CS6220 - Data Mining Techniques
Final Project Information-Preliminary

Team Name: Next Movie

Kalin Pu, Jiazhen Tang, Xiangshi Sun, Liang Han, Yiqin Zhu

Abstract

In this project, the main issues that we would work to solve are movie recommendation system and movie revenue prediction. For better understanding, movie recommendation system would be used mainly for customers, and revenue prediction system for investors.

For revenue prediction, It is an important problem in the film industry that affects financial decisions made by investors. Normally, these predictions are made by statisticals techniques such as regression. With our TMDB movie dataset, we studied the movies released before 2010 and develop a model using regression. In addition, we test the recent movies released after 2010 using this model and try to interpret the major findings.

Movie recommendation system help millions of users narrow the universe of movies to fit their different tastes. It dramatically enhance users' experience by showing users only the movies they would like the best and save their time with movies they won't care about. The objective of this project is to implement a Movie Recommendation System based on TMDB movie dataset that contains around 5000 movies. The system will have the functionality to select and list some movies from the database after the user input the name of a certain movie. The process of building our Movie Recommendation System will include several phases, such as Data Exploration, Data Cleaning, Building the System, Testing and Final Conclusion. The system also allows users to predict movie box offices based on the previous movie's related data regression, which help movie makers and investors make better choices and earn more profits.
By recommending movies that customers would like to purchase with, we would like to have the system learn about users' preferences. If a user rated a movie, the system would recommend similar movies or movies with same genre with rating from 5 to 10 to ensure the quality. By keeping learning users' preference, the system would give better solutions with movies in which type of genre that are most profitable. For example, families' preferences would happen in holiday seasons would give investors good ideas to invest in comedies. With more data, the system would be able to give a solid prediction for investors which kind of movies is worth paying great amount of attention and investments with under different backgrounds.

The final deliverable will be a well-built Movie Recommendation System with the functionality that allows users to input the name of a movie they like. Then the system will make helpful recommendations with a list of movies that the users may like according to the content and characteristics of the entries. It also allows users to predict

certain movie's box offices using regression model based on the analysis of previous movies' dataset.

Introduction

As we all know, the rapid growth of data collection/analysis techniques has led to a new era of information. Many efficient systems are based on data related techniques and this is where recommendation systems come into play. Recommendation systems are a type of information filtering systems as they improve the quality of search results and provides items that are more relevant to the search item or are related to the search history of the user [ref.1].

In our project, we utilize data mining and machine learning skills to implement a movie recommendation system and movie revenue prediction [ref.2]. The system can help users to discover movies and content by analyzing and predicting the user's preference and showing them the movie items that they would enjoy. In addition, the system can predict movie revenue based on the regression model of previous movies' related data, which help movie makers and investors make better choices and earn more profit.

For this project, we are using the TMDB 5000 Movie Dataset. We first go through the general data analysis process and analyze movie-related information from the data sources, such as movie ratings, movie genres, most commented movies, directors, actors, etc. The movie recommendation system is mainly classified into three types--demographic-based filtering, content-based filtering and collaborative filtering. Demographic filtering offers a ranked list of movies to all the users from the highest rating score to the lowest rating score based on weighted ratings formula. Content based filtering suggest similar movies according to a user's input keywords. This one uses movies overviews, or movie cast, crew, keywords, etc, to make recommendations. It will also filter related movies based on the rating scores and number of comments. Since the dataset we used only describe the content of movies, collaborative filtering is not really feasible on this dataset. In our project, we use the first two filtering techniques. In other words, we plan to analyze other movies that are similar to the selected movie by using content-based filtering. If we plan to recommend to all users at the same time, we use demographic-based filtering to work with.

Recommendation system and revenue prediction are very important for users. Recommendation system will dramatically save people's time for those who would like to watch a movie. Furthermore, investors pay huge attention on how profitable a movie

is. Revenue prediction could help investors to see what kind of movies is worth investing. Meanwhile, if we wanted to learn the revenue of movies, we definitely would want to learn about customers. The best way is to analyze data that customers give high ratings and give corresponding recommendations with similar movies. In this way, a link between investors and market has been built up, and we could tell binding up recommendation system and movie revenue predictions are extremely concrete issues that we aim to resolve in this project.

