Erciyes Üniversitesi
Fen Bilimleri Enstitüsü Dergisi
Cilt 37, Sayı 3, 2021

Erciyes University
Journal of Institue Of Science and Technology
Volume 37, Issue 3, 2021

Movie Recommendation Systems Based on Collaborative Filtering: A Case Study on Netflix

Muhammed SÜTÇÜ[*1,2], **Ecem KAYA**[2], **Oğuzkan ERDEM**[2]

[*1] Abdullah Gül University, Faculty of Engineering, Industrial Engineering, KAYSERİ
[2] Abdullah Gül University, Institute of Social Sciences, Data Science for Business and Economics, KAYSERİ

(Alınış / Received: 22.08.2021, Kabul / Accepted: 12.10.2021, Online Yayınlanma / Published Online: 30.12.2021)

Keywords
Movie Recommendation, Recommendation Systems, Collaborative Filtering, Netflix Prize

Abstract: User ratings on items like movies, songs, and shopping products are used by Recommendation Systems (RS) to predict user preferences for items that have not been rated. RS has been utilized to give suggestions to users in various domains and one of the applications of RS is movie recommendation. In this domain, three general algorithms are applied; Collaborative Filtering that provides prediction based on similarities among users, Content-Based Filtering that is fed from the relation between item-user pairs and Hybrid Filtering one which combines these two algorithms. In this paper, we discuss which methods are more efficient in movie recommendation in the framework of Collaborative Filtering. In our analysis, we use Netflix Prize dataset and compare well-known Collaborative Filtering methods which are Singular Value Decomposition, Singular Value Decomposition++, K-Nearest Neighbour and Co-Clustering. The error of each method is calculated by using Root Mean Square Error (RMSE). Finally, we conclude that K-Nearest Neighbour method is more successful in our dataset.

İşbirlikçi Filtreleme Temelinde Film Öneri Sistemleri: Netflix Üzerinde Bir Vaka Çalışması

Anahtar Kelimeler
Film Öneri, Öneri Sistemleri, İşbirlikçi Filtreleme, Netflix Ödül

Öz: Filmler, şarkılar ve alışveriş ürünleri gibi ögelerin kullanıcı değerlendirmeleri Öneri Sistemleri (ÖS) tarafından henüz değerlendirilmemiş ürünleri tahmin etmek için kullanılır. ÖS kullanıcılara çeşitli alanlarda öneri vermek için geliştirilmiştir ve ÖS uygulama alanlarından birisi de film önerisidir. Bu alanda üç genel algoritma kullanılmaktadır; kullanıcılar arası benzerliğe dayanarak tavsiye veren İşbirlikçi Filtreleme, kullanıcı-eşya eşleştirilmesindeki ilişkiden beslenen İçerik Tabanlı Filtreleme ve bu iki algoritmayı birleştiren Hibrit Filtreleme. Bu çalışmamızda İşbirlikçi Filtreleme çerçevesinde hangi metotların daha etkili çalıştığı incelenmiştir. Analizimizde Netflix Ödül veri seti kullanılmış ve iyi bilinen İşbirlikçi Filtreleme metotları olan Tekil Değer Ayrışımı, Tekil Değer Ayrışımı++, K En Yakın Komşu ve Eş Kümeleme kıyaslanmıştır. Her metodun hatası Ortalama Hata Kare Kökü kullanılarak ölçülmüştür. Son olarak, K En Yakın Komşu metodunun veri setimizde daha başarılı olduğu sonuçlanmıştır.

***Corresponding Author, email:** muhammed.sutcu@agu.edu.tr

1. Introduction

Companies have developed new concepts as technology has become an alternative to the classical methods used in marketing and sales. Increasing competition in the market has created a need for new methods to be used in this field. Recommendation systems (RS) is the one of the methods that provide competitive advantage by allowing consumers spend more time on their websites or interface[1]. A recommendation system is an information filtering system that helps user's decision-making process in certain positions by declining a range of possible options and prioritizing these factors in given logic. RS is occurred to give offers to users in various domains and one of the application fields of RS is movie recommendation.

Although recommender systems have broad application areas, Collaborative Filtering (CF) is the one of the most used in research fields [2]. CF is used to predict a user's future preferences based on the user's past preferences. Media content providers like Netflix, Spotify uses collaborative filtering methods in their recommender systems [3]. The aim of this study is to compare results of different collaborative filtering methods as Singular Value Decomposition (SVD), Singular Value Decomposition++ (SVD++), K-Nearest Neighbor (KNN) and Co-Clustering by using Netflix Prize dataset.

