CS 550: Movie Recommendation System using Collaborative Filtering and Content-based Filtering

Abhishek Nayak (an897), Pranathi Vaddela (pv250), Aditya Kaushik Jonnavittula (aj918)
Rutgers University, New Brunswick
an897@scarletmail.rutgers.edu,pv250@scarletmail.rutgers.edu,aj918@scarletmail.rutgers.edu

ABSTRACT

This project focuses on exploring two methods for movie recommendation systems: Content-based Filtering and Collaborative Filtering. The benefits of recommendation systems include increased sales/conversion and user satisfaction. We used the MovieLens dataset to train and test our models, and utilized techniques such as SVD, KNN, and Content-based Filtering. We analyzed the dataset, compared the benefits and limitations of each method, and present a detailed report of our findings.

KEYWORDS

Recommendation systems, Collaborative Filtering, Content-based Filtering, Matrix Factorization, SVD, KNN

1 INTRODUCTION

The recommendation systems are algorithms aimed at suggesting relevant items to users. There are two main elements in every recommendation system: users and items. In this project we have implemented a movie recommendation system. In a movie recommendation system,the system generates movie predictions for its users, while items are the movies themselves. The primary goal of a movie recommendation system is to filter and predict only those movies that a corresponding user is most likely to watch.

We have trained our models on the MovieLens dataset. To begin with the selection of the models, we first performed some exploratory data analysis.

2 RELATED WORK

Recommendation systems have been extensively studied in the research community, and numerous methods have been proposed to improve their performance. In this section, we briefly review some related work in the field of movie recommendation systems.

One popular method for recommendation systems is Collaborative Filtering [6], which leverages user-item interactions to make predictions. CF can be further divided into two categories: user-based and item-based CF. User-based CF predicts a user's rating for an item based on the ratings of other similar users, while item-based CF predicts a user's rating for an item based on the ratings of other similar items. Although CF has shown promising results, it suffers from the cold-start problem and sparsity problem, which can affect its performance when dealing with new users or items with few ratings.

Content-based Filtering [7] is another popular method for recommendation systems, which utilizes the content information of items to make predictions. CBF predicts a user's rating for an item based on the similarity between the content features of the item and the user's preferences. CBF has shown to be effective in dealing with the cold-start problem and can provide personalized recommendations for users with specific interests. However, CBF may suffer from the overspecialization problem, where users may receive recommendations that are too similar to their past preferences.

Matrix Factorization [1] is another widely used technique for recommendation systems. MF models the user-item rating matrix as the product of two low-rank matrices, which can capture the latent factors underlying the interactions between users and items. MF has shown to be effective in dealing with the sparsity problem and can provide accurate recommendations even for users with few ratings. However, MF may suffer from the scalability problem when dealing with large datasets.

In this project, we explore both CF and CBF methods and compare their benefits and limitations. We also utilize Matrix Factorization techniques such as SVD to improve the performance of our models.

3 PROBLEM FORMALIZATION

We are going to overview the two major paradigms of recommender systems : collaborative and content based methods. The next two sections will then describe various methods of collaborative filtering, such as user-user, item-item and matrix factorization. The following section will be dedicated to content based methods and how they work. Finally, we will discuss how to evaluate a recommender system.

3.1 Content Based Filtering

Content-based filtering is a type of recommendation system that relies on the content or features of the items being recommended. In this approach, the recommendation problem is transformed into either a classification or regression problem, depending on whether we are predicting whether a user likes an item or the rating given by a user to an item. We build and learn a model for each item based on user features, trying to answer the question of the probability for each user to like the item. This method is item-centred, meaning modelling, optimization and computation can be done "by item."

Conference'17, Washington, DC, USA

DOI: 10.1145/nnnnnnn.nnnnnnn

However, this method can be less personalized than other methods as the interactions used to train the model come from all users, and even if they have similar characteristics, their preferences can be different. Content filtering, on the other hand, enables the system to understand user preferences by utilizing user/item profiles. It avoids recommending items that do not match the user's preferences. To implement a content-filtering recommendation system, the importance of each feature is reflected using a term frequency-inverse document frequency (TFIDF) technique. The sum product of importance weights and user preferences towards different features are then calculated, and movies are sorted based on this value to suggest the top N candidates as recommendations.

3.2 Collaborative Based Filtering

Collaborative methods for movie recommender systems are techniques that rely solely on past interactions between users and movies to generate new recommendations. These interactions are typically stored in a matrix known as the "user-movie interactions matrix," where each row represents a user and each column represents a movie.

The key concept behind collaborative methods is that these past user-movie interactions contain enough information to identify similar users and/or similar movies and make predictions based on these estimated similarities. Collaborative filtering algorithms are generally divided into two subcategories: Memory-based and Model-based approaches.