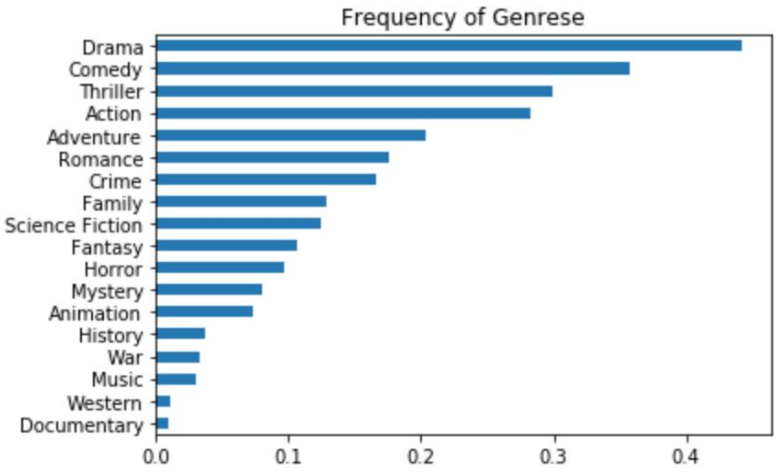
Methodology

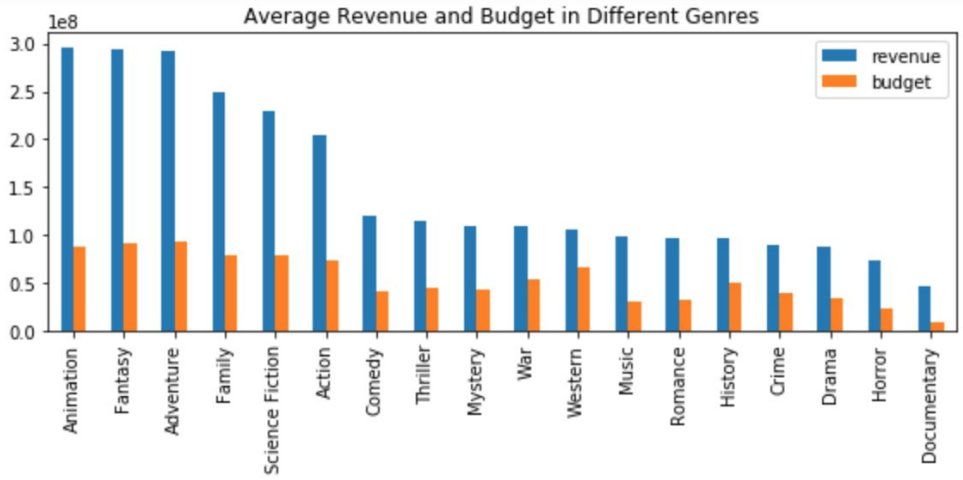
As mentioned in the introduction section, our recommendation system will be implemented using both demographic-based filtering and content-based filtering.

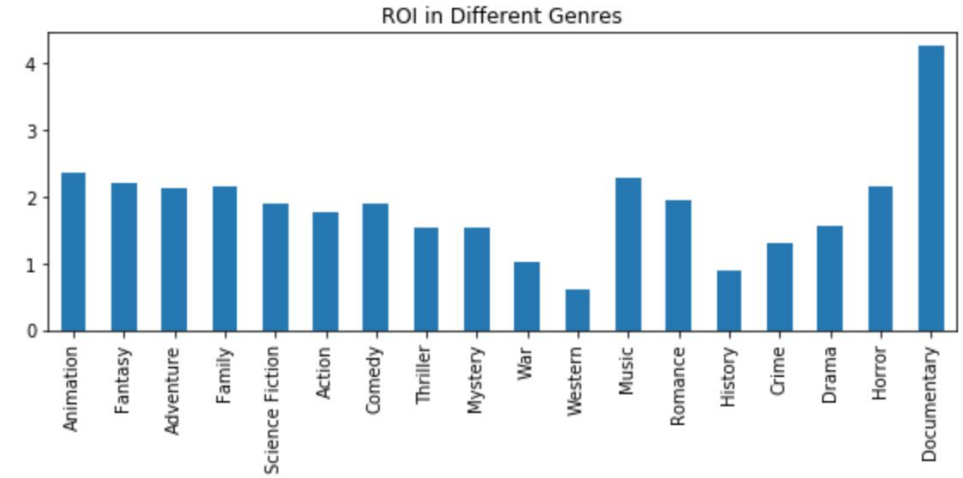
1）initial data analysis and data understanding

