Movie Recommendation System using MovieLens Dataset

Data Mining 7118 Project

Barbara Schmitz
Department of Data Science
University of Memphis
bschmitz@memphis.edu

ABSTRACT

The recommendation systems are the computer programs that seeks to filter choices according to user's interests and previous preferences. A simple movie recommendation system can be designed by applying some data mining techniques such as K-Means clustering and collaborative filtering. These methods are types of unsupervised machine learning algorithm were used to find groups of users that share the common interests, and predict the rating score for unseen movies. By collaborative filtering, we are referring to memory-based algorithms approach that "tries to find users that are similar to the active users and uses their preferences to predict ratings for the active user" [1].

KEYWORDS

Movie Recommendation System, data mining, principal component analysis, k-means clustering, collaborative filtering

1 Introduction

Recommendation system plays an integral role in streaming services nowadays. Most of streaming services such as Amazon, Netflix, Hulu all use recommendation system to improve their services to increase user satisfaction and increase revenue. The main purpose of movie recommendation system is to help users by suggesting which movie to watch without the time-consuming hassle of manually scrolling through hundreds to thousands of movies in a system's database. For Amazon and Netflix, the recommendation system essentially recommends items to users based on ratings and customers search history.

The objective for this project is to use two data mining techniques to create a movie recommendation system based on rating behaviors and movie genres and predict the rating score for unseen movies using K-means clustering and collaborative filtering. The following session introduces the technical background and terminology as well as discussing how they are related to this project.

1.1 The Dataset

MovieLens dataset is provided by GroupLens Research – a research lab in the Department of Computer Science and Engineering at the University of Minnesota [2]. MovieLens dataset were collected over periods of time and were used widely for many experimental tool and interfaces for recommendation. GroupLens Research lab makes their data public, non-commercial and free of advertisements.

In this project, the MovieLens (ml-latest-small) dataset is used for the project. This dataset was last updated September 2018. dataset describes "5-star rating and free-text tagging activity", "contains 100836 ratings and 3683 tag applications across 9742 movies", and "these data were created by 610 users between March 29, 1996, and September 24, 2018., 2018" [3]. According to [3], 610 users were selected randomly, all selected users had rated at least movies, and each user is represented by an id and has no demographic information.

The data format is in Comma Separated Values (CSV) files like ratings.csv, movies.csv, tags.csv, and links.csv, and these files are encoded as UTF-8. In this project, only the movies.csv and the ratings.csv were used for this experiment.

According to [3], the ratings.csv file that contains all ratings, and "each row represents one rating of one movie by one user, and has the following format:

userId, movieId, rating, timestamp

The rating star are made on a 5-star scale, with half star increments, and the timestamps represent seconds since midnight Coordinated Universal Time (UTC) of January 1, 1970".

Similarly, the movies.csv contains all the movie information, and "each line of this file represents one movie, and has the following format:

movieId, title, genres".

The genres include action, adventure, animation, children, comedy, crime, documentary, drama, fantasy, film-noir, horror, musical, mystery, romance, sci-fi, thriller, war, western, and no genres listed.

1.2 K-Means Clustering

K-Means clustering is one popular unsupervised machine learning algorithm. According to [4], "K-Means clustering is a centroid-based algorithm, or a distance-based algorithms, where we calculate the distances to assign a point to a cluster, and each cluster is associated with a centroid". In addition, the number of clusters found by the method is denoted by letter k.

1.3 Collaborative Filtering

Collaborative Filtering is an unsupervised machine learning technique. It is a well-established approach and a popular choice for building recommendation systems. Collaborative Filtering focuses on finding similarity between users and a set of items that

are matched against the past records of user-item interactions within a larger group to predict ratings for items that they have not rated previously.

1.4 Root Mean Square Error

Root-Means-Square Error (RMSE) is one of the most widely used metrics in statistics to measure how much error there is between the predicted values and actual values. RMSE is commonly used for evaluating the quality of prediction.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}} \quad (1)$$

Where $\hat{y}_i$ are the predicted values and y_i are observed values and n is the number of observations.

2 Related Work

From the past decades, movie recommendation system has been a popular topic of research. Many businesses have recognized its useful application to their existing products. In order to create the high-quality recommendation system, many researchers have applied different machine learning techniques which affects the process of getting the final results. In the paper [5], a movie recommendation system has been built by applying K-Mean clustering and soft max regression on user rating, and Root-Means-Square Error has been used for evaluation.

In the paper [6], Collaborative Filtering technique has been used for creating a movie recommendation system by finding the most appropriate similar users based on rating and genres to a particular user to recommend.