Memory Based: Memory-based approaches directly utilize the values of recorded interactions, without assuming any model, and are primarily based on nearest neighbor searches. For instance, the system can identify the most similar users to a given user and recommend the most popular movies among these similar users.

- **User - User:** [2]In a movie recommendation system using the user-user method, the system identifies users with similar movie preferences to the target user and suggests movies those users have enjoyed. This is done by representing each user as a vector of their interactions with movies and calculating a similarity score between the target user and every other user. The system then selects a subset of the most similar users and suggests highly-rated movies they have watched but the target user has not. This method is called "user-centered" as it evaluates the similarity between users based on their interactions with movies.

- **Item-Item:** [3]Item-item recommendation is a method that recommends movies to a user based on their positive interactions with other movies. The method finds similar movies to those the user already liked, based on how other users interacted with those movies. The method represents movies based on user interactions and evaluates distances between movies. To make a recommendation, the method selects the user's favourite movie and finds similar movies that the user has not yet watched. Alternatively, the method can consider multiple favourite movies to recommend movies that are similar to several of them.

Model Based: Model-based approaches, on the other hand, assume the existence of an underlying generative model that explains the user-movie interactions and try to uncover it in order to generate new predictions. These methods can include matrix factorization, clustering, and deep learning models, among others. Overall, collaborative methods are popular and effective techniques for building movie recommender systems due to their ability to leverage past user-movie interactions to generate personalized recommendations.

- **Matrix factorization** is a popular technique used in movie recommendation systems. The main idea is that there is a low-dimensional latent space of features that can be used to represent both users and movies. The interaction between a user and a movie can be obtained by computing the dot product of their corresponding dense vectors in this latent space. The system learns these features on its own without the need for explicit feature engineering, which is a key advantage over content-based approaches. The resulting feature vectors may not have an intuitive interpretation, but they are effective in capturing similarities between users and movies. This enables the system to recommend movies to a user based on the preferences of similar users and the characteristics of similar movies in the latent space.

4 THE PROPOSED MODEL

We propose a comparative study of several recommendation system architectures. For Memory based we used K-Nearest Neighbors (KNN) and for model based we tried to calculate the RMSE and MAE by comparing Singular Value Decomposition (SVD), Non-negative Matrix Factorization (NMF), CoClustering, SlopeOne.

The goal of this project [5] is to compare Content-based filtering and Collaborative Filtering approaches and find which one produces the best results. As a part of this process we plan to demonstrate rating prediction and generating N-predictions (in our case N = 10), for each user and then calculate several metrics to gauge the performance of the said approaches. The metrics we used to study the performance of each of these models are given below.

4.1 RMSE

RMSE or Root Mean Squared Error is an performance metric that indicates the absolute fit of the model to the data. That implies it tells us how closely do our predictions match with the real data. It can be calculated by taking the square root of the variance of the residuals. Mathematically, it is represented as:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{d_i - f_i}{\sigma_i}\right)^2} \tag{1}$$

4.2 MAE

Mean Absolute Error (MAE) is a measure that indicates the average magnitude of errors in our set of predictions, irrespective of the direction. The Mean Absolute Error is a linear metric, i.e. all the individual differences are weighted equally in the average. The formula to calculate MAE for a set of predictions is as shown:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - x_i| \tag{2}$$

4.3 Precision

Precision at k is the proportion of recommended items in the top-k set that are relevant. This measure, tests the ability of the model to make relevant predictions. The formula to calculate precision is:

$$\text{Precision} = \frac{\text{No of relevant recommendations}}{\text{No of total recommendations}} \tag{3}$$

4.4 Recall

Recall at k is the proportion of relevant items in the top-k set. We can always try to maximize this measure. The formula to calculate recall is:

$$\text{Recall} = \frac{\text{No of relevant recommendations}}{\text{No of total relevant items}} \tag{4}$$

4.5 F-measure

The F-measure is a measure of the prediction model's accuracy. We can use the precision and recall to calculate this score. The formula to calculate F-measure is:

$$F\text{-}measure = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{5}$$

5 EXPERIMENTS

We did the following steps to achieve the goal with the processes mentioned above

5.1 Data

The dataset that we considered for our project is from MovieLens, a recommendation service. For this project, we are using the smaller 100k version dataset, for the sake of simplicity and to reduce computations. It contains 100,836 ratings and 3683 tag applications across 9742 movies. Users were selected at random for inclusion and all the selected users had rated at least 20 movies. The dataset as a whole consisted of movies.csv, tags.csv, ratings.csv and links.csv.

Figure 1: Word Cloud Generated from the Dataset

5.2 Data Exploration and Analysis

We performed data exploration to get a gist of the dataset initially. As it is evident from the histogram plotted in Figure 3, a lot of the movies in the dataset were from Drama and comedy genres, although, there were a few from the western and documentary genres too. Further, in Figure 4, we have plotted the ratings and a majority of the movies were rated close 4 stars. The 'movies' and the 'ratings' tables were merged based on the movieID field and we then added a column for the number of ratings each movie got in order to determine the popularity.