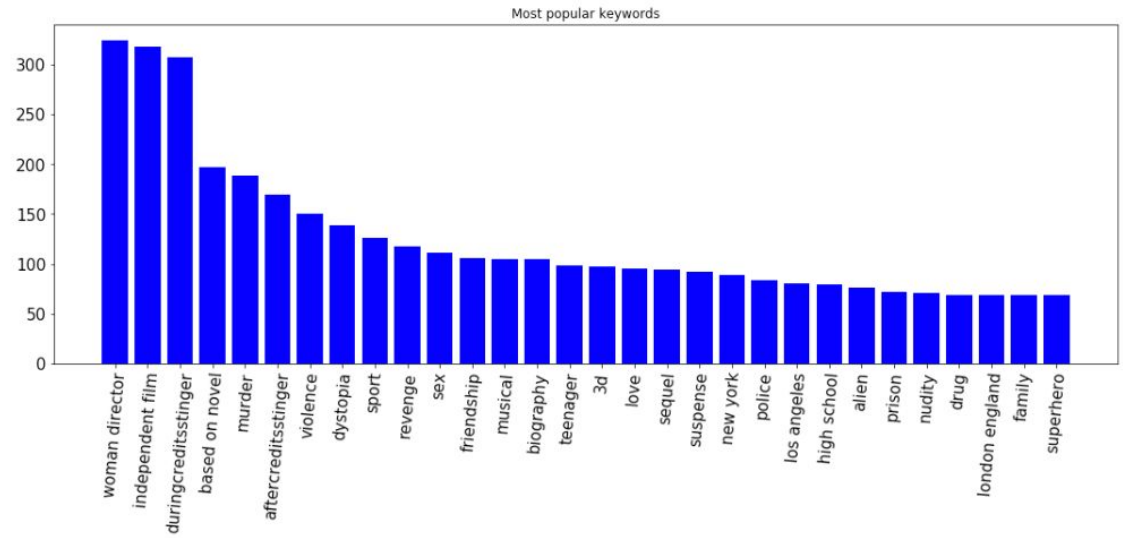
We used TMDB 5000 movie dataset and we load the dataset and transfer the data format from JSON to pandas dataframe. We do a visual exploration of the data to understand what is in the dataset and the different characteristics of the data, as well as information on the columns types in the dataset.

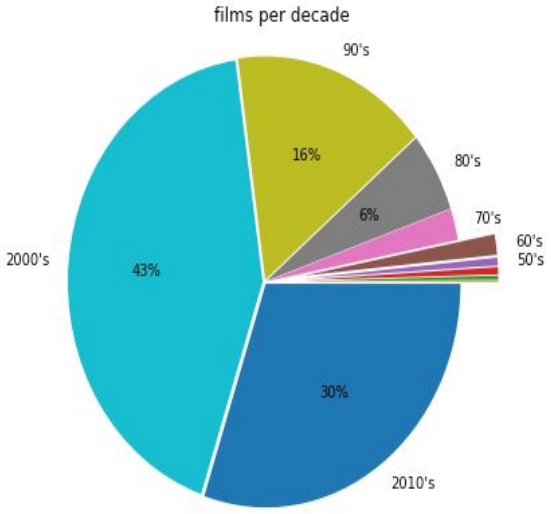
Here we display some important attributes from our data to provide a more intuitive way of understanding the dataset.