Recommendation Systems is one of the most useful information systems for companies. General usage of these systems contains two main purposes. One of them is providing more personalized suggestions to users so that users can satisfy their needs among the various items or services. The other purpose is that companies can increase their profits with accurate and immediate solutions for customers. By this way, companies can create customer portfolios which serves to profit maximization purposes of companies[4]. According to the Forbes, Amazon's recommendation engine achieves %35 of annual total revenue of Amazon[5]. These data highlight the importance of recommendation systems for the development of all technology companies, especially e-commerce. Differently from previous articles, we present comprehensive comparable models as we know how profitable it is to discover the right model in practice. Like in other recommendation systems, movie recommendation systems are deployed with various techniques. Main techniques in the literature are Collaborative Filtering, Content Based Filtering and Hybrid Filtering.

Collaborative Filtering is one of the most effective technique which based on the assumption that users who have co-operated in the past will likely co-operate in the future. With this understanding CF tries to cluster users based on their similarities. Therefore, unrated movie or item by a specific user can be inferred by analysing the cluster that specific user belongs to [6][7][8][9].

Content Based Filtering (CBF) uses item-user pairing to give recommendation to users. Main assumption in the CBF is that if a user was interested in item(s) in the past, he or she will likely have interest in it in the future. In the movie recommendation field, CBF is used in various datasets [10][11].

Hybrid Filtering (HF), combines both Content Based Filtering and Collaborative Filtering, aims to maximize recommendation accuracy. For instance, one RS can use CF to recommend similar things to similar uses and it can also use CBF to give recommendation to specific user based on his or her past preferences. There are also studies which uses HF in the literature [12][13][14][15].

In this study, data is gathered from kaggle.com, Netflix Prize Dataset where the data was published officially by Netflix. The dataset includes movie ID, customer ID, ratings out of 5, and timestamp of the rating. Each movie ID and each customer ID represents a specific movie and customer. Although the dataset is introduced with four text files, in this study, we can use only one file to analyze because of insufficient technical infrastructure. Thus, our dataset comprises 24.053.764 ratings given by 470.758 customers to 4.499 movies.

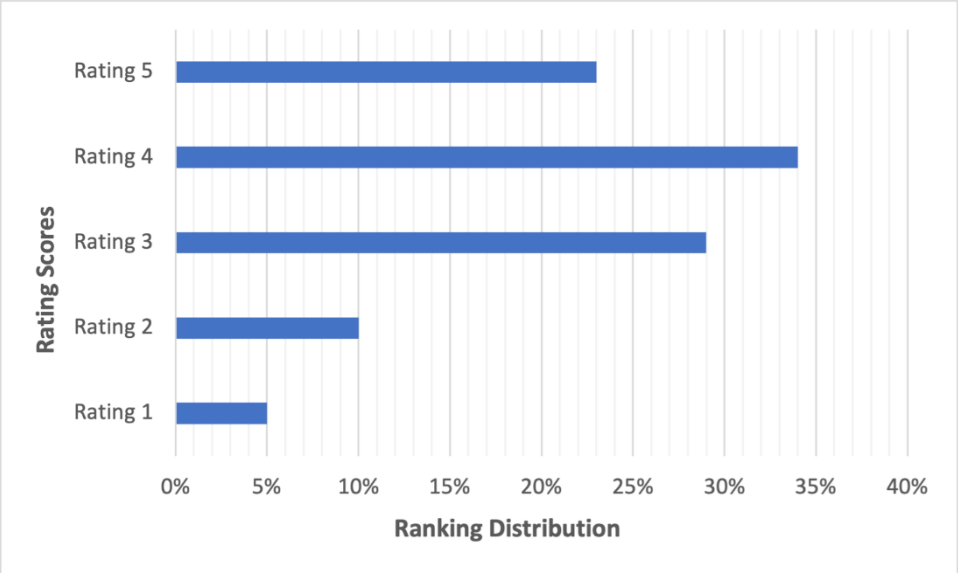

Figure 1. Distribution of ratings and counts

When the dataset is analyzed, we can easily demonstrate that people vote less at lower scores as shown in Figure 1. We assume that one of the most important reasons for this is that the audience did not choose to vote and closed the movie before it was finished. So, this could be considered a limitation for the dataset.

