merit and demerits for user-item prediction in different scenarios. For leveraging merits from each single model, multiple models are combined for predicting user-item preferences and such models are known as Hybrid or Ensemble models. For implementing Ensemble recommender base models must be independent. Ensemble models improve generalisation error and eliminate limitation of single models (ibid.).

In 2006, the online DVD rental company Netflix announce a challenge in which it released dataset of approximately 3 million ratings with movie id and user id. Goal of challenge was to improve RMSE value of recommender system by 10 percent or more wins a $1 million prize and $50 000 prize to team at first place after every year of contest. This contest spread a ripple in research and development community in dormant recommendation system field. The winner of that contest was team called as BellKor who improved RMSE by 8.43 percent than Netflix. By that time till now various attempts has been made with different using algorithms to improve RMSE (accuracy) value of recommendation system (Koren and Bell, 2007).

This paper aims to improve accuracy by implementing weighted hybrid model which combined Biased matrix factorization, Content-based filtering and item KNN collaborative filtering. Dataset used for this paper is MovieLens 100K dataset.

1.3 Research Problem

The recommendation system leads to build trust between the user and item consumption hence it is very crucial to recommend users with accurate movie recommendations for the growth of the business. This research is conducted using MovieLens 100K dataset on which numbers of recommender algorithms implemented to compare their performance.

1.3.1 Research Question

Comparison of Ensemble based recommendation system using BMF with Ensemble based recommendation system without BMF for movie recom-
mendation.

1.3.2 Aim

Predict movie recommendation with the best accuracy metrics for each user to grow in business.

1.3.3 Objective

To find a recommendation model with the best accuracy using RapidMiner and Spyder for python code implementation. Also, recommend top 10 results as per user preference using Spyder.

1.3.4 Hypothesis

Ensemble model of biased matrix factorization, content-based filtering, and collaborative item KNN outperform Ensemble model of an algorithm using collaborative item KNN and user KNN in accuracy metrics.

1.4 Scope

The scope of this research is to build a recommendation system using separate as well as ensemble algorithm to accurately predict movie ratings.

1.5 Limitations

This research built on a machine which is powered by Windows 10 operating system, i5 intel processor and 8 Gb of RAM without any GPU installed.

1.6 Dissertation Roadmap

This research implemented using a strategic plan, the following is a roadmap followed.

- Introduction: This chapter contains problem definition, research question, aim, objective and hypothesis to be tested.

- Literature Review: This chapter highlights existing research done on recommendation system with use of research journals and textbooks.

- Methodology: This chapter utilizes CRISP-DM approach to conduct this research by following six phases of CRISP-DM.
Data Analysis: The aim of this chapter is to compare finding and performance of each algorithm without including findings.

Discussion: This chapter include interpretation and discussion on result and answering research question.

Conclusion: This chapter will summarise the finding of the research.

2 Literature Review

2.1 Introduction

In order to achieve an efficient recommender system, numerous of techniques and approaches have been put forth so far. This section describes algorithms improved to achieve best accuracy and efficiency with respect to recommender system. Collaborative Filtering widely known technique comes with a motivation to understand user’s similar interest and item matrix determining a relationship between two items (Bordes et al., 2013). On the other hand Content Based approach is another approach that recommends based on item characteristic and user preference (Gemulla et al., 2011). These approaches result to data sparsity and scalability issues reducing the accuracy. An enhanced approach of Item-based K-NN algorithm proposed by (Xia et al., 2010) and user-based by (Lumauag, Sison and Medina, 2019) to achieve lower RMSE and MAE value can be seen further.

Moreover, basic and bias matrix factorization known to be providing higher accuracy than Collaborative Filtering (Peiliang, 2016). Finally an attempt to ensemble two or three algorithms to get the best of both worlds have been carried out. Where author (Venil, G and R, 2019) has proposed a hybrid of user and item based CF, a combination with improved techniques. Author (Song, Liu and Ji, 2017) cites an improved algorithm tweaking and combining the existing approaches. Many such different techniques and algorithms are put ahead to achieve effective results in regards to accuracy by different authors.

2.2 Collaborative Filtering

The history on collaborative filtering goes way long, initially was started on emails (Goldberg et al., 1992) and since then many researchers worked further. An algorithms on user-based comparing and relating the users
using the k nearest neighbours and later more efficient and faster approach comparing items than users was taken into account.

2.2.1 User Based Collaborative Filtering

A taint on accuracy can lead to poor recommendation quality, author (Lumauag, Sison and Medina, 2019) as given a brief overview on traditional approach, the reasons of inaccuracy when it comes to considering predicted rating and overfitting issues as of user’s having only interest to few items. This can be improved with help of modified approach, wherein similar user’s are classified and dissimilar users and inactive users are discarded. Its further computed with most similar user’s using Pearson Correlation Coefficient, setting a threshold and then listed as the nearest neighbors. Using the number of neighbors k from 5 to 50 the RMSE value changes, for k =10 , the RMSE value is equal to 1, for K=20 the value is equal to 0.98.

(Xu et al., 2019) states that user similarity is one of crucial procedure of user-based approach, having that in mind, author have proposed new approach based on user confidence and time context. Confidence refers to the probability that the ratings or values falls in a range and time context done by adding time attenuation formula, the longer the user takes to behave to certain object depending on interest. Both of this experiment is evaluated on MovieLens, implementing time context and user confidence on all of the algorithms cosine-based similarity (COS), the Pearson correlation coefficient (PCC), the Euclidean distance-based similarity (EDS), and the adjusted cosine-based similarity (ACOS). Author shows the result to fall between 1.020 to 0.950 for RMSE value and 0.810 to 0.750 for MAE value.

2.2.2 Item Based Collaborative Filtering

(Sarwar et al., 2001) To understand the relationship between different items, Item-Based techniques first analyze the user-item matrix as K-NN algorithms and then computes for user similarity. (Xia et al., 2010) An approach to improve accuracy by adjusting the deviation has been proposed on MovieLens dataset. The optimization is divided into two types Average Filling, where the missing item ratings are filled with the average for which MAE calculation comes to 0.724. Another type being Deviation Adjustment which helps to minimize the error between predicted and actual preference of users and the MAE calculation falls to 0.709 the K value here is taking to be 20. Author concludes that
both approaches help gain accuracy and rectify several uniform errors. However including a new side features for query/item taking the example of movie dataset could be country or year is challenging in item-based CF also filling an average pre-computing is a time-consuming task especially for much bigger datasets.

Additionally (Clemente, 2008) compares the datasets for different types such as NetFlix, MovieLens, BookCrossing and Jester and comes to conclusion the item-based algorithm quality depends on the dataset used for the experiment and the RMSE values are all lower and acceptable. For movie lens the RMSE value for k 32 to 100 falls between 0.8908 to 0.8928 with that datasets have high implications on the results states author. Nonetheless only four datasets were used to put forth the conclusion which are not certainly enough.

2.3 Content Based Filtering

(Fan, Pan and Jiang, 2007) A content-based refers to past user’s behavior, helping to find the relevant one, just as on basis of user’s like on an item. A approach to alleviate data spareness is by using the algorithm to predict the missing value. To do this user activity is recorded depending on likes, comments on certain items, rating value. With K neighbors taking MovieLens data for k=20 the MAE is shown to be falling at 0.83 which increases as the k value increases. However the approach takes only two factors when the most prominent drawback is struggle of exploring items or factors outside the usual choice.

(Peiliang, 2016) To overcome this issue, an enhanced content-based filtering using diverse collaborative prediction approach. With CF takes in active user accounts and there inputs comparing with the neighbours. Diverse-item selection algorithm is then applied to which calculates dissimilarity, taking out highest score.

2.4 Latent Matrix Factorization

2.4.1 Basic and Biased Matrix Factorization

A matrix presents a memory efficient model providing higher accuracy compared to Collaborative Filtering (Paterek, 2007). Biases are entirely produced by either of one entities user behaviour or item matrix and not on both.