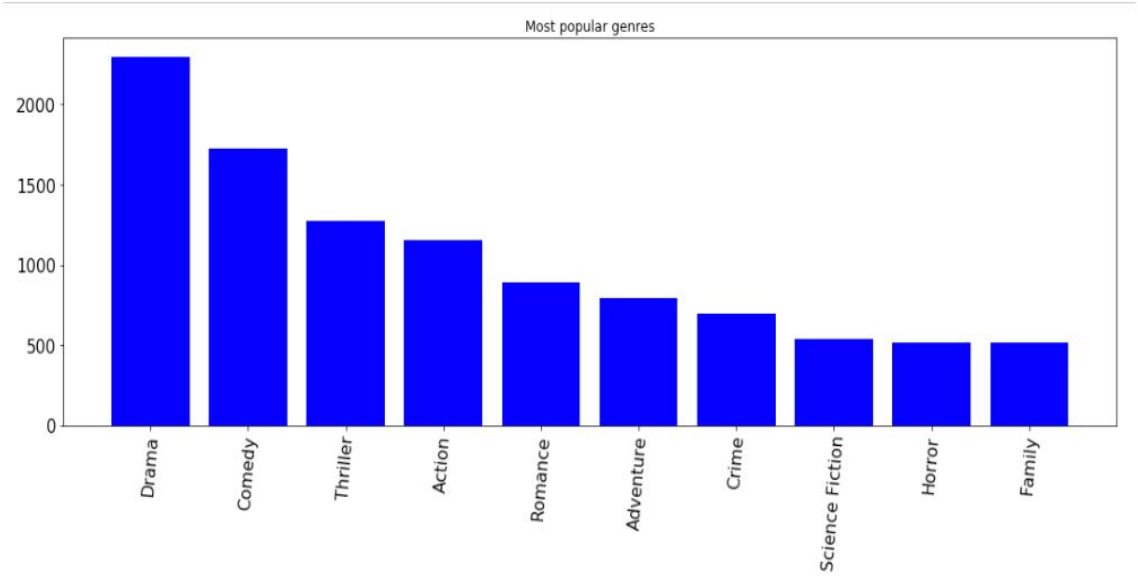


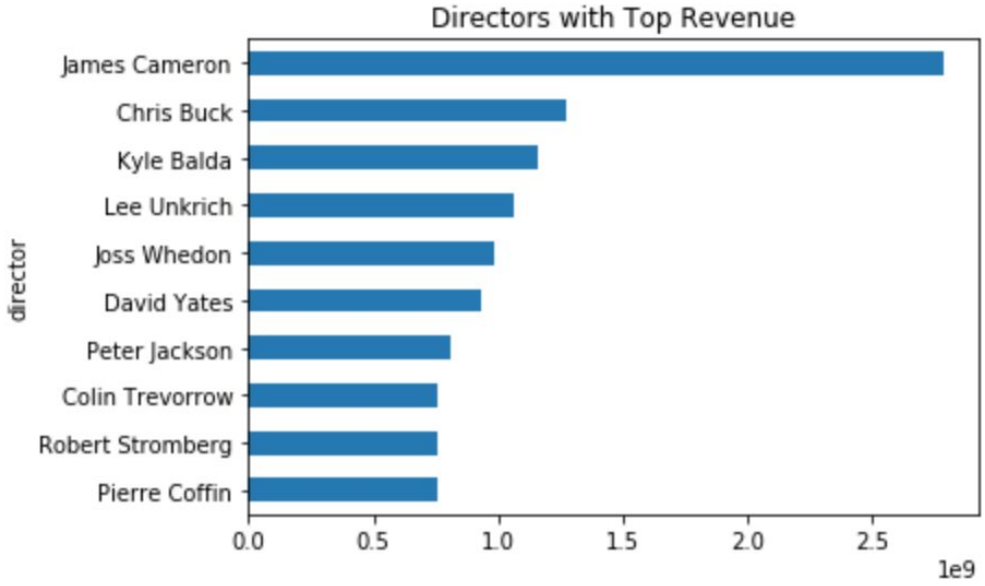

Average Revenue and Budget in Different Genres
1e8
3.0
2.5
2.0
1.5
1.0
0.5
0.0
revenue
budget
Animation
Fantasy
Adventure
Family
Science Fiction
Action
Comedy
Thriller
Mystery
War
Western
Music
Romance
History
Crime
Drama
Horror
Documentary


ROI in Different Genres
4
3
2
1
0
Animation
Fantasy
Adventure
Family
Science Fiction
Action
Comedy
Thriller
Mystery
War
Western
Music
Romance
History
Crime
Drama
Horror
Documentary


Most popular keywords
300
250
200
150
100
50
0
woman director
independent film
duringcreditsstinger
based on novel
murder
aftercreditsstinger
violence
dystopia
sport
revenge
sex
friendship
musical
biography
teenager
3d
love
sequel
suspense
new york
police
los angeles
high school
alien
prison
nudity
drug
london england
family
superhero


films per decade
90's
16%
80's
6%
70's
60's
50's
2000's
43%
30%
2010's





2）demographic-based filtering

In demographic-based filtering, the movie recommendation system observes the common attributes of the users and suggests movies to the users with similar attributes. Recommendation movie system offers generalized recommendations to every user, based on movie genre.

$$\text{Weighted Rating (WR)} = \left(\frac{v}{v+m}.R\right) + \left(\frac{m}{v+m}.C\right)$$

3) content-based filtering

a. Movie Overview

In the movie recommendation system, the content of the movie such as overview is used to find the similarity with other movies. We use TF-IDF vectors for each overview, then calculate the cosine similarity between each other movies, we sort the cosine similarity score for each movies from high to low, and recommend the similar movies to the users.

b. Second approach, the detail methodology is using Movie Metadata

(1) Using more metadata specified and related to the user's like movie would greatly increase the accuracy that the movies we suggest and the users would like.
(2) 'genres', 'keywords', 'credits' and 'production_companies' data are used as our features to generate the similarity score for the movies together.
(3) Clean the data for feature columns we select so that each specific feature element can exist in an overall features paragraph.
(4) Apply cosine similarity matrix techniques to the features paragraph we generated for each movie in our step 3.
(5) The cosine similarity matrix will give us a score for all the movies about their particularly similarity compared with all the other movies.
(6) We retrieve the similarity score for the movie given by the user and we sort its similarity score list from high to low and recommend the top similar movies to the users.