3 Solving the Problem

For this project, Jupyter Notebook version 6.4.8 and Python 3 programming language would be used to perform the experiment.

3.1 Preprocessing Data

In order to make the dataset ready to perform later analysis steps, preprocessing data was necessary. First, the ratings.csv file would be read using Pandas Python library to generate a data frame that contained only favorite movies of all users. In this project, the movies data frame contained the genres together in one column. In an effort to perform better analysis later, each genre became a feature and contained 1 if the movie was classified as that genre or set of genres and 0 if the movie was that genre or set of genres. Also, the movie title contained the year the movie was released. The year was extracted and made a feature as well. For the rating data frame, it was not manipulated. Next, the two data frames were joined by the unique movieId. Missing values were dropped from the joined dataframes.

3.2 Exploratory Data Analysis

In an effort to better understand the data, exploratory data analysis was performed with the pre-processed data.

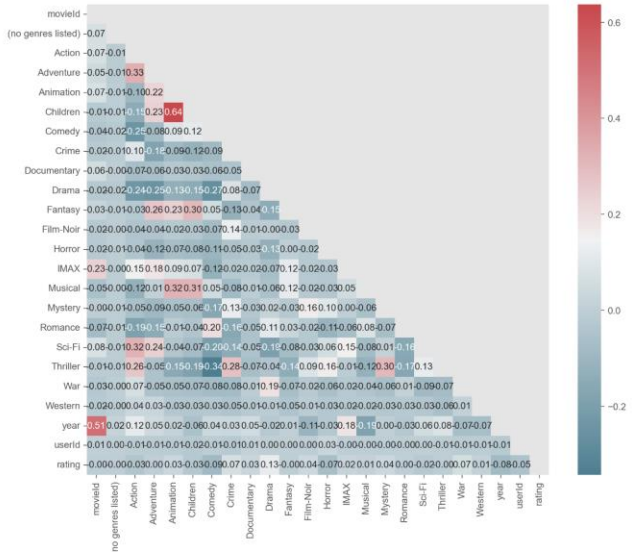

Figure 1. A heat map to better see the relationship between genre features.

Inferences to be made, Animation and Children genres are highly correlated in this data. Action and Adventure, Fantasy and Children, Sci-Fi and Action genres are moderately correlated. These are likely to be in clusters together in K-Means.

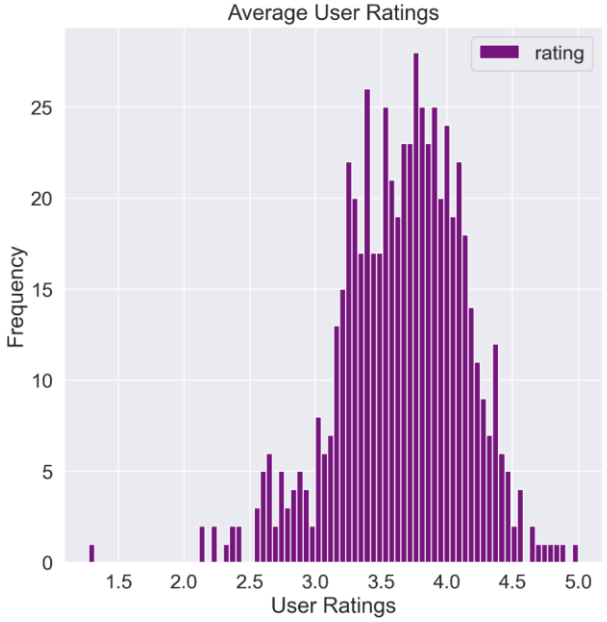

Figure 2. A histogram of the user ratings.

Looking at figure 2, there is a right skewed distribution for the average movie ratings per genre. The central means of tendancy happen between 3 and 4.

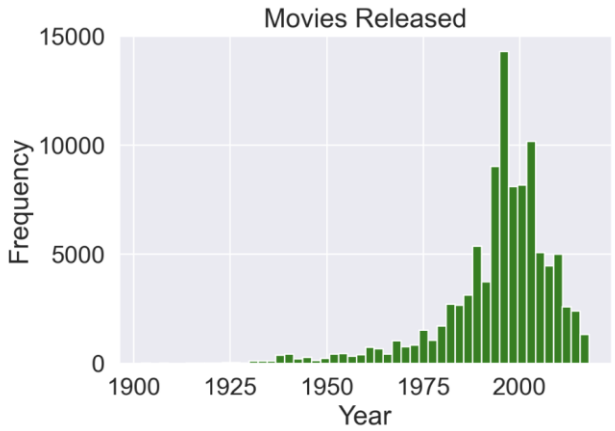

Figure 3. Histogram of movie release years.