(Lak, Caglayan and Bener, 2014) There are different types of biases states has captured the two main types with 80 % of training data and 20 %
for test performed on MATLAB by fine tuning the parameters used in the basic MF model. The study shows use of simple user and bias item for improving accuracy and a regulation factor is added to avoid overfitting. Although the accuracy seems to be increasing, the cost of adding models of biases will increase explains author. (Zhao and Xiao, 2018) Depicts 4 different methods to obtain the lowest RSME or MAE value which indicates better performance. First method using the Basic Matrix Factorization (MF) with predicted rating user u to i as shown below

\[ r_{ui} = \sum_f P_{uf}q_{if} \]

The next method is performed adding biases to the basic matrix \( b_u \) being the biases on user and \( b_i \) as item biases shown below.

\[ r_{ui} = \mu + b_u + b_i + p_u^T q_i \]

Further methods Category weight and Neighbour impact is taken into account, depending on user’s preferences in different categories and impact from friends or another individual recommendation. The result shows the basic MF has latent factors hence performing the poorest, Biases MF helps increase by 1% than that of basic and NICW has about 2% increase. For biases MF MAE value is equal to 0.8057 and RMSE is equal to 1.0389 for latent factor \( F=20 \). However according to (Chen et al., 2018) with neighborhood based CF constructing an item subspace is complex and prediction is poor which is not been considered in this paper.

### 2.4.2 Single Value Decomposition

(Vozalis and Margaritis, 2007) Demand for Single Value Decomposition (SVD) is because of its high scalability and accuracy in recommendation systems. SVD is a part of Latent Semantic Indexing largely used in information retrieval solving synonymy and polysemy issues. (Koren and Bell, 2007) Limitations to SVD is it cannot perform if matrix contains missing values states therefore author presents an approach filling out the missing values with appropriate value, using SVD’s ability to reduce dimension which takes mXn matrix. the filling is performed on the MovieLens dataset with respect to median of user’s ratings, item ratings and total ratings. For a normal when author compares the RMSE value falls at 2.3379 and MAE falls at 2.0014. The value changes for evaluated SVD to 1.0708 and MAE 0.8945. However imputations can be high-priced
due to increase in data, and the data can be misleading also overfitting can be one of the problem in SVD (Canny, 2007).

### 2.5 Ensemble based approach

Ensemble filtering refers to a combination of two or more approaches, this combination helps overcome lack of one approach to another. (Venil, G and R, 2019) proposed an ensemble based k-NN Collaborative Filtering (CF) approach, blending the quality work of user based k-NN algorithm and item-based algorithm. To perform the analysis MovieLens dataset is taken into consideration giving in personalized movie recommendation and is carried out in rapidminer considering only 5 star ratings. The accurate results are shown having lesser values, the Root Mean Squared Error (RMSE) of the ensemble- Based k-NN is 1.07 lesser than that of User- based k-NN 1.08 and item- based k-NN being 1.13. The value of Mean Absolute Error being 0.85 for ensemble based and 0.86 and 0.93 for user based and item based respectively. To address the problem of data sparsity and problem of scalability in conventional collaborative filtering algorithms approach was carried out. However the implementation only considers small dataset size making the experiment incomplete, as the author uses commercial version of rapidminer which takes in only limited amount of data and hence only 9000 instances are taken in from the dataset.

(Song, Liu and Ji, 2017) Another similar Hybrid approach to resolve the data sparse problem and improve the accuracy and efficiency has been carried out by based on personalized recommendation system model on users and items. The two individual algorithms are used on MovieLens-100k dataset using MATLAB taking in user’s behaviour and also similarity between user’s which is improved with help of Pearson correlation coefficient and an Item-based to calculate the similarity between items using correlation coefficient formula is used. Then a combination of both improved techniques is later used showing RMSE value to be 3.11 states author.

(Shi, Ye and Gong, 2009) As discussed CF shows data sparsity issues as both methods only taken in one-directional information from the user-item ratings matrix, to utilize the data author has proposed an algorithm using both methods item-based and user-based taking in vertical and horizontal information in the user-item matrix. This experiment is carried out on MovieLens dataset and results shows the accuracy in terms of decision-
support metrics, helping user to select high quality items. Author con-
cludes having the algorithm helps increase accuracy, however author has
not considered RMSE or MAE values to conclude so.

3 Methodology

3.1 Tools Used

Below are tools used for this research:

- RapidMiner
- Sypder
- Tableau

CRISP-DM (Cross-Industry Standard Process for Data Mining) has
been followed for this research and it is widely adopted Data Science solu-
tion. CRISP-DM model is iterative cycle model and steps involved in this
model are non-linear which means model can take back and forth number
of loops in stages (Shearer, 2013).

Figure 3.1: CRISP-DM model

(Kotu and Deshpande, 2019)
CRISP-DM Figure 3.1 methodology follows six stages to give optimal solution to data science process, which are as below:

1. Business Understanding
2. Data Understanding
3. Data Preparation
4. Modelling
5. Evaluation
6. Deployment

Above stages of CRISP-DM model will be approached in detail for research.

3.2 Business Understanding

Business understanding is most vital part of any data mining project, in this stage project objective is try to understand from business point of view and convert all before hand information from business in Data mining definition to plot future implementation of project according its objective (Shearer, 2013)

This research’s objective is to improve accuracy of Recommendation System using Ensemble Recommendation System approach using Biased Matrix factorization, Item based collaborative method and user profile content-based filtering. In practical, there are various algorithms or techniques employed for improving accuracy of Recommendation System in past such as Collaborative models, Content-based filtering or Hybrid models. Research aim to ensemble all three models linearly together and done comparative analysis with past implemented models with referring accuracy metrics as RMSE and MAE value.

This study is conducted to answer research question “Can ensemble model with BMF, content-based and item KNN outperform not only single implemented techniques but also ensemble model”.

3.3 Data Understanding

Data understanding phase starts with collecting initial data from available data sources. After data collection data has been investigate for initial insight such as quality of collected data or any interesting pattern in subset of collected data (ibid.).
For this research Movielens 100k dataset has been used, which has been collected by GroupLens Research Project at University of Minnesota. Dataset contains collection of movie rating for 1682 movies by 943 users. Not every user has seen or rated each movie, in total there are 100,000 ratings available. Each user and movie have assigned unique Id for identification and rating given is from 0 to 5-star range where 5 is considered as highest rating for respective movie.

Dataset consist of four files as below:

- links.csv
- movies.csv
- rating.csv
- tags.csv

From above four csv files, in this research we have used movies.csv and rating.csv files.

movies.csv file contains below attributes:

- movieId: It is unique identification key for each movie
- title: Title name for each movie
- genres: Name of genres to which each movie is classified into. Genres are delimited by pipes as each movie can have multiple genres.

rating.csv contains below attributes:

It is user-rating matrix file in which user has given his/her rating for specified movie.

- userId: It is identification id for each user and it is not unique.
- movieId: It is identification id for each movie which respective user has rated and it is not unique.
- rating: Rating given by respective user for respective movie, value can be 0 to 5 stars where 5 is considered as highest rating or user like the movie most.
- timestamp: Represent seconds since midnight Coordinated Universal Time (UTC) of January 1, 1970.
3.4 Data Preparation

Data preparation phase carry out all tasks which convert raw data from different sources to clean and final data which is ready to feed to modelling. Task carried out in data preparation like check for empty/null values, transform data, attribute selection for modelling tools (Shearer, 2013). Data preparation done in below steps:

- First check is there any empty on null value present in movie.csv as there are not any empty value present in file and not single duplicate entry for any movie so movie.csv is clean file to go ahead and use for modelling.

- As all attributes in movie.csv are necessary so it is not necessary to apply feature selection on movie.csv file.

- Second file used for modelling is rating.csv which is matrix for user and movie rating, it contains empty values for rating attribute for some movies where user did not see that movie or did not care to rate that movie. As building recommender system, it is model aim to predict rating which user has left blank out.

- From rating.csv file, timestamp attribute has been dropped off from using feature selection as it is redundant for modelling.

- For building recommendation system, it is mandatory to have user rating matrix so movie.csv and rating.csv file has been merged to get consolidated view with title, genre, user and rating for each movie.

3.5 Modelling

Different modelling techniques selected and applied in this phase with optimal required parameters. As there are several different modelling techniques available for same problem statement, so sometimes it become inevitable to go back to Data Preparation stage due to some techniques has specific requirement for form of data. Selection of modelling techniques, generation of test design, model creation and its assessment included in this stage (Shearer, 2013).

In this research, first single models are trained and tested on training and testing dataset and RSME and MAE value has been measured. After that ensemble model has been designed with linearly combining each single model and RMSE and MAE value of ensemble model compared with all
models to check which models perform better for recommender system. Below models are selected for improving recommender system’s accuracy on research conducted on this field:

1. Item based collaborative filtering.
2. Biased Matrix Factorization.
3. User profile content-based filtering.
4. Ensemble based recommender system.

