# CS910 Project：  Movie Recommendation System

ID: 2291588

**Abstract**

The movie recommendation system is a popular data science topic and is important in the business world. Three main recommendation methods are demographic recommendation, content-based recommendation, and user-based recommendation. This essay provided examples of these three recommendations. Simple grouping and encoding method in python and R was used to make a demographic recommendation Classifier such as a naive Bayesian classifier was used to build the Recall strategy. And regression was used to build content-based filtering to predict movie ratings. K-mean clustering was used to make user-based recommendations.  Regression is not suitable for movie recommendation tasks as the movie data set is sparse. The Bayesian classifier works well on recall tasks. K-mean clustering is also suitable for user-based collaborative filtering.  Limitations of this study and an alternative method called factorization machine were discussed.

## Introduction

Analysis of movie data is one of the most popular topics in data science. Film manufacturers can also analyze data to understand what kind of movies will bring profits and become popular. A good movie rating prediction recommendation system can bring huge benefits to online platforms such as Netflix. Netflix set 1 million to reward a 10% improvement in the movie recommendation system‘s accuracy[1] in 2006. This is strong evidence proving the importance of the movie recommendation system.

The main aim of this essay is to find out how regression; classification and clustering can be used in movie recommendations and discuss their drawbacks and benefits. There are three main kinds of recommendation. systems, generalized recommendation; content-based recommendation, and user-based recommendation. This paper will use movie lens and TMDB movie datasets.

The first task is to make a generalized recommendation for all users based on popularity and rating. The relationship between attributes will also be discussed in this part.The second task is to make content-based recommendations using regression. Attributes such as runtime; production country and popularity could be related to the vote. So, we can use these attributes to predict a movie's rating. The third task is to build a recall strategy. In this essay, the data set is small so a recall strategy is not necessary but in the real world, a recall strategy is needed in the movie recommendation system because the recommendation system can not analyze the whole data set every time. So a strategy to get a subset of the whole movie data is needed.  We will use a classifier to classifier which kind of movie is worth to be added in the recommendation pool. The fourth task is using the k-mean cluster to build a user-based recommendation system. The data set is huge but there are only a few pieces of data for each specific user. So collaborative filtering will be used to predict one specific user's rating on a movies rating based on another similar user's ratings on this movie. Then the drawbacks and benefits of using regression; classifiers and clustering in the rating-based movie recommendation systems and the analysis and an alternative method called factorization machine will be discussed.

## Background

**Demographic recommendation -** This method provides generalized recommendations to the user who have similar demographic features. Since every user is different, this approach was considered simplistic[2]. The basic idea behind the system is that the more popular and well-reviewed movies are, the more likely they are to be liked by general audiences. This method is simple but very useful. Many movie websites, such as Netflix's homepage, have this function, they recommend the top 10 latest, most popular, highest-rated movies for all users.Its shortcoming is that it cannot make recommendations considering individual differences and movie features. But it is useful when a generalized recommendation is required such as making the homepage of the website and when the platform has no user behavior data.

**Content-Based recommendation -** This method makes recommendations based on specific factors, such as description, category, actors, etc. The idea is to recommend user content that is similar to the content the user likes[3].There are many possible contend based movie recommendation methods. The most simple way is to recommend movies by categories such as genres and production country.  It also includes using clustering to find similar movies and recommend users “similar to a movie you liked”,  or build a regression model using movie attributes to predict the rating.The drawback of this method is user behavior data and movie attribute data are usually scarce sets. It has to face the cold start issue. Most of the categories or contents are unvisited,  so it is hard to make recommendations for a new user or recommend unvisited content[4]. For example,  the website has many data about a user’s attitude towards Comedy but no data about Thriller movies. It would be hard to find out the user‘s attitude toward thriller movies using content-based recommendations.

**User-based recommendation-** This is also called collaborative filtering in the recommendation, this recommendation method uses methods such as clustering to find similar users.  and then recommends movies that similar people in the same group [5]. It can recommend areas that users have never touched. It assumes the user will like the same movie liked by other similar users. It can solve the item cold start problems as the system can make recommendations based on other people’s attitudes on an unvisited item.The demographic recommendation is the most basic recommendation method and Content-based recommendation and user-based recommendation are the two most used recommendation methods. They all have drawbacks and benefits and they are suitable for different situations.