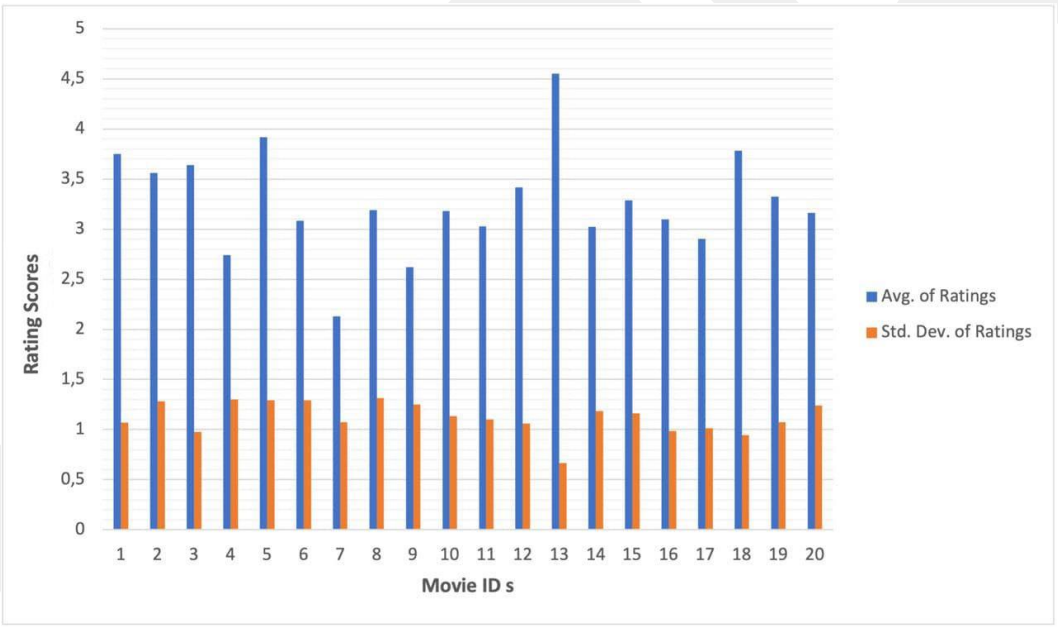

Figure 2. Rating statistics

To understand the dataset better, several statistical analyses based on movie ratings are calculated and shown in Figure 2. In order to easy to illustrate, we indicate the average scores and the standard deviations of the scores given to the first 20 movies. It can be observed that movies that have higher ratings tend to have a lower standard deviation.

Due to the huge number of movies in the dataset and the limited time of the people, every user rate only a small portion of the movies. The grand objective in the study is to develop models to predict the scores that a particular user will assign to a particular movie prior to the rating and propose an approach to the recommendation system of Netflix. Therefore, several data analytics techniques have been applied. Recommender systems is one of the hot topics of the field of data science due to their critical importance in e-commerce and streaming markets. Collaborative filtering is the recommender system method that predicts items depend on users' past experience. In this research, we aim to appraise performance of each collaborative filtering algorithms and demonstrate most effective algorithm.

The remainder of this article is structured as follows: In section 2, we present the basic notations and definitions used in the rest of the article. In section 3, we interpret the outputs of the applied methodologies and the comparison of the outputs among themselves. In section 4, we conclude the article and present our suggestions for future studies.

2. Material and Method

In this section, the methodologies that this article adopted will be explored. Methodology section is examined in four subtitles: Singular Value Decomposition, Singular Value Decomposition++, K-Nearest Neighbor and Co-clustering.

2.1. Singular value decomposition

Singular Value Decomposition (SVD) is the most used model of recommendation systems, and users and projects can create it as an array format. In addition, SVD is a matrix factorization technique that reduces the number of attributes in the data set by reducing the spatial dimension from N dimensions to K dimensions (here K <N)[16]. In the recommendation system, SVD divides the K-dimensional matrix created by users T and D into N-dimensional matrices. The working principle of singular value decomposition is shared as follows:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u \tag{1}$$

where, $\hat{r}_{ui}$ is the prediction set, μ expresses average ratings, b_u expresses average rating given by user u minus μ, b_i represents average rating of item i minus μ, q_i represents each item by vector and similarly p_u represents each user by vector, $q_i^T p_u$ represents dot product.

If user μ is unknown, the deviation b_u and p_u factors are assumed to be zero.

To predict all unknown scores the algorithm tries to minimize regularized squared error as follows:

$$\sum_{r_{ui} \in R_{train}} (r_{ui} - \hat{r}_{ui})^2 + \lambda(b_i^2 + b_u^2 + ||q_i||^2 + ||p_u||^2) \tag{2}$$

where, r_{ui} is rating and $\hat{r}_{ui}$ is prediction, λ represents regularization parameters.