3.5.1 Item based collaborative filtering

Item based collaborative filtering comes under Collaborative type neighbourhood-based recommender system which based on simple idea that user will pick an item if it is recommended by his/her similar minded friend. Collaborative filtering recommendation takes sparse rating matrix as input and returns with full completed rating matrix with most relevant items for each user as shown in Figure 3.2. Algorithm in collaborative filtering predicts user preference for an item by finding similar users who has rated the same product and user-item preference will be deduced by rating given by similar users. Collaborative filtering methods only consider past rating given by user also called as Rating Matrix and are independent of item domain. The same algorithm can be applied to any other items like music, news or homes as collaborative recommendation system only have knowledge about

Figure 3.2: Item Based Collaborative Filtering

Kotu and Deshpande. [2019]
rating given for specific items. Neighbourhood based methods finds similar user or item from rating matrix by calculating similarity score such as Jaccard Similarity, Cosine Similarity and Pearson Correlation (Kotu and Deshpande, 2019).

User-based or Item-based neighbourhood collaborative method? User-based or Item-based neighbourhood techniques very similar, although the user-item preference rating will be different for both techniques when applied on same rating matrix (ibid.). Conceptually, it is feasible to find similar items than similar user as item has specific types related to them which are less to change over time whereas user can has multiple preference dimension and tends to change over time, due to which item-based collaborative filtering becomes effective and deliver better performance (ibid.).

Item based neighbourhood method follow an assumption that user will choose items that will be similar to previously preferred items, which means similar items tends to get similar rating from same user. In rating matrix, if rating pattern of two items are similar then their item vectors are close and neighbour to each other. Therefore, items in neighbouring sets tends to get similar rating when rated by same user. Two steps process taken for predicting new user-item preference:

1. Find a set of N other items for every item i, which have similar ratings when rated by same user.

2. By averaging the rating of N similar items rated by same user, approximate rating for item i.

Using above rating matrix, we will be predicting rating for movie 2001: A Space Odyssey for user Olivia with help of Item-Based Collaborative filtering (ibid.).

Step 1: Transpose rating matrix
In Item-based, similar items need to find unlike User-based Collaborative filtering where similar users need to find for this reason rating matrix will be transposed. Step 2: Identifying similar Items

Similarity between two items calculated by similarity between their rating vector. Rating vector for item 2001: A Space Odyssey is given as rfargo = 0,2,4,1,1,2. Cosine Similarity, Jaccard Similarity or Pearson correlation coefficient are commonly used quantifying metrics for calculating similarity between item vectors.

Cosine similarity between two non-zero item vector can be calculated as:
Figure 3.3: Rating Matrix and User-item Rating Prediction

<table>
<thead>
<tr>
<th></th>
<th>The Godfather</th>
<th>2001: A Space Odyssey</th>
<th>The Hunt for Red October</th>
<th>Fargo</th>
<th>The Imitation Game</th>
</tr>
</thead>
<tbody>
<tr>
<td>Josephine</td>
<td>5</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Olivia</td>
<td>?</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Amelia</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Zoe</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Alanna</td>
<td>5</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Kim</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

Kotu and Deshpande, [2019]

Figure 3.4: Transposed Rating Matrix

<table>
<thead>
<tr>
<th></th>
<th>Josephine</th>
<th>Olivia</th>
<th>Amelia</th>
<th>Zoe</th>
<th>Alanna</th>
<th>Kim</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Godfather</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2001: A Space Odyssey</td>
<td>4</td>
<td>?</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>The Hunt for Red October</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Fargo</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>The Imitation Game</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
</tbody>
</table>

Kotu and Deshpande, [2019]
\[ Cosine \ similarity(|x.y|) = \frac{xy}{||x|| \ ||y||} \]

\[ Cosine \ similarity(r_{olivia}, r_{amelia}) = \frac{0 \times 5 + 0 \times 1 + 2 \times 4 + 2 \times 4 + 4 \times 1}{\sqrt{2^2 + 2^2 + 4^2} \times \sqrt{5^2 + 1^2 + 4^2 + 1^2}} = 0.53 \]

In cosine similarity score lack of rating considers as zero rating for that item, this consideration is not feasible in case of movie rating as lack of rating does not mean that user did not like the movie, it can be due to user did not see that movie. So, enhance similarity measure needs to be applied for not considering lack of rating in rating matrix as low rating. Centred cosine similarity fix this problem by normalizing entire rating matrix in such a way that first item average rating has been calculated and then difference from each rating with average rating calculated, so if normalised rating is positive then it is above average and if negative then below average rating Kotu and Deshpande, [2019].

\[ Cosine \ similarity(r_{olivia}, r_{amelia}) = \frac{0 \times 2.0 + 0 \times -2.0 + -0.7 \times 1.0 + 0.7 \times 1.0 + 1.3 \times -2.0}{\sqrt{(-0.7)^2 + (0.7)^2 + (1.3)^2} \times \sqrt{(0.2)^2 + (-0.2)^2 + (1.0)^2 + (-2.0)^2}} = -0.65 \]

Similarity score can be pre-computed for all possible pair of items and stored in Item-to-Item Similarity matrix for ease of further calculation. Centred score can be negative or positive due to centred positive or negative rating. Assume nearest neighbour K= 2 for finding rating by user

Figure 3.5: Normalized Ratings and Similarity with a movie
Olivia for 2001:Space, so two similar movie which are rated by Olivia are Fargo and The hunt .. scored 0.24 and - 0.36 respectively.

Step 2: From neighbourhood Items deduce rating:

\[ \text{Rating } R = \sum_{\text{neighborhood}} (\text{Similarity core } \times \text{rating}) \]

After finding cohort items finding rating is straightforward by taking average of cohort items rating, but for improvement in predicted rating weighted average score calculated.

\[ r_{xi} = \frac{\sum N S_{xu} r_{ui}}{\sum N S_{xu}} \]

where

\[ S_{xu} = \text{SimilarityScoreofuserxanduseru} \]
\[ r_{ui} = \text{NormalisedRatingforuseruanditemi} \]
\[ N = \text{NumberofSimilarUser} \]

\[ r_{xi} = \frac{0.24 \times -0.7 + (-0.36 \times -0.7)}{(0.24 - 0.36)} = -0.67 \]

Predicted rating is 0.67 star less than average score of olivia which is 2.7, so final predicted rating is 2 stars. Predicted rating majorly depends upon number of cohort user has been considered, if for asadsbove if k =1 then final rating would have been 3 stars.

Limitation: Neighbourhood collaborative filtering methods has most significant limitation when new user or item has added in rating matrix also called as Cold-Start Problem. As collaborative filtering methods based on past rating, new entry make difficult to recommender system to recommend items effectively due to very a smaller number of similar neighbours. Neighbourhood methods becomes time consuming as computational grows linearly with users and items (Kotu and Deshpande, 2019).

3.5.2 Biased Matrix Factorization

Rating matrix has sparse information about user, item and strength in user-item preference interaction. For each user, rating given is at level of individual items like movies, books or newsfeed. In rating matrix number of items can be from thousands to million and in such scenario preference of user can be in specific categories or features of items for movies it can be
serious or comedy and in case of book it can be fiction or biography. In case of movies user preferred movie directed by Christopher Nolan with strong female lead then he may like other movies directed by Christopher Nolan like Interstellar or Iron Lady which depends upon user specific factors (Kotu and Deshpande, 2019).

Matrix Factorization model takes only rating matrix as its input it doesn’t required any external pre-classified factors. With help of latent factors model explain and generalise rating matrix. Number of factors must be specified for matrix factorization model which can be range from hundreds to thousands and latent factors name are not interpretable sometimes it can be classified genres or can be uninterpretable grouping of items. Once latent factors are decided model learns association of same factors with users as well with items. Strong user-item preference is considered when user and item vector are close to each other where plotted against set of latent factors (ibid.).

Figure 3.6: Items and users in latent factor space, (Kotu and Deshpande, 2019)

Above chart shows user with circles and items with squares and plotted against two factors comedy movie against production type of movie. Items
are plotted as per they express themselves with latent factors and users are plotted according to their preference for two factors. From chart it can inffers that, Amelia prefers No Country for old man than Titanic as they are close to each other in plotted latent factor space. User-item preference dictates by similarity between user and item vectors and it is calculated by dot product of item vector and item vector expressed in latent factors. Latent factors form Rating Matrix are discovered using Matrix Factorization Technique and user-item are mapped against those latent factors(Kotu and Deshpande, 2019).

\[ R \approx P.Q^T \]

Where

\( R \) = Rating matrix  
\( P \) = User with n factors  
\( Q^T \) = Transpose matrix of item with n factors

Rating matrix R can be decomposed in such way that dot product of transpose \( Q^T \) and P yield closely approximate Rating matrix R. The decomposition of Rating matrix greatly reduces the size of matrices and yield original rating matrix with more information including missing rating for each user as well as previously given sparse rating by applying dot product of P and QT [bid].

