# Movie Recommendation System

BDA 492: Data Mining and its Applications
Assignment 3

Manav Prabhakar

July 2021

## 1 Introduction

Here, we have implemented a basic version of low-rank matrix factorization for recommendations and have employed it on a dataset of 1 million movie ratings (from 1 to 5) available from the MovieLens project. The MovieLens datasets were created and collected by GroupLens Research at the University of Minnesota. Recommender systems are generally built using the following approaches

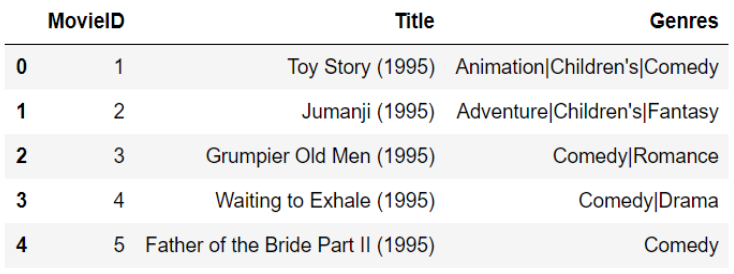
- **Collaborative filtering:** Collaborative Filtering, on the other hand, doesn't need anything else except users' historical preference on a set of items. Because it's based on historical data, the core assumption here is that the users who have agreed in the past tend to also agree in the future. The standard method of Collaborative Filtering is known as Nearest Neighborhood algorithm.
- **Content-Based:** This is more personalized way of recommending. Here, the attributes to be recommended are based on entirely the choices that the user made. Thus, only the information about previously consumed items are used to model user's preferences. It uses item features to recommend other items similar to what the user likes, based on their previous actions or explicit feedback.

We will be using Matrix Factorization for building and testing our recommendation system

## 2 Matrix Factorization

Since sparsity and scalability are the two biggest challenges for standard Collaborative Filtering method, it comes a more advanced method that decompose the original sparse matrix to low-dimensional matrices with latent factors/features and less sparsity. That is Matrix Factorization.

Beside solving the issues of sparsity and scalability, there's an intuitive explanation of why we need low-dimensional matrices to represent users' preference. A user gave good ratings to movie Avatar, Gravity, and Inception. They are not necessarily 3 separate opinions but



| | MovieID | Title | Genres |
|---|---|---|---|
| 0 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy |
| 1 | 2 | Jumanji (1995) | Adventure\|Children's\|Fantasy |
| 2 | 3 | Grumpier Old Men (1995) | Comedy\|Romance |
| 3 | 4 | Waiting to Exhale (1995) | Comedy\|Drama |
| 4 | 5 | Father of the Bride Part II (1995) | Comedy |

Figure 1: The movies that are there in the dataset

showing that this users might be in favor of Sci-Fi movies and there may be many more Sci-Fi movies that this user would like. Unlike specific movies, latent features is expressed by higher-level attributes, and Sci-Fi category is one of latent features in this case.

What matrix factorization eventually gives us is how much a user is aligned with a set of latent features, and how much a movie fits into this set of latent features. The advantage of it over standard nearest neighborhood is that even though two users haven't rated any same movies, it's still possible to find the similarity between them if they share the similar underlying tastes, again latent features.

To see how a matrix being factorized, first thing to understand is Singular Value Decomposition(SVD). We know from Linear Algebra that any real matrix $R$ can be decomposed into 3 matrices $U$, $\Sigma$, and $V$. Continuing using movie example, $U$ is an $n \times r$ user-latent feature matrix, $V$ is an $mr$ movie-latent feature matrix. $\Sigma$ is an $r \times r$ diagonal matrix containing the singular values of original matrix, simply representing how important a specific feature is to predict user preference. At a high level, SVD is an algorithm that decomposes a matrix $R$ into the best lower rank (i.e. smaller/simpler) approximation of the original matrix $R$. Mathematically, it decomposes R into a two unitary matrices and a diagonal matrix:

$$R = U\Sigma V^T$$

where R is users's ratings matrix, $U$ is the user "features" matrix, $\Sigma$ is the diagonal matrix of singular values (essentially weights), and $V^T$ is the movie "features" matrix. $U$ and $V^T$ are orthogonal, and represent different things. $U$ represents how much users "like" each feature and $V^T$ represents how relevant each feature is to each movie.

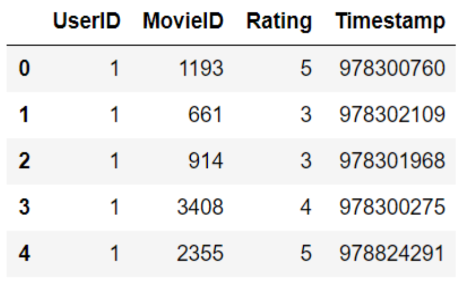
$$U \epsilon R^{n\times r},\ \sigma \epsilon R^{r\times r},\ V \epsilon R^{r\times m}$$

To get the lower rank approximation, we take these matrices and keep only the top $k$ features, which we think of as the underlying tastes and preferences vectors.