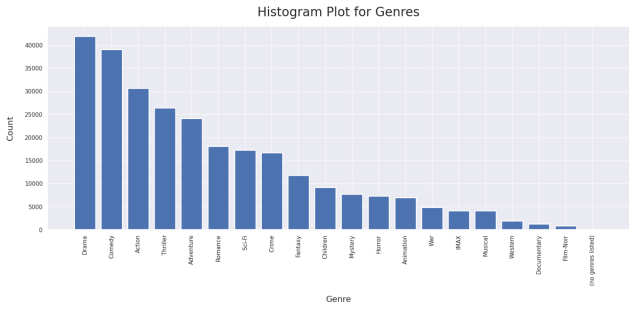

Figure 2: Histogram for different genres

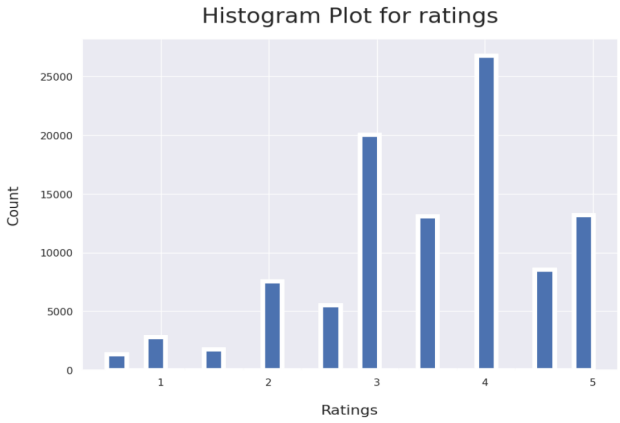

Figure 3: Histogram for Ratings

5.3 Implementation

Content Based:

To implement a content-filtering recommendation system, We utilized TFIDF to reflect the importance of each genre in any movie. And then We calculated the sum product of the importance weights and users' preferences towards different genres (given in user profile). Based on the sum-product, we could simply sort movies and suggest the users the top N candidates as the recommendations.

Collaborative Based:

We used KNN for content for Memory based and SVD for Model based. For finding the best possible Model, we tried to calculate the RMSE and MAE Values by comparing Singular Value Decomposition (SVD), Non-negative Matrix Factorization (NMF), CoClustering, SlopeOne. We got the lowest value of RMSE for SVD, hence we opted for SVD. Comparing the values of RMSE and MAE for KNN and SVD, we got these values:

The algorithmic steps for discovering new recommendations are as follows:

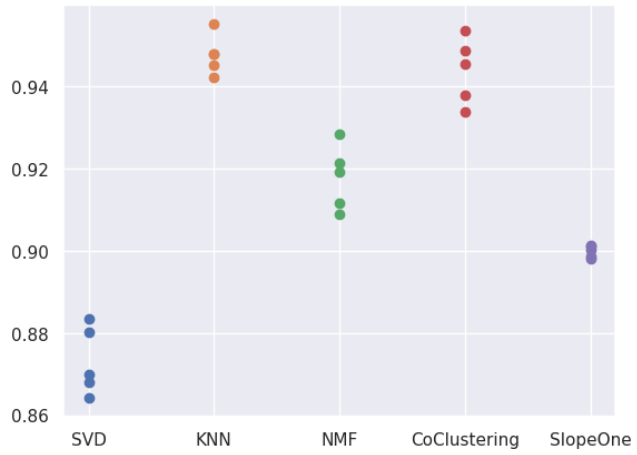

Figure 4: Comparing RMSE and MAE values

Table 1: Comparison of RMSE and MAE for Memory Based (KNN) and Model Based (SVD)

Model	RMSE	MAE
Memory Based (KNN)	0.96	0.74
Model Based (SVD)	0.88	0.67

Figure 5: Network Graph between Users and Movies

(1) For each user U:
   (a) Select all movies that the user has not watched in the training set.
   (b) Predict scores for these movies.
   (c) Arrange the predicted scores in descending order.
   (d) Pick the top 10 movies.
   (e) Calculate metrics for Precision, Recall, F-Measure, and NDCG.
(2) Calculate the Average Precision, Recall, and F-Measure Scores.

Table 2: Comparison of Precision, Recall, F-Score, and NDCG for SVD and KNN Models

Model	Precision	Recall	F-Score	NDCG
SVD	0.68	0.41	0.51	0.81
KNN	0.55	0.38	0.43	0.75

5.4 Advantages and Disadvantages

Advantages:
Collaborative filtering is a highly personalized and user-centric recommendation approach that is based on the user's behavior and preferences. It can handle a large number of users and items, making it scalable to large datasets, and can provide recommendations even for new or unpopular items. Furthermore, it does not require any domain knowledge or metadata about the items being recommended.