From figure 3, it is clear most of the movies released were between 1990-2000.

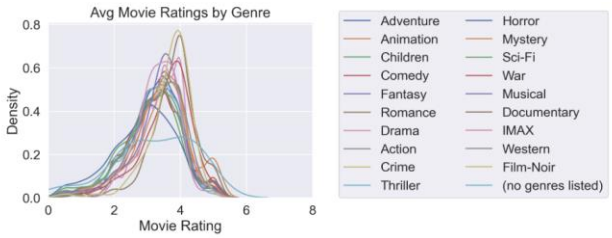

Figure 4. Density curves of ratings per genre.

Using a Probability Distribution Function plot in order to get the distribution of the average movie rating per genre, it is clear that people rarely rate movies lower than 3 stars, which contributes to a right-skewed distribution. The Thriller genre is bit normally distributed.

3.3 Principal Component Analysis

The principal components analysis (PCA) which was used to convert the original variable to a new net of variable would be used to reduce the dimension of cross table above for clustering and visualize. The titles were the target feature. There hundreds of unique movie titles in this dataset so dimensionality reduction was important. For this project, the cross table was projected onto an arbitrarily chosen twenty-dimensional subspace to be able to better reduce to three dimensions. The figure 5 below shows where the variance explained by each principal component tapers off at the components.

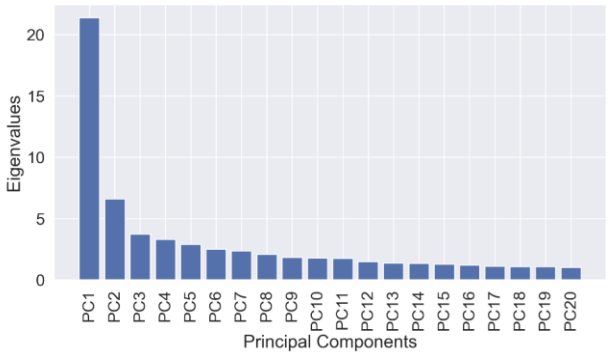

Figure 5. Principal components.

After PC3, visually it is clear the amount of variances explained by two or three or more principal components very steadily declines. Three principal components were chosen.

3.4 K-Means Clustering

The Elbow method was used for determining the optimal number of k clusters. From the given data points that were generated from the previous step, by applying the elbow method with the different values of k from 1 to 9, the optimal number of k fell in the range of 2 to 4 illustrated in the figure 4 below. Intuitively, the elbow was not clear enough for choosing the best value for k-number of clusters since we do not always have a clear cluster data. Thus, for this dataset the value of k could be either 3 or 4. 4 was chosen.

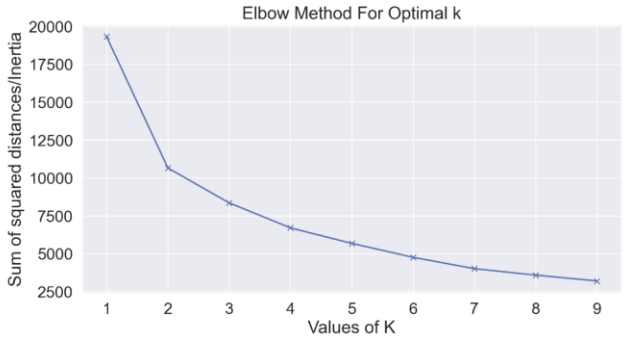

Figure 6. illustrates the optimal k for K-Means using the Elbow method.

Once the optimal k of 4 was found, the K-Means clustering was applied to predict which cluster that each user would belong to from the data points on 3-dimensional subspace. The figure 7 be low illustrated 4 different clusters on 2D plot, and each color represented for each cluster along with their centroids.

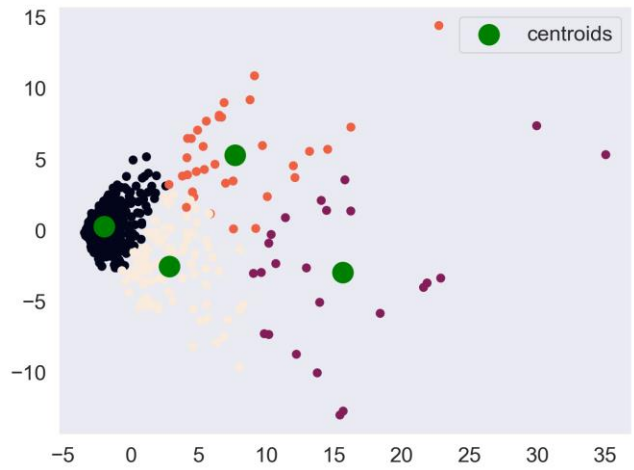

Figure 7. The clusters and centroids on a 2D plot.