## 3 Implementation

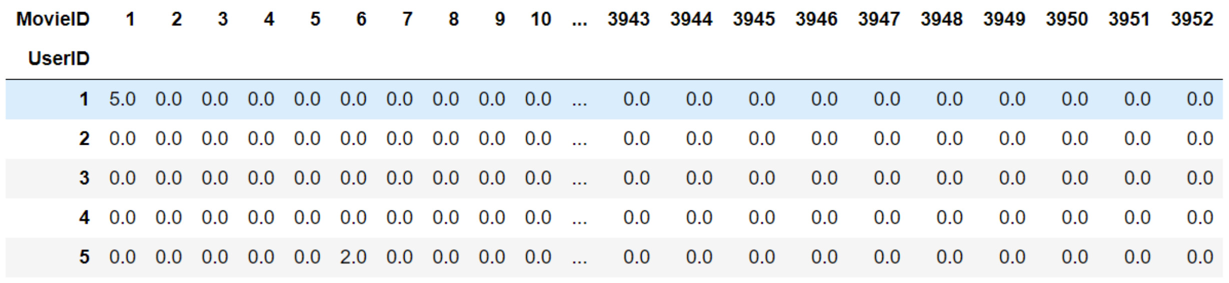
The recommendation system has been implemented using python on MovieLens dataset.

We would require he format of the ratings matrix to be one row per user and one column per movie. Thus, we form another table (see Figure 3 Then, we normalize the data by each



| | UserID | MovieID | Rating | Timestamp |
|---|---|---|---|---|
| **0** | 1 | 1193 | 5 | 978300760 |
| **1** | 1 | 661 | 3 | 978302109 |
| **2** | 1 | 914 | 3 | 978301968 |
| **3** | 1 | 3408 | 4 | 978300275 |
| **4** | 1 | 2355 | 5 | 978824291 |

Figure 2: Ratings by each user for the movies.



| MovieID | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | ... | 3943 | 3944 | 3945 | 3946 | 3947 | 3948 | 3949 | 3950 | 3951 | 3952 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **UserID** | | | | | | | | | | | | | | | | | | | | | |
| **1** | 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 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| **2** | 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 |
| **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 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| **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 | 0.0 | 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 | 2.0 | 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 |

Figure 3: One row per user and one column per movie

user's mean.

We can now perform Singular Value Decomposition and use the decomposed matrix for predictions for each user,

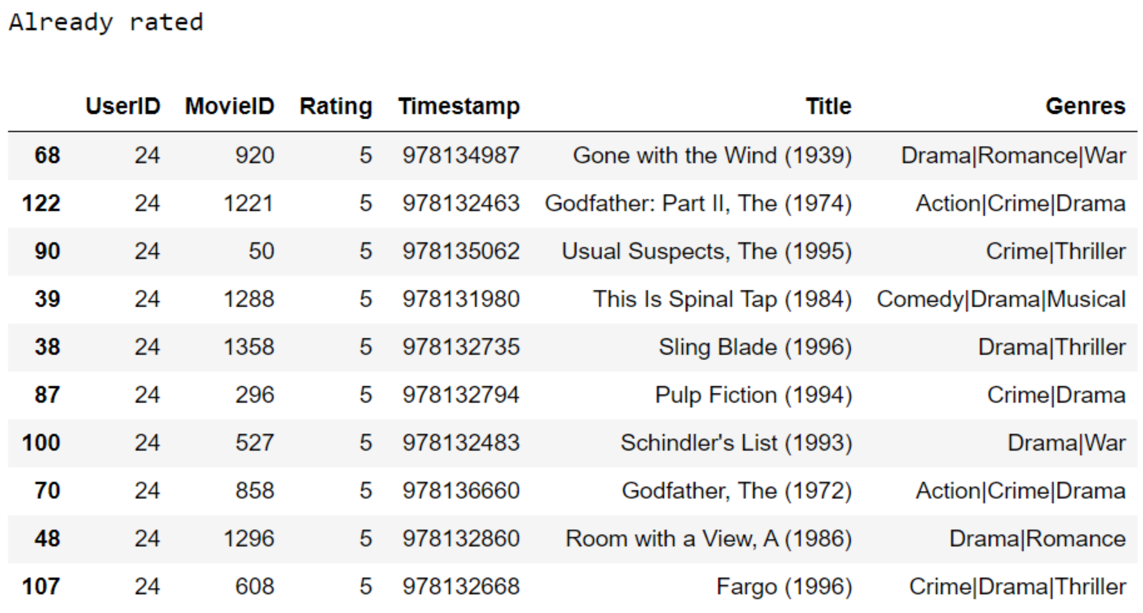
With the predictions matrix for every user, we can recommend movies for any user. by returning the movies with the highest predicted rating(see Figure 5) that the specified user hasn't already rated. The rated movies are shown and displayed in Figure 4. It is important to note that we haven't used any explicit movie content features (such as genre or title).

## 4 Acknowledgement

To Prof. Santosh Singh, Big Data Analytics Centre for providing an opportunity to understand, implement and analyze a Recommendation System.