Disadvantages:
Collaborative filtering suffers from the cold start problem, which makes it difficult to provide recommendations for new users or items that do not have sufficient data. Additionally, it requires a significant amount of user-item interaction data to provide accurate recommendations, which can result in data sparsity issues.

6 CONCLUSIONS AND FUTURE WORK

From the above results we saw that User Based Collaborative filter gave us the least precision when recommending movies. The SVD performed better in both predicting the ratings of the movies as well as recommending movies to a user. We further analyzed that the movies that were being recommended to the users were biased based on their popularity. For example, if a user is interested in comedy movies then the system recommended a movie from the most popular movie that had comedy genre in it.

As a future scope, we aim to utilize more information from the dataset beyond just genres for personalized recommendations. Use of advanced recommendation techniques such as model-based and hybrid filtering could be incorporated. Further, the "filter bubble" could be avoided by recommending a balance of similar and diverse movies. Another improvement could be recommending movies based on recent events, considering the current trends, and recommending related movies.

Technical Methods for achieving privacy in Recommendation System

Differential privacy, Federated learning, and Bloom filters are some of the techniques that can be used to enhance the privacy of recommender systems.

Differential privacy is a framework for measuring the privacy of an algorithm or system. In the context of recommender systems,

it can be used to add noise to user data or recommendation results, ensuring that individual user preferences cannot be inferred. Federated learning allows multiple parties to collaborate on the training of a machine learning model without sharing their data directly. In the context of recommender systems, it can be used to train a model using data from multiple users without requiring those users to share their data directly. [4]Bloom filters can be used to anonymize user data by hashing the data and adding it to the filter. When making recommendations, the recommender system can query the Bloom filter to determine whether a particular item is in the user's set of preferences or not, without actually seeing the user's preferences directly.

ACKNOWLEDGEMENT

We would like to thank Prof. Yongfeng Zhang for giving us the opportunity to work on this project which in turn, helped us in applying the knowledge that we gained through this course and also in gaining hands-on experience.

REFERENCES

[1] 2009. (PDF) matrix factorization techniques for Recommender Systems (2009): Yehuda Koren: 7690 citations. (Aug 2009). https://typeset.io/papers/matrix-factorization-techniques-for-recommender-systems-1ygvnzk1c8

[2] Paul Resnick ATamp;T, Paul Resnick, Atamp;t, Hal R. Varian Dean of the School of Information Management, Systems, Hal R. Varian, Dean of the School of Information Management, Systems, and Other MetricsView Article Metrics. 1997. Recommender Systems. (Mar 1997). https://dl.acm.org/doi/10.1145/245108.245121

[3] Badrul Sarwar GroupLens Research Group/Army HPC Research Center, Badrul Sarwar, GroupLens Research Group/Army HPC Research Center, George Karypis GroupLens Research Group/Army HPC Research Center, George Karypis, Joseph Konstan GroupLens Research Group/Army HPC Research Center, Joseph Konstan, John Riedl GroupLens Research Group/Army HPC Research Center, John Riedl, Hong Kong Univ. of Science, Technology, and et al. 2001. Item-based collaborative filtering recommendation algorithms: Proceedings of the 10th International Conference on World Wide Web. (May 2001). https://dl.acm.org/doi/10.1145/371920.372071

[4] Burton H. Bloom Computer Usage Company, Burton H. Bloom, Computer Usage Company, IBM Scientific Center, and Other MetricsView Article Metrics. 1970. Space/time trade-offs in hash coding with allowable errors. (Jul 1970). https://dl.acm.org/doi/10.1145/362686.362692

[5] Gediminas Adomavicius IEEE, Gediminas Adomavicius, Ieee, IEEEView Profile, Alexander Tuzhilin IEEE, Alexander Tuzhilin, and Other MetricsView Article Metrics. 2005. Toward the next generation of Recommender Systems: A survey of the state-of-the-art and possible extensions: IEEE Transactions on Knowledge and Data Engineering: Vol 17, no 6. (Jun 2005). https://dl.acm.org/doi/10.1109/TKDE.2005.99

[6] Badrul Sarwar, George Karypis, Joseph A Konstan, and John Riedl. 2016. Item-based collaborative filtering recommendation algorithms. (Apr 2016). https://experts.umn.edu/en/publications/item-based-collaborative-filtering-recommendation-algorithms

[7] Jieun Son, (CBF), C.P. Santos, D.H. Park, T. Opsahl, M.E. Newman, P.V. Marsden, B. Lika, S.H. Liao, Y. Li, and et al. 2017. Content-based filtering for recommendation systems using multiattribute Networks. (Aug 2017). https://www.sciencedirect.com/science/article/abs/pii/S0957417417305468