By adopting the above second approach, we can recommend a list of movies that users will enjoy or want to watch given by a user previously enjoy or have watched before.

C. We find that the movie recommendation system should also considering ratings and popularity besides the similarity scores calculated from metadata like "director", "genres" and "actors". We should compare each movie, if it is a terrible movie that should not be recommended to the users although it is similar to user's liked movies. Therefore, we need to remove bad movies and return the most popular movies as well as the similar movies to the users.

4）movie revenue prediction

Regression analysis is a quantitative research method which is used when the study involves modelling and analysing several variables, where the relationship includes a dependent variable and one or more independent variables. In simple terms, regression analysis is a quantitative method used to test the nature of relationships between a dependent variable and one or more independent variables.

Regression model, basically, specifies the relation of dependent variable (Y),which is our revenue data to a function combination of independent variables (X), which can be budget, popularity, year of production in our case and unknown parameters (β)

$$Y \approx f(X, \beta)$$

Regression equation can be used to predict the values of ‘y’, if the value of ‘x’ is given, and both ‘y’ and ‘x’ are the two sets of measures of a sample size of ‘n’. The formulae for regression equation would be:

$$Revenue^* = a + b(Budget) + c(popularity) + \ldots\ldots$$

I. predicting the movie revenue

1.Split data into train and test model based on the year the movie launched, basically we are using historical revenue data to predict more recent movie revenue data. Specifically, we create a training set that consists of movies released before year, let’s say 2000 and the testing set will contain of movies released after 2000.

2.Plot the average movie revenue by genres.

A basic graph that can clearly show important insight into the data, such as the cost and revenue difference among different genres. We can have a general picture of what the most profitable genres are and how big the differences are.

3. Regression:

After learning difference regression models, namely Standard multiple regression and Stepwise multiple regression, we decided to use two regression models, compare them and choose the one with better performance.

a.Standard multiple regression considers all predictor variables at the same time, which is suitable for our movie dataset. The regression can show the relationship of multiple factors with our revenue and also to what degree do each of the individual predictors contribute to that relationship.

b.Stepwise multiple regression will analyze which predictors are best used to predict the choice of neighborhood which means that the stepwise model evaluates the order of importance of the predictor variables and then selects a relevant subset. This type of regression problem uses "steps" to develop the regression equation.

4. Test coefficient significance
As the model we developed in the last step, we want to check for each independent variables if they are significant at p = 0.05 level because we want to remove redundant variables that have less explain power.

5.Test on our test data set
Based on the new regression model in our step 4, we want to fit our test data into it. In addition, we want to see the performance of our model by showing the SSE (sum of squared errors) ,SST(total sum of squares) and R square.

6.Plot predicted revenue with actual revenue

7. Boxplot the top highest rating movie by genres
A box plot along with histogram would be showing a certain number of top highest ratings movies that people have rated. The data analysis would give a brief overview of what movies are popular among customers. Based on this, we could take the data into decent analysis for recommendation system.

II. recommend the right movies to users

For the recommendation task, we want to give users some movies to suggest that they may like and watch.

In general, our report will have two types of recommendation suggestions for the users.

The first approach is to recommend the movies generally for all the users.

The second approach is to recommend a list of movies they may like or watch based on a movie users enjoy or watched.

For the first approach, the detail methodology is

(1) Produce a reasonable metric that can represent the popularity of a movie for overall users

(2) Use that metric to calculate the particular score for every movie we have in the database
(3) Rank the score for each movie from high to low and provide the top movies in the list and recommend them to all the users

Current the metric score our methodology would like to adopt is :

$$(WR) = \left(\frac{v}{v+m} * R\right) + \left(\frac{m}{v+m} * C\right)$$

In the metric formula,

(a) v means the number of votes for the movie, represented by 'vote_count' in the data table
(b) m means the lowest number of the vote count that needed to have for that movie to be considered
(c) R means the vote score for the movie, represented by 'vote_average' in the data table
(d) C means the average mean vote score for all the movies in the data table
(e) WR means weighted rating for the movie

Certainly, we should not sort movies solely on the average ratings because 8.5 average score with only 2 votes definitely not greatly better than a movie with an 8.0 average score with 50 votes count, therefore, we will use the above metric formula to generate a weighted score for each movie in the list and recommend them to all users.

It is good to have this general list, such as 'trending now' section that shows all the users in the top bar section in a movies website, which kindly can represent all users' movie taste, however, a more specific recommendation engine for each particular person's like would be better.