There are more advanced ones such as hybrid recommendations which combined both collaborative filtering and content-based filtering [6]. One example is the factorization machine.  This will not be used in the analysis part but will be discussed in the discussion.

## Data& Cleaning

### Data description

This essay used The movie Len data set on Kaggle[7]. The original data were came from TMDB and Movie Lens website.

**1) TMDB dataset**

The file contains 45, 000 movie details in TMDB released before 2017 July[8], including cast, crew, plot keywords, budget, revenue, posters, release dates, languages, production companies, countries, TMDB vote counts, and vote averages.Most of the attributes will not be used in the analysis so only used attributes used in this analysis will be explained.

**Attributes: id:** Unique movie id
**budget:** The movie’s budget in dollars
**revenue:** The movie’s revenue in dollars
**production countries:** The countries where the movie was made
**popularity:** Popularity on TMDB is generated by several weighted attributes [9]
**vote average:** The average voting of the movie from 1-10 rated by the registered users on TMDB.
**vote amount:** the number of voting received.
**release date:** Date of release
**title:** movie title
**runtime:** the length of the movie in minutes
**genres:** genres of the movie such as Action comedy, Animation etc.

**2) Rating data set:** This data set includes 27,0000 users' ratings on 45,000 movies from MovieLens.  There is a total of 26 million movie ratings.  This data set will only be used in the clustering part. This essay will use a subset of the rating file including 100,004 ratings from 641 users on 9000 movies.

**Attributes:**
**User ID:** the user id of each unique user
**Movie ID:** unique numerical ID for each movie
**Rating:**  the rating of the movie from 0.5-5 given by the registered movie lens user, the gap is 0.5. Note: The voting method for TMDB is different from the rating Movie lens here.

### Data Cleaning:

**Rating data set:**

This data set has no outlier and missing value. The range of rating is 0.5-5 and the number of Null object is zero.

**TMDB data set:**

**Editing the data:** There are 45466 data in this data set we only need several lines in the CSV file but we can not delete the unwanted attributes in CSV file because the file is large and editing the file in Excel will cause an error. I used data frame. drop in python to drop these attributes.

**Missing value:** After checking the data set, I found out some objects in the budget attribute and revenue attribute are not numerical objects, these should be changed to null, also the budget and revenue should not be zero. If these values are zero it means they are missing values. So I changed all budget and revenue=0 to null. The following figure from python shows the number of non-missing values for all attributes in python. Many budget and revenue values are missing but most of the other attributes are non-null. So I created two data set. The one with budget and revenue and the one without budget and revenue because budget and revenue will only be used in a few cases. Then I deleted all lines with null objects

There are 42275 complete objects for all attributes without budget and revenue. This data set will be used in recommendations without using budget and revenue. Missing values can be deleted as the data set is huge. And we can not fill these values because some of them are categorical variables. There are 5360 non-null objects if we include the budget and revenue as there are many missing values of these two attributes. But we can not fill them with average because this makes no sense.

**Outliers:**

**Vote average:** The distribution of rounded votes is shown in Fig.1 .The vote is range from 1-10. So there should not be an average vote smaller than 1. It is weird that the vote average has zero value and value smaller than 1. The website will wrongly allow some users to give 0 ratings. Sometimes vote account will be recorded when the user didn't give a rate/ didn't finish the rating. A discussion of this default can be found on TMDB's website.[4]Objects where the average vote is zero but the vote count is zero has been deleted.

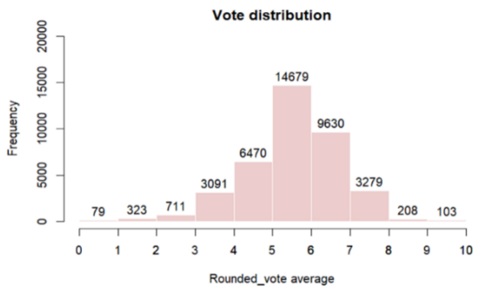


Fig 1. Rounded_ vote average

**Popularity :** As the left side of Fig2 shows, popularity has a large range but most of the values are around 10. So, log(popularity) was used to replace popularity, and the popularity value is larger than 5 or smaller than -2. The distribution of popularity after editing is shown on the right side of Fig 2. Now there is no outlier.