\( r_{ui}^\wedge = p_u.q_i^T \)

For each item i vector qi measure number of latent factors expressed by item and for every user u, vector pu measure presence of latent factor which possessed by respective item. Approximation for user preference for item given by dot product of pu and qit [ibid.] Objective of Matrix Factorization is to learn vector P and Q for main Rating Matrix therefore P and Q should be learnt in such a way that difference between actual rating and predicted rating should be minimum. Error or Loss function given by,

\[
\min \sum_{(u,i) \in K} (r_{ui}^a - r_{ui})^2
\]

\[
\min \sum_{(u,i) \in K} (r_{ui}^a - p_u.q_i^T)^2
\]

Where K is the known ratings for (u,i) pairs.

Like all machine learning predictive models, Matrix Factorization suffers
from overfitting problem. As key aim of this technique is to predict unknown rating more precisely than know rating, overfitting problem try to minimise by implementing Regularization technique which penalise learning of more complex models to avoid overfitting.

\[
\min \sum_{(u,i) \in K} (r_{ui} - p_u q_i^T)^2 + \lambda(||p_u||^2 + ||q_i||^2)
\]

By minimizing regularised squared error regularisation avoids overfitting in above equation. Tuning parameter \( \lambda \) is used for regularisation ranges from 0 to infinity. Magnitude of learned parameter penalised from regularisation. In real world application bias is present due to some user critically rates movies or some rated reluctantly, even some movies are blockbuster, so they tend to get more high rating. To get effective result considering bias in rating Global Baseline model used in matrix factorisation (Kotu and Deshpande, 2019).

\[
b^u_i = \mu + b_u + b_i
\]

Above equation shows global baseline model. In addition to user-item preference calculated by matrix factorisation, effects of bias can be used for prediction in rating.

\[
r^\wedge_{ui} = \mu + b_u + b_i + p_u . q_i^T
\]

Overall objective function of biased matrix factorisation is shown in below equation where lambdaaa is regularisation parameter, \( \mu \) is global average, \( b_u, b_i \) are user and item biased respectively

\[
\min \sum_{(u,i) \in K} (r_{ui} - (\mu + b_u + b_i + p_u . q_i^T))^2 + \lambda(||p_u||^2 + ||q_i||^2 + b_u^2 + b_i^2)
\]

Stochastic Gradient Descent (SGD) is commonly used to learn factor vector \( p_u \) and \( q_u \). For given number of factors SGD calculates error rate by initialising vector \( P \) and \( Q \) which is difference between actual rating and predicted rating. SGD is an iterative algorithm and stops when there is no meaningful change in error rate while slowly changing value for \( P \) and \( Q \) (ibid.).

Limitation: Biased Matrix factorization cannot explain why recommendation are given as latent factors are decided internally by algorithm with specified number of factors.
### 3.5.3 User profile Content based filtering

Past user-item interaction data is used Collaborative filtering method whereas in content-based or attribute-based recommender system use explicit attributes of items along with past user-item interaction as input. Content-based filtering operated under assumption that user will prefer same item which has same content or properties as past consumed items. User will most likely to get recommendation with same Director or with same star-cast which user has preferred in past with given high rating (Kotu and Deshpande, 2019). Above fig shows framework of content-based recommendation system where it take sparse Rating matrix as well as item profile as input and give full predicated rating matrix or top relevant items to each user. Item profile can be built from item providers or third party meta data providers such as IMDB in case of movies where attributes such as Director of Movie, star-cast to year of released categorised for each movie. Two steps approach is used for predicting user-item rating, first step is to calculate good item profile where each item can be represented as its attribute profile vector and second step is to predict rating using item profile and rating matrix.

User profile content-based filtering is type of content-based filtering in which user-item preference calculated by building addition user feature profile with item profile matrix. Same features in user profile matrix maps user preference with same attributes used in item profile for measuring strength of preference for user to feature. Proximity of user vector to item vector in same attribute space indicates the strength of user item preference as each item and user can be represented in vector form. Centred cosine similarity metric which used in collaborative filtering used to measure preference between user and item. Matrix factorisation and content
based filtering follow similar approach as both methods express rating in specific feature which are same for item and user, But major difference between both approach is that User profile content-based filtering uses known attributes for item profile which are precomputed as passed as input to recommender system where as in Matrix Factorisation technique specified feature from rating matrix used Kotu and Deshpande, 2019

• Building Item profile:
  Item profile is set of features or characteristic about item defined in form of matrix in which each item has been described. Each item is considered as vector with its attributes like for movie it can be Male and Female lead, Director, Genre of movie and attributes can range to thousands. In item profile matrix cell can be filled as Boolean flag with item is associated with respective feature or not. Item profile matrix can be sparse and large.

Figure 3.8: User profile Content based filtering

![Table 11.8 Item Profile](image)

(Kotu and Deshpande, 2019)

Item feature are extracted from third party documents or from item’s seller in form of documents and item’s feature from those documents can be extracted by different mining techniques like TF-IDF (Term frequency inverse document frequency). After building item profile each item describe itself with respective set of features and matrix contains information which item contains which attributes.

• Building user profile
  From combination of known Rating matrix and item profile, user profile has been built. Below figure shows R is rating matrix with m users and n items and I is an item profile matrix with n items and f features, user profile matrix built by combining R and I matrix which will have m users and f factors.
Figure 3.9: User profile from utility matrix and item profile

Figure 3.11 shows Rating matrix and item profile, Rating matrix contains detail for movies which user has liked explicitly, and item profile is Boolean set of items respect to features (Kotu and Deshpande, 2019) Formula used to compute each cell in user profile $U$ is the number of times attributes appears in movies liked by user divided by total number of movies liked by users. From user profile vector for user Amelia $U = 1,0,\ldots,0,0$ and vector for item for Sleepless in Seattle is $I = 1,0,\ldots,0,0,1,0$ which is closely similar to user Amelia so it will get recommended to user Amelia (ibid.).

As recommender system gets more update about user preference, item and user profile continuously gets updated and recommendation suggested accordingly. As content-based filtering depends upon only user itself unlike collaborative filtering where other similar user preference considers. Content-based filtering recommenders effectively solves cold start problem which is major limitation of Collaborative filtering as when new item is added in item matrix it has pre-computed features ready beforehand, so new item recommended to user respectively. But when new user added it still required some user preference otherwise cold start problem affects content-based recommender system (ibid.).
Figure 3.10: Rating Matrix R and Item Profile

<table>
<thead>
<tr>
<th>Table 11.9 Ratings Matrix R</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movie</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>Josephine</td>
</tr>
<tr>
<td>Olivia</td>
</tr>
<tr>
<td>Amelia</td>
</tr>
<tr>
<td>Zoe</td>
</tr>
<tr>
<td>Alanna</td>
</tr>
<tr>
<td>Kim</td>
</tr>
</tbody>
</table>

Figure 3.11: User Profile U

<table>
<thead>
<tr>
<th>Table 11.10 Item Profile I</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movie</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>Fargo</td>
</tr>
<tr>
<td>Forrest Gump</td>
</tr>
<tr>
<td>Queen</td>
</tr>
<tr>
<td>Sleepless in Seattle</td>
</tr>
<tr>
<td>Eye in the Sky</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 11.11 User Profile U</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movie</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>Josephine</td>
</tr>
<tr>
<td>Olivia</td>
</tr>
<tr>
<td>Amelia</td>
</tr>
<tr>
<td>Zoe</td>
</tr>
<tr>
<td>Alanna</td>
</tr>
<tr>
<td>Kim</td>
</tr>
</tbody>
</table>

Kotu and Deshpande, 2019
3.6 **Ensemble based recommendation system**

Central idea of each recommender system is to predict rating for unseen user-item rating through applying its own unique approach. Every recommendation technique has its own strength and limitation, some suffers from overfitting. Ensemble based recommender system combine output of multiple base recommender model into one hybrid recommender if all base models are independent to each other. Ensemble based approach overcomes limitation of single recommender system by improving performance by reducing generalising error (Kotu and Deshpande, 2019).