The minimization process is performed by a very basic stochastic gradient descent as follows:

$$\begin{aligned} b_u &\leftarrow b_u + \gamma(e_{ui} - \lambda b_u) \\ b_i &\leftarrow b_i + \gamma(e_{ui} - \lambda b_i) \\ p_u &\leftarrow p_u + \gamma(e_{ui} \cdot q_i - \lambda p_u) \\ q_i &\leftarrow q_i + \gamma(e_{ui} \cdot p_u - \lambda q_i) \end{aligned} \tag{3}$$

where, $e_{ui} = r_{ui} - \hat{r}_{ui}$ and γ represents learning rate.

2.2. Singular value decomposition++

In matrix factorization techniques explicit feedbacks are considered. Different from SVD, SVD++ consolidates both implicit and explicit feedback. In this regard, communities and user groups are performed as implicit feedback [17]. Since user groups are communities are served, the formulation of SVD turns into the following one:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T (p_u + |I_u|^{-\frac{1}{2}} \sum_{j \in I_u} y_j) \tag{4}$$

where y_j terms there is a new set of item factors capturing implicit ratings.

2.3. K-nearest neighbor

K-Nearest Neighbor basically groups the users who have same minds about similar items. It is used for classification and regression of similar inputs in the k-closest environment. In our dataset, we have data classification problem to be solved and, we also try to cluster users to give them better recommendations. In this context, we try to use user ratings to calculate similarities. In our analysis, we use 4 different KNN algorithms to make comprehensive analysis. These are KNN Basic, KNN With Means which considers mean ratings of each

individual user, KNN With Z Score which normalize z score of each user and KNN With Baseline which uses baseline rating. The mathematical formulation of each KNN is given below in our analysis.

2.3.1. K-nearest neighbor basic

$$\hat{r}_{ui} = \frac{\sum_{v \in N_i^k(u)} \text{sim}(u,v) \cdot r_{vi}}{\sum_{v \in N_i^k(u)} \text{sim}(u,v)}$$

or

$$\hat{r}_{ui} = \frac{\sum_{j \in N_u^k(i)} \text{sim}(i,j) \cdot r_{uj}}{\sum_{j \in N_u^k(i)} \text{sim}(i,j)} \quad (5)$$

where, $\hat{r}_{ui}$ represents prediction, k represents number of neighbors, $\text{sim}(u,v)$ and $\text{sim}(i,j)$ express similarity measures as they are explained in results section, r_{vi} represents rating of user v on item i and similarly r_{uj} represents rating of user u on item j.

2.3.2. K-nearest neighbor with means

$$\hat{r}_{ui} = \mu_u + \frac{\sum_{v \in N_i^k(u)} \text{sim}(u,v) \cdot (r_{vi} - \mu_v)}{\sum_{v \in N_i^k(u)} \text{sim}(u,v)}$$

or

$$\hat{r}_{ui} = \mu_i + \frac{\sum_{j \in N_u^k(i)} \text{sim}(i,j) \cdot (r_{uj} - \mu_j)}{\sum_{j \in N_u^k(i)} \text{sim}(i,j)} \quad (6)$$

where, $\hat{r}_{ui}$ represents prediction, μ_u represents mean of user u and similarly μ_i represents mean of item i, k represents number of neighbors, $\text{sim}(u,v)$ and $\text{sim}(i,j)$ express similarity measures as they are explained in results section, r_{vi} represents rating of user v on item i and similarly r_{uj} represents rating of user u on item j, μ_v represents mean of user v and similarly μ_j represents mean of item j.

2.3.3. K-nearest neighbor with z score

$$\hat{r}_{ui} = \mu_u + \sigma_u \frac{\sum_{v \in N_i^k(u)} \text{sim}(u,v) \cdot (r_{vi} - \mu_v)/\sigma_v}{\sum_{v \in N_i^k(u)} \text{sim}(u,v)}$$

or

$$\hat{r}_{ui} = \mu_i + \sigma_i \frac{\sum_{j \in N_u^k(i)} \text{sim}(i,j) \cdot (r_{uj} - \mu_j)/\sigma_j}{\sum_{j \in N_u^k(i)} \text{sim}(i,j)} \quad (7)$$

where, $\hat{r}_{ui}$ represents prediction, μ_u represents mean of user u and similarly μ_i represents mean of item i, σ_u and σ_v represents the standard deviation of all ratings given by user u and v, k represents number of neighbors, $\text{sim}(u,v)$ and $\text{sim}(i,j)$ express similarity measures as they are explained in results section, r_{vi} represents rating of user v on item i and similarly r_{uj} represents rating of user u on item j, μ_v represents mean of user v and similarly μ_j represents mean of item j, σ_i and σ_j the standard deviation of all ratings given to item i and j respectively.