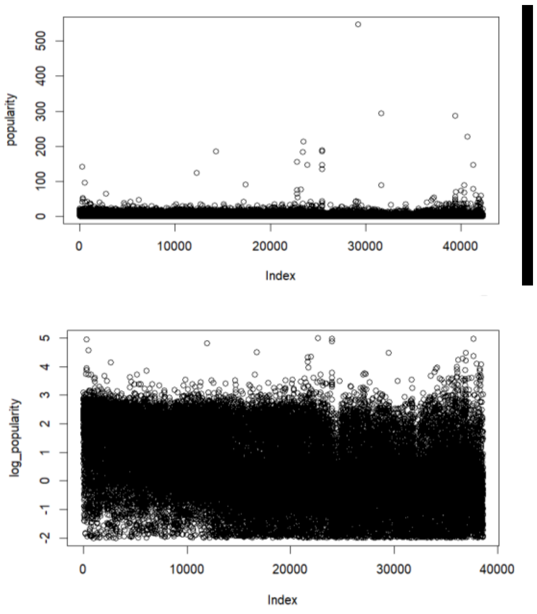


Fig 2. Popularity's distribution & log popularity after removing outliers

**Other Data pre-process:**

**Return:** Each movie has a different budget and revenue. To compare them we must build a new attribute called return. Return=revenue/budget.

**Timestamp:** The original release date is dd/mm/yyyy it has to be transferred into a timestamp to be used in the classifier and regression model.

**Country and genre:** The type of country is ISO 3166 but the default coding type is UTF-8. So the encoding type has to be changed. Also, when editing genre and country, the ' was changed to " so the JSON function in R language can correctly read these. Some country names such as the People's republic of China includes ' and the R language will report an error. So' was deleted. There are some further data processes in the analysis part.

## Data processing

**Task1 Generalized recommendation**

A generalized recommendation based on popularity and rating was made. Fig 3 shows the Top 10 highest-rating movies when the vote count is larger than 250. Because the average vote value is not credible when the vote count number is small. The vote count was also shown in the graph. Fig 4 shows the Top 10 highest popularity movies ranked by vote average. We can see that the Top 10 popular movies don't all have a high rating. The dark Knight and pump fiction are the only two movies both in the top 10 popular and top 10 rating lists. This suggests that high-rating movies are not always popular. The correlation coefficient between popularity and vote average is 0.1604938， which is also weak.

“The circle”has high popularity score because it was published in 2017， which is close to the date this data set was created. It is possible that the popularity score is highly dependent on time. But the correlation coefficient between popularity and timestamp is only 0.11. Showing the correlation is weak. The correlation coefficient between popularity and vote_count is 0.48， which is the strongest among all attributes in the data set. The correlation coefficient between rating and return was also checked. The correlation coefficient is only 0.015 which is close to zero. this suggests that rating is not an important factor in predicting whether the movie is profitable.

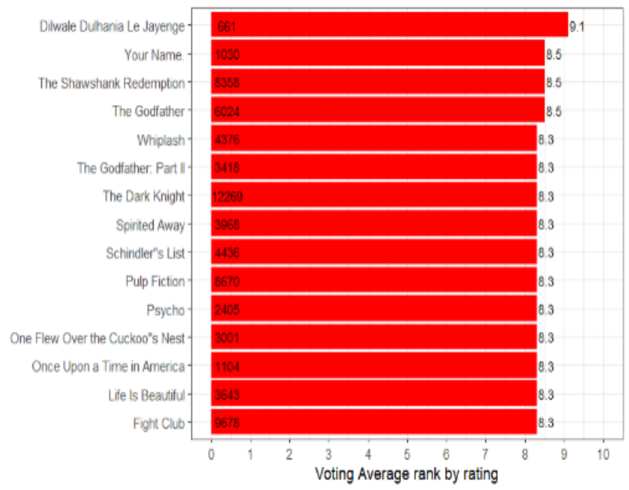


Fig 3. Vote average rank by rating

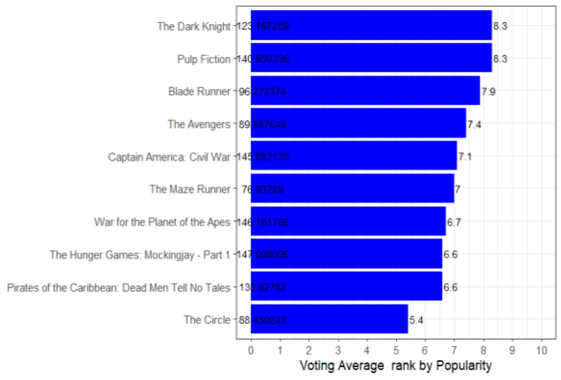


Fig 4. Vote average rank by popularity

**Simple content-based recommendation**

This method recommends movies simply based on the production country and genre. Some coding in R language was used to build a new attribute called type and country to get the production country and Genre type. In this part I only use a vote average larger than 250 as a vote average in a small vote number doesn’t make much sense.