In Ensemble off-the-shelf algorithm are combined to give robust output such as combining collaborative filtering with content-based filtering. Significance difference are there while implementing single RM model, but basic principal of Ensemble is similar to ensemble data mining techniques such as clustering, outlier analysis or classification. It is not necessary to combine different type of model for building ensemble models one can combine similar models such as collaborative filtering or content-based filtering to obtain more accurate recommendation. In fact, winning entries in Netflix prize contest were both ensemble system called as “Bellkor’s Pragmatic Chaos” and “The Ensemble” (Agarwal, 2016).

Ensemble System can be broadly classified into below categories:

- Weighted: Score of multiple recommender models are combined together into single unified score by calculating weighted averages of scores from each single models. Figure Figure 3.12 illustrates weighted ensemble model. [ibid.](Agarwal, 2016)

Figure 3.12: Weighted Ensemble Model
• Switching: Algorithm switches between various models depending upon the need such as using collaborative filtering when more data available and changing to content-based when needs limit cold start problem.

• Training process of one recommender system is bias by output of previous applied recommender system and output is combined.

• Feature Augmentation: The output of one recommender is given input to another recommender model. This approach shares logic similar to stacking which is used in classification models where output of one classifier used feature to next classifier model.

3.6.1 Weighted Ensemble models

In weighed ensemble, output of multiple recommender models are combined using set weights. \( R_1 \ldots R_q \) be the \( m \times n \) completely specified matrix in which blank rating are predicted by \( q \) different algorithm. For set of weights \( \alpha_1 \ldots \alpha_q \), weighted hybrid model calculates combined prediction matrix \( R_k \) Agarwal, 2016

\[
R = \sum_{i=1}^{q} \alpha_i R_i^\land
\]

To determine optimal weights for each model, it is necessary to check effectiveness of combination of weights \( \alpha_1 \ldots \alpha_n \). In simplest case it is possible to choose weights as \( \alpha_1 = \alpha_2 = \ldots = \alpha_q = 1/q \). It is ideal to give weights to recommender system in differential way such as more weights should be assigned to model which is giving better accuracy [ibid.].

Performance of combined model is evaluated using metrics such as RMSE and MAE. For minimising error value, it is required to combine all model together using liner regression using optimised weights for models. Such linearly regressed models have highly performed in Netflix challenge. RMSE value majorly affects by presence of Noise or outlier values as it largely influenced by outlier data whereas MAE value is robust to outliers. To calculate optimised weights value commonly approach used such as Gradient Descent method [ibid.]

Gradient in terms of individual partial derivates is given as,

\[
\nabla MAE = \left( \frac{\delta MAE(\alpha)}{\delta \alpha} \ldots \frac{\delta MAE(\alpha^-)}{\delta \alpha_q} \right)
\]

Gradient descent is iterative method which keep iterating till value of MAE stops changing for minimal error value. To avoid overfitting regularisation
parameter lambda_used. Ensemble based models gives more accuracy in output compare to individual implemented models (Agarwal, 2016).

### 3.6.2 Evaluation

It is vital to thoroughly evaluate the built model and review model construction whether if it is aiming to solve business objective before final deployment. Process review, evaluation result and determination of next step are key steps taken in evaluation phase (Shearer, 2013).

Like any machine learning prediction models, recommender system also must undergo evaluation step to check whether designed model giving effective accuracy and to check model is not overfitting. Some of already known rating data split into test set and training set for evaluating of models (Kotu and Deshpande, 2019).

\[
RMSE = \sqrt{\frac{\sum_{i=1}^{n} (\hat{r}_{ui} - r_{ui})^2}{n}}
\]

\[
MAE = \frac{\sum_{i=1}^{n} |\hat{r}_{ui} - r_{ui}|}{n}
\]

where
\[
\hat{r}_{ui} - r_{ui} \text{ is difference between predicted and actual ratings.}
\]
\[n \text{ is the occurrence.}
\]

Performance of recommender system most measured with two metrics RMSE (Root Mean Square Error) and MAE (Mean Absolute Error) which measure the delta between predicted rating by recommender system and actual known rating. RMSE penalised for high errors whereas MAE has easy interpretation advantage. Both this metrics must be continuously monitored against any changes in rating matrix for having good accuracy in recommendation system. In this paper, for evaluating each model, known rating matrix split into training set as 95% to 5% as testing set. Accuracy for each model has been calculating using RMSE and MAE value in two different environments RapidMiner and Python.

### 3.6.3 Deployment

Task in not finished just by building machine learning models as model must reach to potential customer to leverage built solution for any business. For reaching to customer it has to deployed or make Live means to deploy it on webserver as Web application. Deployed web application can serve as reporting tool to recommending different items depending upon objective of model (Shearer, 2013).
In this research, the deployment phase is under development for predicting movies with the best accuracy. A web-app will be deployed using PyFlask so the user can use the application in a more intuitive GUI based environment.

4 Analysis

To identify recommender system technique which will give more accuracy, the finding of result will be explained in mentioned below roadmap:

4.1 Exploratory Data Analysis

EDA is conducted on MovieLens dataset to check data imported correctly and different characteristic and correlation present in attributes present in dataset.

- Long tail:

![Figure 4.1: Long Tail](image)

Above plot in Figure 4.1 shows long tail of used Movielens dataset, from plot it can infer that most popular movies received most of the number rating as they are more popular compare to other movies which distributed in plot in long tail part. Movies which received more rating which are small
in numbers are known as “Head”. In recommender system items from long tail are majorly used in giving obscure recommendation to user.

- Histogram of Rating distribution

![Histogram](image)

In dataset movies are rated from 0 to 5 star from different users, above graph shows rating distribution for combining all movies. It is evident that most of the movies rated by users are rated from 3 to 4 stars range. Very few numbers of movies are rated from 0 to 2.5 stars compare to movies which are rated 3 to 4 stars.

- Top rated movies:

Above bar graph depicts top 10 rated movies in Movielens dataset, it is evident that all above top 10 movies are popular due to which they received high rating from all user. First five movies are rated more compare to last five movies in above graph as total rating received to first five movies is reaching almost 1000 stars when from rating matrix.
Figure 4.3: Top Rated Movies

<table>
<thead>
<tr>
<th>Title</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shawshank Red.</td>
<td>1400</td>
</tr>
<tr>
<td>Forrest Gump (1994)</td>
<td>1400</td>
</tr>
<tr>
<td>Pulp Fiction (1994)</td>
<td>1400</td>
</tr>
<tr>
<td>Matrix, The (1999)</td>
<td>1200</td>
</tr>
<tr>
<td>Silence of the Lambs, The (1992)</td>
<td>1100</td>
</tr>
<tr>
<td>Star Wars: Episode...</td>
<td>1000</td>
</tr>
<tr>
<td>Braveheart (1995)</td>
<td>900</td>
</tr>
<tr>
<td>Fight Club (1999)</td>
<td>800</td>
</tr>
<tr>
<td>Schindler's List (1993)</td>
<td>700</td>
</tr>
<tr>
<td>Jurassic Park (1993)</td>
<td>600</td>
</tr>
</tbody>
</table>
• Tree map for genres:

Figure 4.4: Tree Map

Above tree map shows how movies are categorised into different genres. Movies in MovieLens dataset are categorised into different genres and each movie can have more than one genre associated with it. Tree map depicts genre Action, Comedy and Drama movies are majorly included in MovieLens dataset. Movies with genre Musical, Film-noir, Western and War are in less number in dataset.

4.2 Model Result

Recommendation models has been implemented using two different tools first is RapidMiner and second using Spyder for Python coding.
<table>
<thead>
<tr>
<th>Recommender Model</th>
<th>Rapid Miner</th>
<th>Python</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RMSE</td>
<td>MAE</td>
</tr>
<tr>
<td>ItemKNN</td>
<td>0.83</td>
<td>0.627</td>
</tr>
<tr>
<td>BMF</td>
<td>0.89</td>
<td>0.68</td>
</tr>
<tr>
<td>ContentBased</td>
<td>0.854</td>
<td>0.653</td>
</tr>
<tr>
<td>EnsembleWithoutBMF</td>
<td>0.821</td>
<td>0.629</td>
</tr>
<tr>
<td>EnsemblewithBMF</td>
<td>0.818</td>
<td>0.629</td>
</tr>
</tbody>
</table>

**Accuracy Comparison of all Recommender Models**

4.2.1 Using RapidMiner

4.2.1.1 Item-based collaborative filtering

Item-based collaborative filtering implemented in RapidMiner by using central operator Item-KNN from RapidMiner extension Recommenders. Known Rating matrix spit into 95% training set to 5% testing set for performance evaluation. Performance calculated using RMSE and MAE metrics.