2.3.4. K-nearest neighbor baseline

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{v \in N_i^k(u)} \text{sim}(u,v) \cdot (r_{vi} - b_{vi})}{\sum_{v \in N_i^k(u)} \text{sim}(u,v)}$$

or

(8)

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{j \in N_u^k(i)} \text{sim}(i,j) \cdot (r_{uj} - b_{uj})}{\sum_{j \in N_u^k(i)} \text{sim}(i,j)}$$

where, $\hat{r}_{ui}$ represents prediction, b_{ui} represents the baseline rating of user u for item i, k represents number of neighbors, $\text{sim}(u,v)$ and $\text{sim}(i,j)$ express similarity measures as they are explained in results section, r_{vi} represents rating of user v on item i and similarly r_{uj} represents rating of user u on item j, b_{vi} represents the baseline rating of user v for item i and similarly b_{uj} represents the baseline rating of user u for item j.

2.4. Co-clustering

Co-Clustering concept was introduced by Thomas George and Srujana Merugu as novel algorithm under the framework of collaborative filtering [18]. In this algorithm users and items form some clusters and some co-clusters. Basically, users and items are assigned into some clusters C_u, C_i and some co-clusters C_{ui}.
The prediction $\hat{r}_{ui}$ is set follows as:

$$\hat{r}_{ui} = \overline{C_{ui}} + (\mu_u - \overline{C_u}) + (\mu_i - \overline{C_i}) \quad (9)$$

where, $\hat{r}_{ui}$ represents prediction, $\overline{C_{ui}}$ is average rating of co-cluster C_{ui}, $\overline{C_u}$ is the average rating of u's cluster, and $\overline{C_i}$ is the average rating of i's cluster. If the user is unknown, the prediction is $\hat{r}_{ui} = \mu_i$. If the item is unknown, the prediction is $\hat{r}_{ui} = \mu_u$. If both the user and the item are unknown, the prediction is $\hat{r}_{ui} = \mu$.

3. Results

In our study, we aim to compare the Collaborative Filtering methods on the Netflix Prize dataset and find out which method is more effective in the movie recommendation system. First, we import the data and then we clean the missing data and make the data ready for model writing. These operations are made of using the Python language 3.8.2. version and in Jupyter Notebook as IDE as an environment. We use Pandas, NumPy, Math, and Re libraries for data cleaning and missing data operations.

Movie_Id	3	8	16	17	18	26	28	30	32	33	...	4472	4474	4478	4479	4485	4488	4490	4492	4493	4496
Cust_Id																					
6	NaN	NaN	NaN	NaN	NaN	NaN	NaN	3.0	NaN	NaN	...	3.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
7	NaN	5.0	NaN	NaN	NaN	NaN	4.0	5.0	NaN	NaN	...	3.0	NaN	NaN	5.0	NaN	NaN	NaN	NaN	NaN	NaN
79	NaN	NaN	NaN	NaN	NaN	NaN	NaN	3.0	NaN	NaN	...	4.0	NaN	NaN	NaN	NaN	NaN	4.0	NaN	NaN	NaN
97	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
134	NaN	NaN	NaN	NaN	NaN	NaN	5.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2649370	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2649378	NaN	NaN	NaN	NaN	NaN	NaN	3.0	3.0	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2649388	NaN	NaN	NaN	NaN	NaN	NaN	NaN	3.0	NaN	NaN	...	3.0	NaN	NaN	3.0	NaN	3.0	NaN	NaN	NaN	NaN
2649426	NaN	NaN	NaN	4.0	NaN	NaN	4.0	4.0	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2649429	NaN	NaN	NaN	NaN	NaN	NaN	NaN	5.0	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