The most popular types are Drama; Comedy; Thriller and Action(Fig 5.)

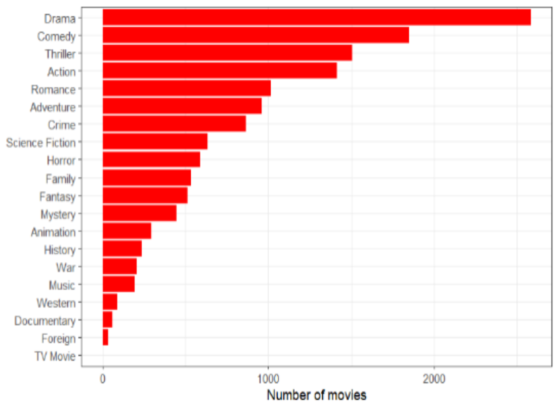


Fig 5. Number of movies in different genre.

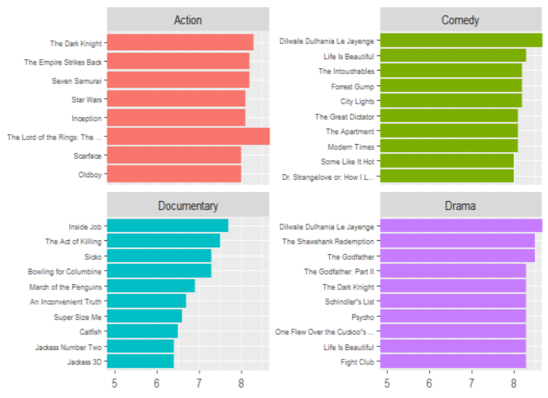


Fig 6. Recommendation by Genre

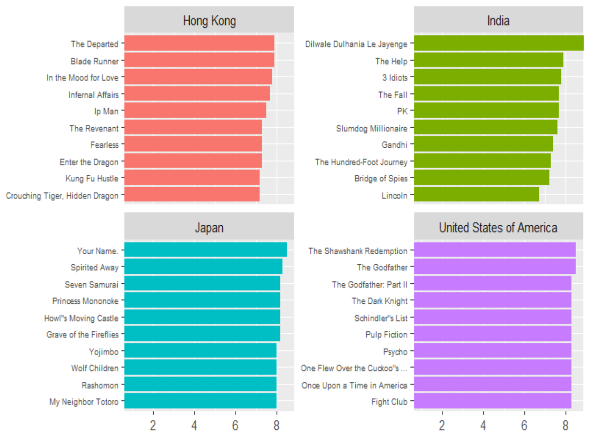


Fig 7. Recommendations by production country

Some recommendations by genre were shown in Fig 6. The top 10 recommendations in the Drama section have high rating values and the top 10 in the Documentary has lower rating values, which suggests that there are fewer high-rating documentaries. The rating trend recommendation based on production countries was shown in Fig7. The top 10 movies produced in India have a lower rating than the Top 10 movies produced in the US.

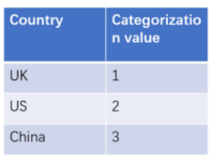
**Use regression model to Predicting rating**

A multiple linear regression model without interactions was calculated to predict the Vote average using genre; production country; popularity; budget and timestamp to predict rating. This data set selects vote account >1000 to reduce the number of data and make the analysis easier, there are 4562 data in this data set.

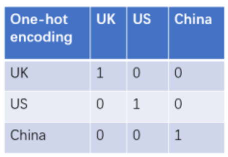
The model includes too many predictors as production country and genre are categorical variable with many levels. There are totally 63 predictors in this model. This model's F statistics and R-square is: $R^2=.75$, $F(63,4498)=43.22$ , $p<.001$ log(Popularity) is a significant predictor for vote-average,$t(4498)=15.336,p<.001$.Run time is also a significant predictor for Other-rating,$t(4498)=26.929,p<.01$.