4.2.1.2 Biased Matrix Factorization

Table 4.2.1: Result illustrating accuracy for Biased Matrix Factorization

<table>
<thead>
<tr>
<th>Row No.</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.830</td>
<td>0.627</td>
</tr>
</tbody>
</table>

From RapidMiner extension BMF operator is used to implement Biased Matrix Factorization. Rating matrix split into 95% training set and 5% testing set for performance evaluation. RMSE and MAE metrics used to check accuracy.

Table 4.2.2: Result illustrating accuracy for Biased Matrix Factorization

<table>
<thead>
<tr>
<th>Row No.</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.890</td>
<td>0.680</td>
</tr>
</tbody>
</table>
4.2.1.3 User profile content-based filtering
From RapidMiner extension Item-attribute KNN operator is used to implement user profile content-based filtering. Rating matrix split into 95% training set and 5% testing set for performance evaluation. RMSE and MAE metrics used to check accuracy.

<table>
<thead>
<tr>
<th>Row No.</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.854</td>
<td>0.653</td>
</tr>
</tbody>
</table>

Table 4.2.3: Result illustrating accuracy for User profile content-based filtering
4.2.1.4 Ensemble based recommender without using BMF
It is hybrid model built using combining Item-KNN and User-KNN collaborative filtering models. Rating matrix split into 95% training set and 5% testing for performance evaluation. RMSE and MAE used for check accuracy.

Table 4.2.4: Result illustrating accuracy for Ensemble based recommender without BMF

<table>
<thead>
<tr>
<th>Row No.</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.821</td>
<td>0.629</td>
</tr>
</tbody>
</table>

4.2.1.5 Ensemble based recommender system
In this, Ensemble based recommender model User profile content based, Item-KNN and BMF models are combined to get maximum accuracy. RMSE and MAE metric user for performance evaluation.

Table 4.2.5: Result illustrating accuracy for Ensemble based recommender system

<table>
<thead>
<tr>
<th>Row No.</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.818</td>
<td>0.629</td>
</tr>
</tbody>
</table>

4.2.2 Using Spyder for python:
Recommender models which are implemented in RapidMiner are implemented in Python by using Python Scikit library Surprise which is specially designed for implementation of predictive Recommendation system. For implementing all recommendation models using Surprise movie and rating files are merged to create movie-user rating matrix. Known rating matrix then split to 95% training model and 5% for testing implemented models. 4.2.2.1. Item based collaborative filtering. Item based collaborative filtering implemented in Surprise using KNN Basic Algorithm.

4.2.2.1 Item based collaborative filtering
Item based collaborative filtering implemented in Surprise using KNN Basic Algorithm.
Table 4.2.6: Result illustrating accuracy for Item based collaborative filtering using Python

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item KNN</td>
<td>0.9969</td>
<td>0.7802</td>
</tr>
</tbody>
</table>

4.2.2.2 Biased Matrix Factorization
In Surprise library Biased Matrix Factorization is implemented by SVD algorithm and it gives below result

Table 4.2.7: Result illustrating accuracy for Biased Matrix Factorization using Python

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVD</td>
<td>0.8756</td>
<td>0.6781</td>
</tr>
</tbody>
</table>

4.2.2.3 User profile content-based filtering
User profile content-based filtering has been implemented in Surprise as customised recommender System. Result as below

Table 4.2.8: Result illustrating accuracy for User profile content-based filtering using Python

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>content KNN</td>
<td>0.9271</td>
<td>0.7245</td>
</tr>
</tbody>
</table>

4.2.2.4 Ensemble based recommender without using BMF
In this hybrid model, neighbourhood based collaborative filtering combined to improve accuracy of recommendation system. KNN Basic algorithm applied in Surprise for ensemble model, results of model
Table 4.2.9: Result illustrating accuracy for Ensemble based recommender without using BMF

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserKNN</td>
<td>0.9795</td>
<td>0.7581</td>
</tr>
<tr>
<td>ItemKNN</td>
<td>0.9969</td>
<td>0.7802</td>
</tr>
<tr>
<td>Hybrid</td>
<td>0.9202</td>
<td>0.7155</td>
</tr>
</tbody>
</table>

4.2.2.5 Ensemble based recommender system

Ensemble model combines Biased Matrix factorization, Item-KNN collaborative filtering and content-based filtering algorithm. Results is

Table 4.2.10: Result illustrating accuracy for Ensemble based recommender without using BMF

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVD</td>
<td>0.8782</td>
<td>0.6775</td>
</tr>
<tr>
<td>ContentKNN</td>
<td>0.9271</td>
<td>0.7245</td>
</tr>
<tr>
<td>ITEMKNN</td>
<td>0.9969</td>
<td>0.7802</td>
</tr>
<tr>
<td>Hybrid</td>
<td>0.9099</td>
<td>0.7093</td>
</tr>
</tbody>
</table>

5 Discussion

5.1 Data Overview

On conducting Exploratory Data Analysis on MovieLens dataset it reveals pattern exist in dataset for user rating for movies. From graph it shows that most popular movies are rated more number of times.

5.2 Model Result

Comparing performance of each recommendation model implemented individually and combined in ensemble model evaluated on metrics RMSE and MAE to check whether ensemble model outperform individually implemented model in accuracy. For implementation of recommender model in Python Scikit Library called as Surprise used which is specially designed to implement predictive recommender models.
5.2.1 Item based collaborative filtering

1. RapidMiner

Movies and rating dataset merged to for movie user rating matrix as part of data preparation. Feature selection done by select attribute operator where ‘timestamp’ attribute dropped from matrix as it is not useful for modelling. Set Role operator is used for declaring ‘user identification’ and ‘item identification’ and label from user rating matrix. Rating matrix split into 95% training and 5% testing set using Split Data operator. Central operator used Item-kNN for recommendation using nearest neighbour $K = 10$ and setting learning and regularisation parameter at default value given by RapidMiner and cosine similarity metric used for calculating item similarity. Performance measured using performance operator.

Table 5.2.1: Result Item based collaborative filtering

<table>
<thead>
<tr>
<th>Row No.</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.830</td>
<td>0.627</td>
</tr>
</tbody>
</table>

Item-kNN gives result with accuracy as RMSE value 0.830 stars and
MAE value 0.627 stars for MovieLens dataset. In general, it means predicted rating are 0.627 stars away from actual rating.

2. Using Python Code

In Surprise Library, Neighbourhood collaborative filtering implemented using defined class as KNNBasic. Both UserKNN and ItemKNN can be implemented using same class, just parameter to method call need to be change respectively. For implementing ItemKNN below is code snippet.

```python
# Item-based KNN
ItemKNN = KNNBasic(sim_options = {'name': 'cosine', 'user_based': False})
evaluator.AddAlgorithm(ItemKNN, "Item KNN")
```

Figure 5.2: Code snippet Item based collaborative filtering

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ItemKNN</td>
<td>0.9969</td>
<td>0.7802</td>
</tr>
</tbody>
</table>

RMSE and MAE given by python code is 0.6669 stars and 0.7802 stars respectively. Item-KNN implemented in RapidMiner perform better than Python code implementation for MovieLens dataset.

For every recommender system with good performance, top recommendation of item is important, below are Top 10 movies recommended by Item-KNN algorithm using python code for test user id 85.
### Table 5.2.3: Top 10 recommendation

<table>
<thead>
<tr>
<th>ID</th>
<th>Movie Name and Year</th>
<th>Ratings</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Best Men(1997)</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>Galaxy Of Terror(Quest)(1981)</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>Looker(1981)</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>Andriod(1982)</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>Alien Contamination(1980)</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>Master of the Flying Guillotine(1975)</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>Priceless(HOrs de prix)(2006)</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>Lassie (1994)</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>Act of Valor (2012)</td>
<td>5</td>
</tr>
<tr>
<td>10</td>
<td>Honey(2003)</td>
<td>5</td>
</tr>
</tbody>
</table>

### 5.2.2 Biased Matrix Factorization

1. RapidMiner

For implementing BMF in RapidMiner central operator used is BMF (Bias Matrix Factorization), data has been prepared like Item-KNN process with same merging, selecting role and splitting data before applying to BMF operator. Number of features selected as 12 in BMF operator, Bias, Learning operator for Gradient descent and regularization parameter are used as default value of 0.0001, 0.005 and 0.015 respectively. After training apply model with testing data and measured performance using performance operator.