Code

A. Data Analysis

1. Characteristics of a movie generate higher revenue.

```
import pandas as pd
filepath = r"tmdb_5000_movies.csv"
df = pd.read_csv(filepath)
b = df[['runtime','revenue']].nlargest(10, 'revenue')
c = df[['title','revenue']].nlargest(10, 'revenue')['title']
```

d = df[['vote_average','revenue']].nlargest(10, 'revenue')

2. Average movie ratings for some specific production companies.
df.groupby('production_countries').vote_average.mean().nlargest()

3. Average movie ratings for some specific production countries.
df.groupby('production_companies').vote_average.mean().nlargest()

4. Top 100 most revenue movies
df.sort_values('revenue').head(100)['title']

5. Top 100 most commented movies
df.sort_values('vote_count').head(100)['title']

6. Recommend users with a movies list that they will enjoy.

7. Predict the movie's specific revenue based on some factors (regression)

lm = LinearRegression(normalize=True)
lm.fit(x_train,y_train)
y_pred = lm.predict(x_test)

B. Demographic Filtering - Trending now for All Users

#initial data cleaning
movies_credits.drop('title', axis = 1, inplace = **True**)

# C is the mean vote
C = movies['vote_average'].mean()

# m is the minimum votes
m = movies['vote_count'].quantile(0.8)

def weighted_rating_score(data, m=m, C=C):

v = data['vote_count']

R = data['vote_average']

```
# weighted rating score calculation formula return (v/(v+m) * R) + (m/(m+v) * C)


# calculate weighted rating score for each movie we have
qualified_movies_demographic['weighted_score'] = qualified_movies_demographic.appl
y(weighted_rating_score, axis=1)

# descending sort movies based on the weighted rating score
qualified_movies_demographic = qualified_movies_demographic.sort_values('weighte
d_score', ascending=False)

# print out the top 20 movies under the trending now section
qualified_movies_demographic[['title', 'vote_count', 'vote_average', 'weighted_s
core']].head(20)
```

C. Content Based Filtering - using Movie Overviews

```
#Import TfIdfVectorizer from scikit-learn
from sklearn.feature_extraction.text import TfidfVectorizer
#Define a TF-IDF Vectorizer Object. Remove all english stop words such as 'the', 'a'
tfidf = TfidfVectorizer(stop_words='english')
#Replace NaN with an empty string
movies['overview'] = movies['overview'].fillna('')
#Construct the required TF-IDF matrix by fitting and transforming the data
tfidf_matrix = tfidf.fit_transform(movies['overview']) #Output the shape of tfidf_matrix
tfidf_matrix.shape

# Import linear_kernel
from sklearn.metrics.pairwise import linear_kernel # Compute the cosine similarity
matrix
cosine_sim_overviews = linear_kernel(tfidf_matrix, tfidf_matrix)

#Construct a reverse map of indices and movie titles
indices = pd.Series(movies.index, index=movies['title']).drop_duplicates()

Recommend similar movies to users given a movie title input
def get_recommendation_movies(title, cosine_sim): # get the movie index
idx = indices[title]
    # calculate the pairwise similarity scores with that movie
```

```
    sim_scores = list(enumerate(cosine_sim[idx]))
    # sort the movies based on the similarity scores
sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True) # get the 10 most
similar movies scores
    sim_scores = sim_scores[1:11]
    # get movie indices
movie_indices = [i[0] for i in sim_scores] # return the recommended similar movies
return movies['title'].iloc[movie_indices]
```

D. Content Based Filtering - using Movie Metadata ('genres', 'keywords', 'cast', 'crew', 'production_companies')

```
# convert all strings into lower case and
# strip the spaces between names so as to be specific
def clean_data(data):
if isinstance(data, list):
return [str.lower(i.replace(" ", "")) for i in data] else:
if isinstance(data, str):
return str.lower(data.replace(" ", ""))
else:
return ''

# import CountVectorizer, create count matrix
from sklearn.feature_extraction.text import CountVectorizer

count = CountVectorizer(stop_words='english')
count_matrix = count.fit_transform(movies['data_soup'])

# compute the Cosine Similarity matrix
from sklearn.metrics.pairwise import cosine_similarity

cosine_sim_metadata = cosine_similarity(count_matrix, count_matrix)
```