There is no place to report other attributes' t-statistics. But timestamp and budget are significant negative predictors for vote_ average.One explanation could be recent "Hollywood" movies usually have high budgets but old movies on websites with the lower budgets are usually high quality. or they will vanish after a long time. The correlation coefficient between budget and timestamp is 0.3, suggesting they are positively related.For genre, genres such as drama and comedy are positive predictors of vote average. Horror is a negative predictor of vote average. As for countries, Denmark is an example positive significant predictor of vote average and Thailand is a negative significant predictor of vote average.

Regression is not a good prediction method for movie recommendation. Most recommendation system will avoid using regression because the data usually includes many categorical variables which usually needs one-hot encoding. An example of one hot encoding is in Fig 10, we can not put the country in one variable because we have to make the distance between different categorical variables the same.  If we write like left table in Fig 10, then 3UK=1 China and 3UK=1 US. This makes no sense in a regression model.



| Country | Categorization value |
|---|---|
| UK | 1 |
| US | 2 |
| China | 3 |



| One-hot encoding | UK | US | China |
|---|---|---|---|
| UK | 1 | 0 | 0 |
| US | 0 | 1 | 0 |
| China | 0 | 0 | 1 |

Fig 8. One hot encoding.

After the one hot encoding, the data set will become very sparse. For example , there are 4562 data in this data set but there is only 1 Thailand movie called  <Mechanic: Resurrection> the vote average is 5.3.There is only 1 Brazil movie called <City of God> the vote average is 8.2 . That is why brazil is a positive predictor and Thailand is a negative predictor. But this makes no sense.  So normal regression model can not deal with data sets with many categorical variables. It also can not deal with the interaction between factors. We usually want to know the interaction between predictors in the Movie recommendation system. But adding interaction has high complexity. Imagine we only want to use production countries to predict the vote average(Sometimes there are more than one production country for a movie). If we have 200 countries there will be n(n-1)/2 interaction items. The complexity is $O(n^2)$.In conclusion, regression is not a good method for movie recommendation.

**Using Classifier to build s recalling system**

The movie recommendation system required a recalling strategy because in the real world, the recommendation analysis cannot be used on the whole data set. Because the whole data set is very We need to filter a subset of data and perform the analysis on the small data set. I want to build a classifier to get only movies worth recommending. I set the worth recommend the movie as vote average > 6.7  so the data set will be balanced.  In this case， the data set has 4562 total movies， 2100 of them will be worth recommending and 2462 of them will be not worth recommending. Attributes are budget; log(popularity), production country timestamp runtime, and genre.

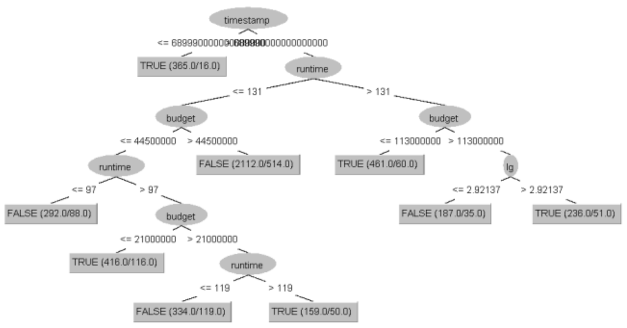


Fig 9. J48 decision tree.

The classifier was built based on 66% training split.

The aim is to reduce the false positive rate because missing good movies is fine, but we don't want to recommend bad movies. The first one is the ZeroR classifier. The accuracy is 54.9% and the FP rate is 0. It is useless because if the number of the unworthy movie is large it will recommend nothing as everything will be classified as unworthy. If the number of the worthy movie is higher, then the FP rate will be really high.The second one is the Naïve Bayes classification. The accuracy is 73.5% and the FP rate is 0.160.The third one is neighbor classifier (lazy/IBk) the best kNN is 7 the accuracy is 73.4% The false positive rate is 0.204.SVM classification was also used. The accuracy rate is 74.0% The false positive rate is 0.192.The heaviest weight attributes are timestamp 4.84. The last one is the J48 classifier accuracy of J48 when the min number of the object is 100 is 75.8% and the false positive rate is 0.211. The root is just the heaviest weight attribute timestamp in SVM. Budget is also a heavy-weighted attribute weighing 3.69.