143458 rows × 1350 columns

Figure 4. Data sample before data cleaning

Movie_Id	3	8	16	17	18	26	28	30	32	33	...	4472	4474	4478	4479	4485	4488	4490	4492	4493	4496
Cust_Id																					
6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.0	0.0	...	3.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
7	0.0	5.0	0.0	0.0	0.0	0.0	4.0	5.0	0.0	0.0	...	3.0	0.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0
79	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.0	0.0	...	4.0	0.0	0.0	0.0	0.0	0.0	4.0	0.0	0.0	0.0
97	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
134	0.0	0.0	0.0	0.0	0.0	0.0	5.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2649370	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2649378	0.0	0.0	0.0	0.0	0.0	0.0	3.0	3.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2649388	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.0	0.0	...	3.0	0.0	0.0	3.0	0.0	3.0	0.0	0.0	0.0	0.0
2649426	0.0	0.0	0.0	4.0	0.0	0.0	4.0	4.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2649429	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

143458 rows × 1350 columns

Figure 5. Data sample after data cleaning

Since every movie is not rated by every user, some attributes have NaN value. Also, if any data is corrupted, it is cleaned out. In Figure 5, data which is missing or corrupted as in Figure 4 is handled.

To visualize our dataset, we use Matplotlib and Seaborn libraries. Finally, we use Surprise library to apply methods, similarity measures, creating train sets, validation sets, test sets and measuring error rates.

After we get our data ready for testing, we conduct SVD, SVD++, Co-Clustering methods. With these methods we only get singular results. However, K-Nearest Neighbor methods offer different similarity measures. These are Cosine, Mean Squared Difference (MSD), Pearson, and Pearson Baseline. In total K-Nearest Neighbor gives us 16 different outcomes. Therefore, in grand total we get 19 different outcomes to evaluate which method is the best. Similarity measures' mathematical formulations as follow:

$$\text{cosine\_sim}(u,v) = \frac{\sum_{i \in I_{uv}} r_{ui} \cdot r_{vi}}{\sqrt{\sum_{i \in I_{uv}} r_{ui}^2} \cdot \sqrt{\sum_{i \in I_{uv}} r_{vi}^2}}$$

or

$$\text{cosine\_sim}(i,j) = \frac{\sum_{u \in U_{ij}} r_{ui} \cdot r_{uj}}{\sqrt{\sum_{u \in U_{ij}} r_{ui}^2} \cdot \sqrt{\sum_{u \in U_{ij}} r_{uj}^2}} \quad (10)$$

$$MSD(u,v) = \frac{1}{|I_{uv}|} \cdot \sum_{i \in I_{uv}} (r_{ui} - r_{vi})^2$$

or

$$MSD(i,j) = \frac{1}{|U_{ij}|} \cdot \sum_{u \in U_{ij}} (r_{ui} - r_{uj})^2 \quad (11)$$

MSD defined then as follows:

$$\begin{aligned} \text{MSD\_sim}(u,v) &= \frac{1}{MSD(u,v)+1} \\ \text{MSD\_sim}(i,j) &= \frac{1}{MSD(i,j)+1} \end{aligned} \tag{12}$$

where, +1 represents avoiding possible dividing by zero problem.

$$\text{pearson\_sim}(u,v) = \frac{\sum_{i\in I_{uv}} (r_{ui}-\mu_u)\cdot(r_{vi}-\mu_v)}{\sqrt{\sum_{i\in I_{uv}} (r_{ui}-\mu_u)^2}\cdot\sqrt{\sum_{i\in I_{uv}} (r_{vi}-\mu_v)^2}}$$

or

$$\text{pearson\_sim}(i,j) = \frac{\sum_{u\in U_{ij}} (r_{ui}-\mu_i)\cdot(r_{uj}-\mu_j)}{\sqrt{\sum_{u\in U_{ij}} (r_{ui}-\mu_i)^2}\cdot\sqrt{\sum_{u\in U_{ij}} (r_{uj}-\mu_j)^2}} \tag{13}$$

$$\text{pearson\_baseline\_sim}(u,v) = \hat{\rho}_{uv} = \frac{\sum_{i\in I_{uv}} (r_{ui}-b_{ui})\cdot(r_{vi}-b_{vi})}{\sqrt{\sum_{i\in I_{uv}} (r_{ui}-b_{ui})^2}\cdot\sqrt{\sum_{i\in I_{uv}} (r_{vi}-b_{vi})^2}}$$

or

$$\text{pearson\_baseline\_sim}(i,j) = \hat{\rho}_{ij} = \frac{\sum_{u\in U_{ij}} (r_{ui}-b_{ui})\cdot(r_{uj}-b_{uj})}{\sqrt{\sum_{u\in U_{ij}} (r_{ui}-b_{ui})^2}\cdot\sqrt{\sum_{u\in U_{ij}} (r_{uj}-b_{uj})^2}} \tag{14}$$