E Content Based Filtering - using Movie Metadata ('genres', 'keywords', 'cast', 'crew', 'production_companies') and voting scores

```
Def improved_recommendations_metadata_votes(title, cosine_sim):
idx = indices[title]
sim_scores = list(enumerate(cosine_sim[idx]))
```

```
sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True) sim_scores =
sim_scores[1:36]
movie_indices = [i[0] for i in sim_scores]
   movies_improved_recomm = movies.copy().iloc[movie_indices][['title', 'vote_cou
nt', 'vote_average']]
   vote_counts_improved =
movies_improved_recomm[movies_improved_recomm['vote_cou
nt'].notnull()]['vote_count'].astype('int')
   vote_averages_improved =
movies_improved_recomm[movies_improved_recomm['vote_average'].notnull()]['vote_a
verage'].astype('int')
   C_improved = vote_counts_improved.mean()
   m_improved = vote_averages_improved.quantile(0.60)
   qualified_improve =
movies_improved_recomm[(movies_improved_recomm['vote_count
'] >= m_improved) & (movies_improved_recomm['vote_count'].notnull()) & (movies_imp
roved_recomm['vote_average'].notnull())]
   qualified_improve['wr'] = qualified_improve.apply(weighted_rating_score, axis=
1)
qualified_improve = qualified_improve.sort_values('wr', ascending=False).head( 10)
return qualified_improve
```

For more details, please check the python notebook.
“#” means basic explanation.

Results

In a fictitious study we were studying data collected from the content of the TMDB dataset that contains around 5000 movies and TV series. We can state the trends shown in the data, like in all movies the most famous movie genres produce the highest revenues. For example, Avatar, Star Wars and Titanic movies are top 3 of the revenues.

- **Movie Recommendation**

From the code recommendation we can see that use weighted rating to generate recommendations based on the user demographic attributes.

```
# calculate weighted rating score for each movie we have
qualified_movies_demographic['weighted_score'] = qualified_movies_demographic.appl
```

y(weighted_rating_score, axis=1)

1. Movie recommendation results from Demographic Based method

Through this method, in our results, the movies we recommend are all have a large amount of review counts and high review scores. We use a weighted rating score formula to calculate each movies' score.

In the results, some of the top movies we recommend are "The Shawshank Redemption", "Fight Club", "The Godfather", "Pulp Fiction", "The Dark Knight" and "Star Wars", which are all movies that famous for all people around the world.

2. Recommendation results from Content Based method using Movie Overviews.

Through this method, we recommend movies based on the keywords in the movies' overview, such as "Batman" and "Gotham City" in the movie "The Dark Knight Rises" 's overview, we examine the similarity score between different movies. What's more, in our method, we will remove unnecessary words like "the", "and".

In the results, for example, if user has seen a movie like "The Dark Knight Rises", then our results will recommend movies to users like "The Dark Knight", "Batman Returns", "Batman", "Batman Begins" etc.

Another example result list of movies we would recommend for users who have seen a movie like "The Avengers" are "Avengers: Age of Ultron", "Plastic", "This Thing of Ours", "Wall Street: Money Never Sleeps" and "Team America: World Police" etc.

3. Recommendation results from Content Based method using Movie Metadata.

In this method, we use each movie's metadata like "director", "genres", "actors, actress", "production companies" to find similarity between each movie and make the recommendation.

In our results, for example, if a user wants to be recommended with a list of movies after he/she watches "The Avengers", then "Avengers: Age of Ultron", "Captain America: Civil War", "Iron Man 2", "Captain America: The Winter Soldier" will be shown to users.

4. Recommendation results from Content Based method using Movie Metadata and vote scores

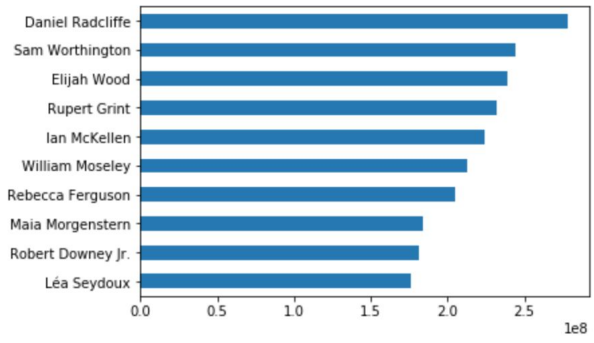
In this method, we combined the movie's meta information like "director", "genres", "actors, actress", "production companies" together with "vote scores" since we certainly don't want to recommend users with similar metadata but with a bad vote reviews movie.

In our example results, when user has reviewed the "The Dark Knight Rises" movie, our recommendation system results will show "The Dark Knight Rises", "The Dark Knight", "Interstellar", "Batman Begins" and "Man of Steel " to users.

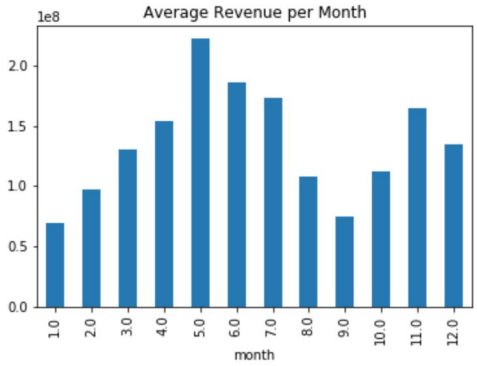
Another example recommend list for movie "The Avengers" are "Guardians of the Galaxy", "Captain America: The Winter Soldier", "Star Trek", "Ant-Man".