The best classifier is the Naïve Bayesian classifier. The accuracy is 73.5% it is lower than the J48 classifier but it has the lowest False positive rate which is 0.160 I think a lower False positive rate is more important. In conclusion, the classifier using these attributes can work well as a recall strategy.

**Content-based collaborative filtering**

Using methods such as clustering to make content-based recommendations is a better way. Put similar items in clusters and make predictions based on clusters. The simplest way is to predict a new movie's rating using the mean value of other similar movie's mean ratings. This uses the same method as user-based collaborative filtering. So only one analysis will be conducted.

**User-based Recommendation**

**K means Clustering**

This section uses Kmeans clustering to predict the rating. The simplest way was used in this analysis: Build a cluster to find similar users and use users' average ratings on a specific movie in the same cluster to predict movie ratings. Drama and Comedy are used as two attributes. To make things simple, assume there are only two kinds of users. drama lover and comedy lover. User who both likes drama and comedy was deleted (mean Rating of drama >3 and rating of comedy >3). Each user's drama rating and comedy rating are shown in Fig10.

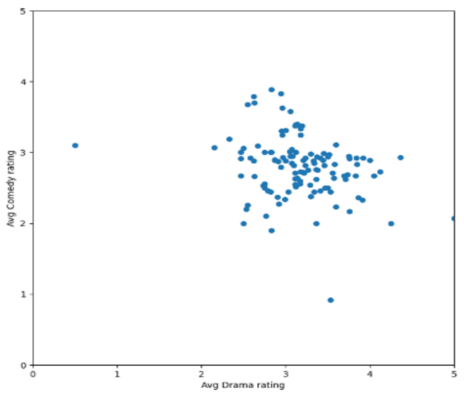


Fig 10. The plot of the average drama rating and average comedy rating

Kmeans function from sklearn.cluster in python was used to cluster the user. At first, the number of clusters was set as 2. The result of clustering is in Fig 11. The yellow cluster could be explained as Comedy lovers and the purple cluster could be explained as Drama lovers.

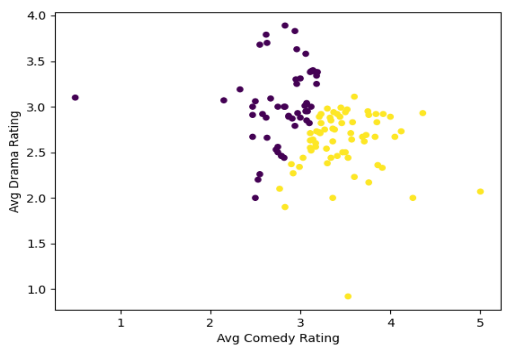


Fig 11. Cluster result Number of centers=2

The clustering result when the number of centers is 3 is in Fig 12. The clusters could be explained by drama lover= yellow; Comedy lover =green. Hate both=purple. Finding a number of k is important in K-means clustering. Silhouette Score and the elbow method are examples of methods that can find the best cluster parameter. When more attributes were added, it could be harder to conclude what is the meaning of clusters and when attributes >3, it is hard to visualize the cluster.

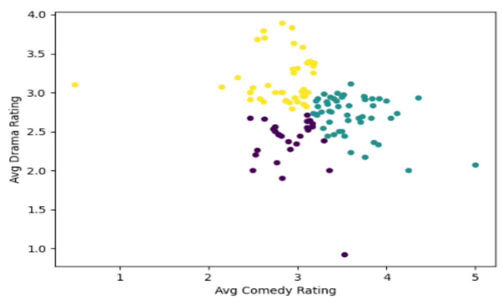


Fig 12. Cluster result Number of centers=3

The example here is the simplest collaborative filtering. More user attributes and movie attributes can be added in the model. Collaborative filtering can solve the item cold start issue and works well in sparse sets. It can also reduce complexity. In latent variable collaborative filtering, Instead of finding the direct relationship between all the user attributes and movie attributes. We only need to find the relationship between user attributes and limited k numbers of latent variables then find the relationship between k numbers of latent variables with movie attributes. This will reduce the complexity from $O(n^2)$ to $O(kn)$. where n is a lot larger than k.

The process of k-mean clustering is similar to finding the relationship between user attributes and latent variables. User attributes were used to form user clusters and then find the relationship between clusters and movies. In this case, the relationship between cluster and movie rating was simplified as the mean rating of cluster means rating on the specific movies. The method used in this section is simple but it clearly shows how collaborative filtering works.