From the result of K-Means algorithm, the characteristic of each cluster could be easily identified by getting the genres of the top 20 movies in each cluster. This could be done by assigning cluster number to each user in the cross table above and collected all the movies that each user had watched in each cluster and compute the weight for each movie in the cluster. The top 20 movies were selected by sorting the weight in descending order. The figure 8 below illustrated the top 20 movies for the first cluster.

```
title
Shawshank Redemption, The (1994)              0.453
Forrest Gump (1994)                           0.436
Pulp Fiction (1994)                           0.426
Silence of the Lambs, The (1991)              0.387
Braveheart (1995)                             0.334
Matrix, The (1999)                            0.330
Schindler's List (1993)                       0.317
Jurassic Park (1993)                          0.306
Apollo 13 (1995)                              0.289
Star Wars: Episode IV - A New Hope (1977)     0.287
Terminator 2: Judgment Day (1991)             0.283
Fugitive, The (1993)                          0.266
Toy Story (1995)                              0.255
Batman (1989)                                 0.251
True Lies (1994)                              0.247
Usual Suspects, The (1995)                    0.245
Dances with Wolves (1990)                     0.243
Seven (a.k.a. Se7en) (1995)                   0.238
Independence Day (a.k.a. ID4) (1996)          0.236
Fight Club (1999)                             0.234
dtype: float64
```
Figure 8. First cluster and its top 20 movies.

By applying the algorithm above, the characteristic of each cluster was found.
From the result, the top 5 popular genres in each cluster are showed in the table below:

Cluster 0	Cluster 1	Cluster 2	Cluster 3
Thriller: 580676 Drama: 522369	Adventure: 434165 Action: 359488	Action: 486496 Sci-Fi: 434832	Drama: 456278 Action: 431508
Action: 459007 Crime: 397440 Adventure: 305655	Fantasy: 23291 Sci-Fi: 213468 Comedy: 151165	Thriller: 396744 Drama: 345772 Adventure: 314650	Adventure: 387642 Comedy: 306521 Crime: 302073

From the output, a recommendation system could be created to suggest which movie to watch for a user by finding the movies that user has not watched and movies that user has watched from the cluster the user belonged to. Using the information from ratings.csv file, the list of unwatched movies of a user could be retrieved to filter out the movies that user has watched in the corresponding
cluster.

Algorithm to find recommended movie to user:
def final_recommendation_system(userId):
find cluster id which user belongs to

retrieve a list of movies that user has watched from ratinggs.csv

retrieve a list of movies that user has not watched from r atings.csv

retrieve the top 60 movies in the cluster which user belo ngs to

result = filter out movies that user has watch in the top 6 0 movies

return result

The figure 9 below shows an example result from the experiment.

```
0
cluster0
                                          Title  movieId
0             Shawshank Redemption, The (1994)      318
1                         Forrest Gump (1994)      356
2                         Pulp Fiction (1994)      296
3            Silence of the Lambs, The (1991)      593
4                          Braveheart (1995)      110
5                           Matrix, The (1999)     2571
6                       Jurassic Park (1993)      480
7                           Apollo 13 (1995)      150
8   Star Wars: Episode IV - A New Hope (1977)      260
9          Terminator 2: Judgment Day (1991)      589
10                      Fugitive, The (1993)      457
11                          Toy Story (1995)        1
12                             Batman (1989)      592
13                          True Lies (1994)      380
14                Usual Suspects, The (1995)       50
15                 Dances with Wolves (1990)      590
16               Seven (a.k.a. Se7en) (1995)       47
17       Independence Day (a.k.a. ID4) (1996)      780
18                          Fight Club (1999)     2959
19                     American Beauty (1999)     2858
20                              Aladdin (1992)      588
```

Figure 9. A list of recommended movies to user whose id was 3.