### Table 5.2.4: Result of Biased Matrix Factorization

<table>
<thead>
<tr>
<th>Row No.</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.890</td>
<td>0.680</td>
</tr>
</tbody>
</table>
Factorisation gives accuracy of 0.890 stars RMSE and 0.680 stars MAE which is poor to compare to Item-KNN accuracy previously implemented in RapidMiner.

2. Python Code

Biased Matrix factorisation implemented in Surprise library using class as SVD in which one can implement biased as well as unbiased matrix factorisation recommendation model for prediction of movie rating. Below code snippet of implementation.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVD</td>
<td>0.8756</td>
<td>0.6781</td>
</tr>
</tbody>
</table>

RMSE and MAE for BMF is 0.8756 stars and 0.6781 stars respectively. Result from python code is better than results obtained from implementation in RapidMiner.
For BMF top 10 recommenders for user id 85 is below.
Table 5.2.6: Top 10 recommendation

<table>
<thead>
<tr>
<th>ID</th>
<th>Movie Name and Year</th>
<th>Ratings</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Double Indemnity (1944)</td>
<td>4.755</td>
</tr>
<tr>
<td>2</td>
<td>Harold and Maude</td>
<td>4.599</td>
</tr>
<tr>
<td>3</td>
<td>Wallace and Gromit</td>
<td>4.570</td>
</tr>
<tr>
<td>4</td>
<td>Old Boy(2003)</td>
<td>4.569</td>
</tr>
<tr>
<td>5</td>
<td>Brazil(1985)</td>
<td>4.544</td>
</tr>
<tr>
<td>6</td>
<td>Aliens(1986)</td>
<td>4.528</td>
</tr>
<tr>
<td>7</td>
<td>Rosemary’s Baby</td>
<td>4.526</td>
</tr>
<tr>
<td>8</td>
<td>GoodFellas (1974)</td>
<td>4.5221</td>
</tr>
<tr>
<td>9</td>
<td>chinatown</td>
<td>4.514</td>
</tr>
</tbody>
</table>

5.2.3 User profile content-based filtering

1. RapidMiner

In User-profile content-based filtering data is prepared similar to previous models such as create rating matrix by merging rating and movies files, dropping timestamp attribute, role has been selected in Set Role operator and then data was split for training and test set. Item profile has been created for MoviLens dataset, each item can have multiple genre, so matrix of movie and respective genre has to be created which is created by Item profile sub process as below, Text mining process is used to create item profile, Text to attribute using text mining operator transform concatenated multiple genre value into multiple attributes. Rename replace operator assigned naming to each column. De-Pivot operator convert column information to rows due to which genre in each movie is now one distinct column. Output of De-pivot has both negative as well as positive values, negative values are filtered out and set role operator used for selecting attribute as item identification and user identification and label. Value of
Figure 5.5: Process of User profile content-based filtering

Figure 5.6: Process of Item profile generation
neighbour K selected as 10 and modelled is trained and performance measured on test set.

RMSE and MAE value of user profile content-based filtering are

<table>
<thead>
<tr>
<th>Row No.</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.854</td>
<td>0.653</td>
</tr>
</tbody>
</table>

0.854 stars and 0.653 stars respectively.

2. In Python Code

In Surprise user profile content has to be customised implement inheriting main class called AlgoBase from Surprise library.

```python
c = ContentKNNAlgorithm()
evaluator.AddAlgorithm(contentKNN, "ContentKNN")
print('... END OF KNN ADDITION')
```

Figure 5.7: Code snippet of User profile content-based filtering

Performance obtain from implementation of user profile content-based recommendation algorithm RMSE is 0.9271 stars and MAE is 0.7245 stars which is not as better when implemented in RapidMiner.

Top 10 movies recommended to test user 85 are
Table 5.2.9: Top 10 recommendation

<table>
<thead>
<tr>
<th>ID</th>
<th>Movie Name and Year</th>
<th>Ratings</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Pinochhio(1940)</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>James and Giant Peach(1996)</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>Wizard of Oz(1939)</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>Escape to Witch Mountain(1975)</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>winne the Pooh and the Blustery Day(1968)</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>Three Caballeros(1945)</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>Sword in the Stone(1963)</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>Pete’s Dragon (1977)</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>Bedknobs and Broomsticks (1971)</td>
<td>5</td>
</tr>
<tr>
<td>10</td>
<td>Alice in Wonderland(1951)</td>
<td>5</td>
</tr>
</tbody>
</table>

5.2.4 Ensemble Based recommender System without BMF

1. RapidMiner

Hybrid recommenders are implemented by combining 2 or more independent models together by giving appropriate weights to each algorithm. This hybrid algorithm have two models User-KNN and Item-KNN collaborative filtering and combined using model combiner where both models have given same weights and output given to model to test accuracy on test data which was split from MovieLens dataset with nearest neighbour value $k = 10$. 
2. Using Python Code Using Surprise library ensemble model was built using KNNBasic class which provide both User-KNN and Item-KNN algorithm with passing different argument to function call. Both algorithms have given equal weight equal to one. From above result it is evident that Hybrid model combining UserKNN and ItemKNN perform better in accuracy. RMSE and MAE value for hybrid is 0.9202 stars and MAE is 0.7155 stars. Again, results in RapidMiner for same hybrid algorithm for MoviLens dataset are better.
Figure 5.9: Process of Ensemble Based recommender System without BMF

```python
# user-based KNN
UserKNN = KNNBasic(sim_options = {'name': 'cosine', 'user_based': True})
#evaluator.AddAlgorithm(UserKNN, "User KNN")

# Item-based KNN
ItemKNN = KNNBasic(sim_options = {'name': 'cosine', 'user_based': False})
#evaluator.AddAlgorithm(ItemKNN, "Item KNN")

#Combine them
Hybrid = HybridAlgorithm([ItemKNN, UserKNN], [1,1])

evaluator.AddAlgorithm(UserKNN, "UserKNN")
evaluator.AddAlgorithm(ItemKNN, "ItemKNN")
evaluator.AddAlgorithm(Hybrid, "Hybrid")
```

than algorithm implemented in Python.

Below is list of top ten recommended movies recommender to test user id 85 by hybrid algorithm.
<table>
<thead>
<tr>
<th>ID</th>
<th>Movie Name and Year</th>
<th>Ratings</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Galaxy Of Terro(1981)</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>Andriod (1982)</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>Alien Contamination(1980)</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>Rivers and Tides(2001)</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>Advise and Consent</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>Shogun Assassin(1980)</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>Sonatine(Sonachine)(1963)</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>Palindromes (2004)</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>Awfully Big Adventure, An(1955)</td>
<td>5</td>
</tr>
<tr>
<td>10</td>
<td>What Happened Was (1944)</td>
<td>5</td>
</tr>
</tbody>
</table>

5.2.5 Ensemble Based recommender System

1. RapidMiner

Ensemble based recommendation system combined using three algorithm, Bias Matrix Factorization, ItemKNN collaborative filtering and user profile content-based filtering. All three models combined using model combiner with applying same weight. After training models, test data is applied to model for checking performance from performance operator.

RSME and MAE value for ensemble recommendation model are 0.818 stars and 0.629 stars which are best measured metrics in all RapidMiner implemented individual model as well as Ensemble model.
Figure 5.10: Process of Ensemble Based recommender System

<table>
<thead>
<tr>
<th>Row No.</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.818</td>
<td>0.629</td>
</tr>
</tbody>
</table>

without BMF.

2. Using Python code
Ensemble model using Surprise built by combining ItemKNN, user profile content-based content based filtering and Biased Matrix Factorisation with assigning equal weight to all three models. Below is code snippet for python code for ensemble model.