The prediction models are tested on validation set at first and then they are tested on test sets. To evaluate performances of each method and similarity measures we use Root Mean Square Error as performance indicator. RMSE measures the square root of the difference between actual values and model predicted outcomes, with a correction factor. RMSE take values between 0 and positive infinity. Ideally, lower RMSE score, closer to 0, means that model fits for the data more accurately since the difference between actual value and predicted value gets smaller. On the contrary, higher RMSE means that model gets far between actual value and predicted value. Therefore, accuracy of the model decreases with this gap between actual and predicted value.

$$\text{RMSE} = \sqrt{\frac{1}{|\hat{R}|}\sum_{\hat{r}_{ui}\in\hat{R}} (r_{ui}-\hat{r}_{ui})^2} \tag{15}$$

RMSE values for the models are as given in the following tables.

Table 1. K-nearest neighbor methods and related similarity measures.

SIMILARITY MEASURES / METHODS	Cosine	MSD	Pearson	Pearson Baseline
KNN BASIC	0.9621	0.9022	0.9198	0.8855 *
KNN WITH MEANS	0.8789	0.8788	0.8677	0.8488 ***
KNN WITH Z SCORE	0.8819	0.8798	0.8667	0.8479 ****
KNN BASELINE	0.8806	0.8782	0.8674	0.8482 **

Table 2. SVD,SVD++ and Co-Clustering results.

METHODS	SINGULAR OUTCOMES
SVD	0.8506
SVD++	0.8507
CO-CLUSTERING	0.8966

As it is explained in the previous sections, K Nearest Neighbour methods allow us to use similarity measures as well. In the Table 1, K Nearest Neighbour methods are compared among themselves and the most successful similarity measure is marked with star sign in each line. Also starred outcomes are compared among themselves and having more stars at the end of each row means more successful results.

On the other hand, SVD and SVD++ models performed nearly same results in the Table 2. They have only 0.0001 point difference. When it comes to Co-Clustering method, we get relatively higher score which means that Co-Clustering method is not fitting well the data as SVD and SVD++ does. So, it can be concluded that SVD has more successful results.

When we compare all results together, with the Pearson Baseline similarity measure KNN With Means, KNN With Z Score and KNN Baseline brings better results. However, SVD and SVD++ shows singular results and they are better than KNNs with other similarity measures. Therefore, we can conclude that KNN with Means, KNN With Z Score and KNN Baseline are better if they are conducted with Pearson Baseline method. Alternatively, SVD and SVD++ gives more consistent results relatively since they don't have similarity measures. In overall performance, KNN With Z Score with Pearson Baseline similarity measure provides the best fit in our data set.

4. Discussion and Conclusion

In this study we propose a comparative collaborative filtering algorithms using the Netflix Prize dataset under the framework of recommender systems. In the literature we find that each method under the collaborative filtering is studied in different aspects and contexts. However, we use comparative method for collaborative filtering for Netflix Prize dataset in particular. We try to find out which collaborative method is most suitable solution for the Netflix Prize dataset. Using Python and related libraries, we process the data first and then imply different collaborative filtering algorithms. Each algorithm's and model's success is measured with the Root Mean Square Error. It is concluded KNN With Z Score method with Pearson Baseline similarity measure brings the most successful outcome.

On the other hand, we only study on the quarter of the data because of the unsufficient technical infrastructure. Thus, this study's results may change if all data is processed in collective way but since the data is portioned it is not expected to experience dramatic changes. Furthermore, Netflix dataset does not contain any information about the contents of the movies. Therefore, content-based filtering and hybrid models (that works with content-based filtering) cannot be used. In future studies content-based filtering and therefore hybrid models can be used by retrieving data from IMDB or Netflix itself. These studies might reflect on movies' contents, genres and so on. Also, Hybrid models support the collaborative filtering. Therefore, more comprehensive study can be performed.

Acknowledgment

We thank RA. Sami Kaya (Abdullah Gül University) for providing data visuals and RA. Suat Mumcu (Abdullah Gül University) for assistance in the writing process.

References

[1] W. Deng, R. Patil, L. Najjar, Y. Shi, and Z. Chen, "Incorporating Community Detection and Clustering Techniques into Collaborative Filtering Model," *Procedia Comput. Sci.*, vol. 31, pp. 66–74, 2014, doi: 10.1016/j.procs.2014.05.246.

[2] J. Bobadilla, F. Serradilla, and A. Hernando, "Collaborative filtering adapted to recommender systems of E-learning," *Knowledge-Based Syst.*, vol. 22, pp. 261–265, May 2009, doi: 10.1016/j.knosys.2009.01.008.