## Discussion & Conclusion

At first, the generalized recommendation was made based on popularity and rating. Ratings are not strongly correlated with popularity, nor does the timestamp. but vote count is strongly correlated with popularity. A regression model using genre; timestamp; production country; runtime and log(popularity) as predictors was built to predict the vote average. the model's F statistic is significant and it includes many significant predictors.
But after checking the categorical variables, A conclusion was drawn that the regression model is not suitable to deal with the movie data sets as there are many categorical variables that require one-hot encoding. The data set would become very sparse. The result of the regression model would become not credible and inaccurate. In addition, this can not add an interaction item because the complexity is O(n2) which is very high. A classifier using the same attributes was built as a recall strategy. It works well. The best classifier is the Naive Bayesian classifier as it has the lowest False positive rate. Collaborative filtering is a better recommendation method compared with regression. A simple collaborative filtering example using k-means clustering was conducted to predict ratings by members' average rating of movies in the same cluster. It can work well in sparse data sets such as movie data set and the algorithm has low complexity. The drawback is everyone in the same cluster will get the same predicted value, it cannot add other attributes as a predictor to predict movie ratings like regression.

**Limitation and Future study:**

1)An alternative method called factorization machine can combine the collaborative filtering and regression model Factorization machine, produced by Rendle in 2010[12]. It can be seen as a combination of regression models and collaborative filtering by replacing the interaction items of the regression model with latent variable collaborative filtering. This method can both add attributes as predictors and works well for sparse data set and reduce complexity. This method and its advanced version are one of the most commonly used recommendation algorithms.

2)Most of the models are based on ratings. But the rating is not highly correlated with popularity and profit. The method mainly using RMSE value to judge a recommendation system could be questionable. Movie recommendation systems are mostly designed for business purposes but not academic purposes. Movie recommendation systems should not only rely on predicting user ratings.

3)This data set only contains attributes at a specific time. The popularity: rating might change over time. Data including attributes in different times could help build a better recommendation model.

4)Most of the analyses were simplified because the laptop is not a good device to run the model. It cannot run models with high complexity. Future studies can add more cluster centers and attributes but not only comedy and drama to make a better model.

## Reference

[1]Bennett, J., & Lanning, S. (2007, August). The netflix prize. In Proceedings of KDD cup and workshop (Vol. 2007, p. 35).

[2]Rastogi, S., Agarwal, D., Jain, J., Arjun, K.P. (2022). Demographic Filtering for Movie Recommendation System Using Machine Learning. In: Mahapatra, R.P., Peddoju, S.K., Roy, S., Parwekar, P., Goel, L. (eds) Proceedings of International Conference on Recent Trends in Computing . Lecture Notes in Networks and Systems, vol 341. Springer, Singapore. https://doi.org/10.1007/978-981-16-7118-0_47

[3]Pazzani, M.J., Billsus, D. (2007). Content-Based Recommendation Systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds) The Adaptive Web. Lecture Notes in Computer Science, vol 4321. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-72079-9_10

[4] Volkovs, M., Yu, G. W., & Poutanen, T. (2017). Content-based neighbor models for cold start in recommender systems. In Proceedings of the Recommender Systems Challenge 2017 (pp. 1-6).

[5]Subramaniyaswamy, V., Logesh, R., Chandrashekhar, M., Challa, A., & Vijayakumar, V. (2017). A personalised movie recommendation system based on collaborative filtering. International Journal of High Performance Computing and Networking, 10(1-2), 54-63.

[6]Lekakos, G., Caravelas, P. A hybrid approach for movie recommendation. Multimed Tools Appl 36, 55–70 (2008). https://doi.org/10.1007/s11042-006-0082-7

[7]Kaggle The movie data set (2017) : LINK https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset

[8]The movie data base (TMDB) LINK https://www.themoviedb.org/

[9]The movie database API Popularity LINK https://developers.themoviedb.org/3/getting-started/popularity

[10]Movie Lens Link: https://movielens.org/

[11]TMDB Zero Value issue : https://www.themoviedb.org/talk/58a1c36ac3a3682351004a50

[12]Rendle, S. (2010, December). Factorization machines. In 2010 IEEE International conference on data mining (pp. 995-1000). IEEE.