Table 5.2.13: Result of Biased Matrix Factorization

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVD</td>
<td>0.8782</td>
<td>0.6775</td>
</tr>
<tr>
<td>ContentKNN</td>
<td>0.9271</td>
<td>0.7245</td>
</tr>
<tr>
<td>ItemKNN</td>
<td>0.9969</td>
<td>0.7802</td>
</tr>
<tr>
<td>Hybrid</td>
<td>0.9099</td>
<td>0.7093</td>
</tr>
</tbody>
</table>
Figure 5.11: Code snippet of Ensemble Based recommender System

```python
SVD = SVD(n_factors = 10)
#evaluator.AddAlgorithm(SVD, "SVD")

# Content based KNN
contentKNN = ContentKNNAlgorithm()
#evaluator.AddAlgorithm(contentKNN, "ContentKNN")

# Item-based KNN
ItemKNN = KNNBasic(sim_options = {'name': 'cosine', 'user_based': False})
#evaluator.AddAlgorithm(ItemKNN, "Item KNN")

#Combine them
Hybrid = HybridAlgorithm([SVD, ItemKNN, contentKNN], [1,1,1])

#evaluator.AddAlgorithm(SVD, "SVD")
#evaluator.AddAlgorithm(contentKNN, "contentKNN")
#evaluator.AddAlgorithm(ItemKNN, "ItemKNN")
#evaluator.AddAlgorithm(Hybrid, "Hybrid")
```

Above are result for all individual model as well as combining all three models. SVD is giving best RMSE and MAE value while implementation in Python. Still, ensemble model using Biased Matrix factorisation is outperforming Ensemble model built without BMF.

Below are top 10 recommended movies for test user 85, recommended by using Ensemble recommender system.
5.3 **Comparison of Algorithm**

To find solution of research question “Can Ensemble model using Biased Matrix factorisation can outperform all separately implemented model as well as Ensemble model with UserKNN and ItemKNN” comparison of all models accuracy performance using RapidMiner and Python conducted as follows.

From above table it is evident that, result from RapidMiner and Python using Surprise library for MovieLens dataset are different for same implemented recommendation model.

For predicting user-item preference accurately implemented recommender system should deliver as much less as possible RMSE and MAE value. When implemented in RapidMiner Ensemble model with BMF, ItemKNN and user profile content-based filtering gives best result of RMSE of 0.818 stars and 0.629 stars MAE whereas Ensemble model without BMF gave RMSE of 0.821 stars and 0.629 stars whose RMSE is marginally poorer.
to Ensemble model with BMF but MAE value of both ensemble model is similar.

When recommendation models are implemented individually in RapidMiner their accuracies are comparatively low with Ensemble models. When implemented independently itemKNN accuracy was better than BMF and user profile content-based filtering model. Biased Matrix factorisation accuracy was worst compared to all other recommendation models implanted independently which was RMSE of 0.890 stars and MAE of 0.680 stars.

When recommendation models are implemented in Python using Surprise library single implemented algorithm SVD outperformed all other recommendation algorithm including Ensemble based model using SVD. But when compared to hybrid model implemented with SVD outperformed ensemble model which did not have SVD and accuracy was 0.9099 stars RMSE and 0.7093 stars MAE compared to 0.9202 RMSE and 0.7155 stars.

Using Python, for every implemented algorithm Top 10 movies has been recommended to test user and each algorithm recommend different movies to same user respect to implemented algorithm. When comparing for same test user 85 recommended movies are different for both ensemble models, Top first movie recommended by Ensemble model using BMF was ”Best Man (1997)" surprisingly was not present Ensemble model without BMF. But two same movies was recommended by same user.

Previous referred research was focused on improving the accuracy of the recommender system using ItemKNN and UserKNN. Below can be a contribution from this research paper.

1. When implemented each model individually accuracy is lower than the ensemble model.

2. Ensemble model with BMF outperform in accuracy ensemble model without BMF.

5.4 Limitation

Limitation of this research, in this dataset, could have added more attributes such as demographic or implicit data so recommendation could be done at a more granular level.
Python code implemented using Surprise SciKit library which has its internal working where more advanced parameters setting could have lead with the same accuracy performance given by RapidMiner.
6 Conclusion

The proposed research started with a literature survey through which an idea to create an ensemble based recommendation system incepted. While studying literature surveys it seems that there is a need to build a recommendation system using BMF model in ensemble based recommendation system to improve accuracy. Comparing the result of the recommendation system implemented individually and in ensemble way inferred that accuracy is better in the ensemble based model with BMF. Each model implemented using RapidMiner and Spyder tool for measuring accuracy. RapidMiner results are more promising than Spyder where RMSE and MAE accuracy scores are 0.821 stars and 0.629 stars whereas in Spyder its 0.9099 stars and 0.7093 stars respectively. Comparing RMSE and MAE score for each recommendation model, it is evident that ensemble based recommendation model outperforms other models with better accuracy metrics. Therefore for predicting item recommendation in the business world, it is important to build a recommendation model using an ensemble based recommendation model to improve accuracy and aims to engage the customer with it. MovieLens 100K dataset has been analyzed using RapidMiner and Spyder for improving enhancing accuracy of recommender system and Ensemble based recommendation system gave the best accuracy compared to other techniques.

6.1 Future Work

As future work, the accuracy of the recommendation system can be evaluated using more metrics such as Hit rate, Coverage, diversity and Novelty addition to traditional metrics RMSE and MAE. Evaluating recommenders system on the above metrics will lead to building not only accurate but also more engaging developed recommender system. As proposed recommendation system built offline, it is very important to check accuracy with A/B testing by deploying it to online with proper real-world user interaction.

7 Appendices

This document will guide you through the contents of the Artifacts and the essential steps to execute the Python code in Spyder and implement RapidMiner processes for dissertation project titled "Performance Evaluation of Ensemble based Recommendation System using Biased Matrix
Factorization and Content-Based Filtering”.

7.1 Dataset

- links.csv - Id’s for IMDB and TMDB movies database for respective movie.
- movies.csv - List of movies in dataset.
- rating.csv - User rating for each movie.
- tags.csv - Tags denoted by user for each movie.

7.2 Models

7.2.1 RapidMiner Processes

- Item k-NN_recommendation - Process of Item-KNN model.
- factorization_BMF - Process of Biased Matrix Factorisation.
- Content based - attribute - Process of User profile content-based.
- Hybrid recommenders_without_BMF - Process of Ensemble model without BMF.
- Hybrid recommenders - Process of Ensemble model using BMF.

7.2.2 Using Python

- collaborativeFiltering - Python code files for implementation of ItemKNN RS.
- contentBased - Python code files for implementation of user profile content based filtering.
- matrixFactorization - Python code files for implementation of Biased Matrix Factorisation.
- hybrid_without_bmf - Python code files for implementation of Ensemble model without BMF.
- hybrid - Python code files for implementation of Ensemble model using BMF.
7.2.3 Readme file

Readme is steps to be follow file to setup RapidMiner and Python Environment for implementation of each Recommendation System.

References


Chen, Rui et al. (2018). “A Survey of Collaborative Filtering-Based Recommender Systems: From Traditional Methods to Hybrid Methods Based on Social Networks”. In: IEEE Access 6.5, p. 20. DOI: [10.1109/ACCESS.2018.2877203](http://dx.doi.org/10.1109/ACCESS.2018.2877203)

Clemente, Maria Laura (2008). “Experimental Results on Item-Based Algorithms for Independent Domain Collaborative Filtering”. In: 2008 International Conference on Automated Solutions for Cross Media Content and Multi-Channel Distribution 13, p. 6. DOI: [10.1109/AXMEDIS.2008.33](http://dx.doi.org/10.1109/AXMEDIS.2008.33)


Gemulla, Rainer et al. (2011). “Large-Scale Matrix Factorization with Distributed Stochastic Gradient Descent”. In: Institute of Informatics, Warsaw University 15, p. 6.


Peiliang, Ye Hanmin; Zhang Qiuling; Huang (2016). “Collaborative Filtering Algorithm Research Based on Matrix Factorization and Muti-path Trust Degree Fusion”. In:


Xia, Shuang et al. (2010). “Optimizations for item-based Collaborative Filtering algorithm”. In: *Proceedings of the 6th International Conference on Natural Language Processing and Knowledge Engineering(NLPKE-2010)* 6.10, p. 5. DOI: [10.1109/NLPKE.2010.5587833](https://doi.org/10.1109/NLPKE.2010.5587833).

Xu, Guangxia et al. (2019). “A Collaborative Filtering Recommendation Algorithm Based on User Confidence and Time Context Guangxia User-Based Collaborative Filtering”. In: 4.9, p. 5.

Zhao, Lin and Bo Xiao (2018). “Matrix Factorization Based Models Considering Item Categories and User Neighbors”. In: *8th International Symposium on Computational Intelligence and Design (ISCID)* 8.5, p. 20. DOI: [10.1109/ISCID.2015.154](https://doi.org/10.1109/ISCID.2015.154).