3.5 Collaborative Filtering

User-User Collaborative Filtering and Item-Item are two 2 types of Memory based Collaborative Filtering. In this project, I used User-User approach to find the similarity of all users to the active user – the user whom the prediction was for. To calculate the similarity, the similarity matrix was needed and created by merging the data frames of ratings.csv and movies.csv into one data frame and filled out the row wise NaN's (converted to 0) with the corresponding user 's mean ratings. This also mean, the average ratings were assigned to the movies that were not rated, and the Pearson Correlation could be carried out. The figure 10 showed the correlation matrix of all users. Each cell in the matrix represented the correlation value of a pair of users among 610 users.

title	10 Things I Hate About You (1999)	12 Angry Men (1957)	2001: A Space Odyssey (1968)	28 Days Later (2002)	300 (2007)	40-Year-Old Virgin, The (2005)	A.I. Artificial Intelligence (2001)	Abyss, The (1989)	Ace Ventura: Pet Detective (1994)	Ace Ventura: When Nature Calls (1995)	...
userId											
1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	4.000	0.000	0.000	...
2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	...
3	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	...
4	0.000	5.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	...
5	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	3.000	0.000	...

Figure 10. Correlation Matrix of 610 users.

From the data in correlation matrix above, the predicted rating score for an unseen movie could be predicted by using the following algorithm:

Algorithm for calculating the predict rating score	
1	Initialize the sum of correlation to 0
2	Initialize the rating score to 0
3	Retrieve the list of correlation values of all users except the active user X
4	Retrieve the correlation of top N users to active user X
5	For each user in top N users
6	Rating += correlation value of user * rating of movie Y
7	Sum of correlation += correlation value of user
8	Return predicted_score = rating/sum of correlation

The predicted rating score was computed by the formula:

> Predicted Rating Score = Sum of [weight*rating]/ sum of weights
> Weight is the correlation value of corresponding user to the user X.

Here was an example result from the experiment to find the predict rating score for a movie that user with id=222 had not watched. The similarity of all other user to user who had user id = 222 from the correlation matrix showed in the figure 11 below.

```
userId
224    1.000
367    0.343
275    0.322
201    0.303
597    0.293
       ...
18    -0.084
73    -0.085
139   -0.091
405   -0.098
393   -0.111
```

Figure 11. The similarity of all other user to user 222 from the correlation matrix.

From the result of finding recommended movies in the previous section, the user (userId= 222) had not watched the movie "Forest Gump (1995)" which had movieId = 356.
The predicted rating score for that movie was 4.36, and this was closed to the actual mean rating which was 4.16.

4 Analysis

The accuracy of prediction and some challenges that I observed after completing the experiment would be discussed in this section.

4.1 Results

In this project Root-Means-Square Error was used to evaluate the result of prediction. By recording predicted rating scores and actual mean rating scores of 200 ratings from 16 random users. The RMSE value was approximate 0.19.
This value indicated that the model could relatively predict the rating score accurately.

4.2 Challenges

These unsupervised learning techniques worked well on small dataset. For this project, I used MacBook Air M1 with 8GB RAM, and the time for processing data, finding recommended items and prediction took less than 8 minutes. However, there were issues when trying the larger 25M dataset and I was not able to create a cross table let alone exploratory data analysis.

Another challenge should be mentioned was finding the optimal k value for K-Means clustering. I used Elbow Curve method but the optimal k was moderately obvious but not extremely obvious. The optimal k was between 3 and 4 but arbitrarily, 4 was chosen.

5 Conclusions and Future Work

A movie recommendation system could be built by applying K-Means clustering and memory based collaborative filtering. These techniques would help in finding the best movies using rating and genres, and it would be able to predict the rating score for unseen movies by finding the best similar users. However, it would need a stronger computing system to work on larger dataset. For future work, since K-means and collaborative filtering are two unsupervised learning techniques, the movie recommendation system is not supervised, and it will be good to apply some supervised learning tasks or use the other features in the dataset to cluster users properly.

REFERENCES

[1] J.S. Breese, D.Heckerman, and C.Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the Fourteenth Conference on Uncertainty in Artifical Intelligence, 1998.

[2] Harper, F.M., & Konstan, J.A. (2015). The MovieLens Datasets: History and Context. ACM Trans. Interact. Intell. Syst., 5, 19:1-19:19.

[3] Harper, F.M., & Konstan, J.A. (2015). The MovieLens Datasets: History and Context. ACM Trans. Interact. Intell. Syst., 5, 4:1-19:19.z

[4] Pulkits. (2019, August 19). The Most Comprehensive Guide to K-Means Clustering You'll Ever Need. Analytics Vidhya. Retrieved April 16, 2022, from https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/

[5] Byström, H. (2011). Movie Recommendations from User Ratings.

http://cs229.stanford.edu/proj2013/Bystrom-MovieRecommendationsFromUserRatings.pdf

[6] Singh, Ashwani & Soundarabai, Paulsingh. (2017). Collaborative filtering in movie Recommendation System based on Rating and Genre. IJARCCE. 6. 465-467. 10.17148/IJARCCE.2017.63107.