## 5 Takeaways and Conclusion

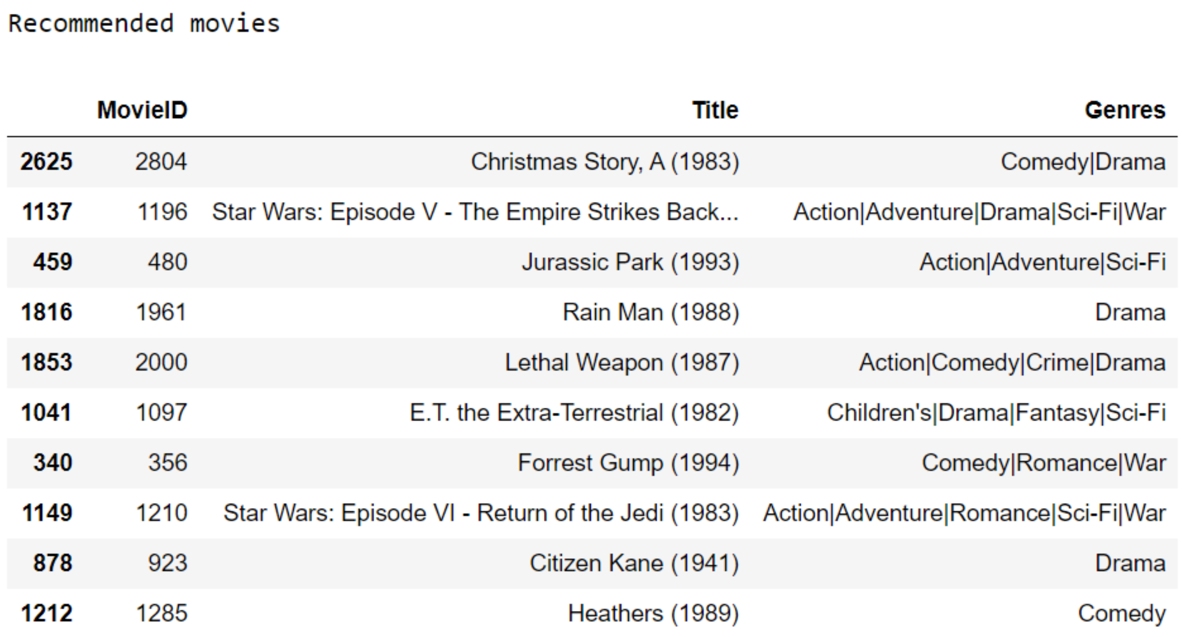
A decent recommendation system was successfully implemented using the Matrix Factorization method. Unlike supervised learning, we do not have any metrics to compare to for checking our accuracy and precision. However, backed by mathematical principles, the approach and implementation seem to be good.



```
Already rated
```

| | UserID | MovieID | Rating | Timestamp | Title | Genres |
|---|---|---|---|---|---|---|
| **68** | 24 | 920 | 5 | 978134987 | Gone with the Wind (1939) | Drama\|Romance\|War |
| **122** | 24 | 1221 | 5 | 978132463 | Godfather: Part II, The (1974) | Action\|Crime\|Drama |
| **90** | 24 | 50 | 5 | 978135062 | Usual Suspects, The (1995) | Crime\|Thriller |
| **39** | 24 | 1288 | 5 | 978131980 | This Is Spinal Tap (1984) | Comedy\|Drama\|Musical |
| **38** | 24 | 1358 | 5 | 978132735 | Sling Blade (1996) | Drama\|Thriller |
| **87** | 24 | 296 | 5 | 978132794 | Pulp Fiction (1994) | Crime\|Drama |
| **100** | 24 | 527 | 5 | 978132483 | Schindler's List (1993) | Drama\|War |
| **70** | 24 | 858 | 5 | 978136660 | Godfather, The (1972) | Action\|Crime\|Drama |
| **48** | 24 | 1296 | 5 | 978132860 | Room with a View, A (1986) | Drama\|Romance |
| **107** | 24 | 608 | 5 | 978132668 | Fargo (1996) | Crime\|Drama\|Thriller |

Figure 4: Already rated movies



```
Recommended movies
```

| | MovieID | Title | Genres |
|---|---|---|---|
| **2625** | 2804 | Christmas Story, A (1983) | Comedy\|Drama |
| **1137** | 1196 | Star Wars: Episode V - The Empire Strikes Back... | Action\|Adventure\|Drama\|Sci-Fi\|War |
| **459** | 480 | Jurassic Park (1993) | Action\|Adventure\|Sci-Fi |
| **1816** | 1961 | Rain Man (1988) | Drama |
| **1853** | 2000 | Lethal Weapon (1987) | Action\|Comedy\|Crime\|Drama |
| **1041** | 1097 | E.T. the Extra-Terrestrial (1982) | Children's\|Drama\|Fantasy\|Sci-Fi |
| **340** | 356 | Forrest Gump (1994) | Comedy\|Romance\|War |
| **1149** | 1210 | Star Wars: Episode VI - Return of the Jedi (1983) | Action\|Adventure\|Romance\|Sci-Fi\|War |
| **878** | 923 | Citizen Kane (1941) | Drama |
| **1212** | 1285 | Heathers (1989) | Comedy |

Figure 5: Recommended movies for the given user