[3] I. Tobías, "Matrix factorization models for cross-domain recommendation: Addressing the cold start in collaborative filtering," 2017.

[4] F. Şahin and C. Söylemez, "Güdülenmiş Tüketici Yenilikçiliğinin Dijital Medya Platformlarının Algılanan Tüketici Temelli Marka Değeri Ve Marka Tutumu Üzerine Etkisi: Covıd-19 Döneminde Netflıx Üzerine Bir İnceleme," *Erciyes Üniversitesi İktisadi ve İdari Bilim. Fakültesi Derg.*, vol. 58, pp. 301–332, 2021.

[5] B. Morgan, "How Amazon Has Reorganized Around Artificial Intelligence And Machine Learning." https://www.forbes.com/sites/blakemorgan/2018/07/16/how-amazon-has-re-organized-around-artificial-intelligence-and-machine-learning/?sh=704109107361. (Access date: 10.10.2021)

[6] T. Anwar and V. Uma, "Comparative study of recommender system approaches and movie recommendation using collaborative filtering," *Int. J. Syst. Assur. Eng. Manag.*, vol. 12, no. 3, pp. 426–436, 2021, doi: 10.1007/s13198-021-01087-x.

[7] Z. Wang, X. Yu, N. Feng, and Z. Wang, "An improved collaborative movie recommendation system using computational intelligence," *J. Vis. Lang. Comput.*, vol. 25, no. 6, pp. 667–675, 2014, doi: 10.1016/j.jvlc.2014.09.011.

[8] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *Knowl. Data Eng. IEEE Trans.*, vol. 17, pp. 734–749, Aug. 2005, doi: 10.1109/TKDE.2005.99.

[9] N. Yi, C. Li, X. Feng, and M. Shi, "Design and implementation of movie recommender system based on graph database," *Proc. - 2017 14th Web Inf. Syst. Appl. Conf. WISA 2017*, vol. 2018-Janua, pp. 132–135, 2018, doi: 10.1109/WISA.2017.34.

[10] S. R. S. Reddy, S. Nalluri, S. Kunisetti, S. Ashok, and V. Bachu, "Content-Based Movie Recommendation System Using Genre Correlation: Proceedings of the Second International Conference on SCI 2018, Volume 2," 2019, pp. 391–397.

[11] J. Son and S. B. Kim, "Content-based filtering for recommendation systems using multiattribute networks," *Expert Syst. Appl.*, vol. 89, pp. 404–412, 2017, doi: https://doi.org/10.1016/j.eswa.2017.08.008.

[12] Y. Wang, M. Wang, and W. Xu, "A Sentiment-Enhanced Hybrid Recommender System for Movie Recommendation: A Big Data Analytics Framework," *Wirel. Commun. Mob. Comput.*, vol. 2018, p. 8263704, 2018, doi: 10.1155/2018/8263704.

[13] K. N. Jain, V. Kumar, P. Kumar, and T. Choudhury, "Movie Recommendation System: Hybrid Information Filtering System," in *Intelligent Computing and Information and Communication*, 2018, pp. 677–686.

[14] X. Li, W. Jiang, W. Chen, J. Wu, and G. Wang, "HAES: A New Hybrid Approach for Movie Recommendation with Elastic Serendipity," Aug. 2019, pp. 1503–1512, doi: 10.1145/3357384.3357868.

[15] C. Christakou, S. Vrettos, and A. Stafylopatis, "A hybrid movie recommender system based on neural networks," *Int. J. Artif. Intell. Tools*, vol. 16, no. 5, pp. 771–792, 2007, doi: 10.1142/S0218213007003540.

[16] R. Salakhutdinov and A. Mnih, "Bayesian Probabilistic Matrix Factorization Using Markov Chain Monte Carlo," in *Proceedings of the 25th International Conference on Machine Learning*, 2008, pp. 880–887, doi: 10.1145/1390156.1390267.

[17] L. Baltrunas, T. Makcinskas, and F. Ricci, "Group Recommendations with Rank Aggregation and Collaborative Filtering," in *Proceedings of the Fourth ACM Conference on Recommender Systems*, 2010, pp. 119–126, doi: 10.1145/1864708.1864733.

[18] T. George and S. Merugu, "A Scalable Collaborative Filtering Framework Based on Co-Clustering," in *Proceedings of the Fifth IEEE International Conference on Data Mining*, 2005, pp. 625–628, doi: 10.1109/ICDM.2005.14.