- **Revenue prediction**

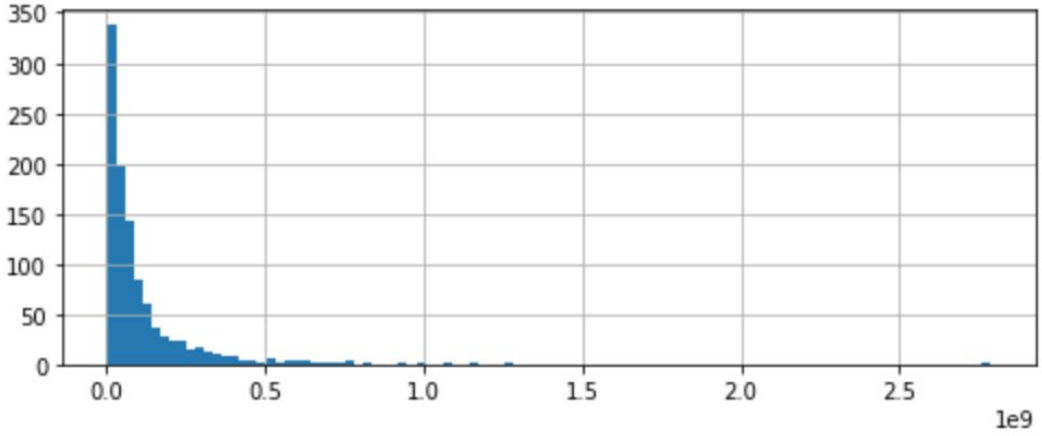
Data analysis results
The highest revenues for actors



Average revenue per month



Revenues of directors



According to our previous analysis, drama is the most common kind of movies, so we picked two best regression models, namely for all the movies and for all drama movies. The two models are shown as below:

1. Revenue = 140682 * budget + 2289*release date + 775034*run time + 268278 *vote count – 1562024 * vote average
2. Revenue = 166008 * budget - 10344*release date + 470666*run time + 288108 *vote count – 1346243 * vote average

Discussion

- **Movie Recommendation System**

1. Recommendation results from Demographic Based method

Our movie recommendation results in this method shows the general popular list of movies that a very large amount of users would like to watch, probably above 95% of users.

For example, in our results, we recommend “The Shawshank Redemption”, “Fight Club”, “The Godfather”, “Pulp Fiction”, “The Dark Knight”, “Inception” and “Interstellar”. These movies are all highly rated with a large number of fans audience.

However, the problem is that these recommendations are very general to all the users. It doesn't take into consideration of each user’s specific taste, therefore, the result is not concrete and we have the below other methods.

2. Recommendation results from Content Based method using Movie Overviews.

In this method, we recommend movies based on the keywords in the movies’ overview, such as “Batman” and “Gotham City” in the movie “The Dark Knight Rises”, we examine the similarity score between different movies.

The recommendation results for the movie “The Dark Knight Rises” is decent, the reason is because some specific keywords like “Batman” and “Gotham City” are only shown in the Batman series of movies, so that the recommended engine can accurately determine the correct similar results to users.

However, when specific terms are not shown in the overview, like “The Avengers”, then, the recommended list in our result is not accurate, for example “The Fountain” is shown in our result list however that movie is not very related with “The Avengers” movie.

For further analysis, the recommendation system only based on overviews is not a well-defined solution to recommend to the correct users. While sometimes, overview could be considered as a factor to be used, however, other metadata like movie’s “director”, “genres” can probably better defined and represent the movie as a single element.

3. Recommendation results from Content Based method using Movie Metadata.

Through this method, each movie’s metadata like “director”, “genres”, “actors, actress”, “production companies”, “keywords” are used to find the similarity between each movie and make the recommendation.

In our recommendation results for the “The Dark Knight Rises” movie, all of our recommendations are reasonable like “The Dark Knight”, “Batman Begins”, “Man of Steel”, “The Prestige” and “Superman Returns”. The recommendation list for the movie “The Avengers” are reasonable as well such as “Avengers: Age of Ultron”, “Captain America: Civil War”, “Iron Man 2”, “Captain America: The First Avenger”, “Captain America: The Winter Soldier” and “The Incredible Hulk”.

The problem is that sometimes, although these 2 movies are similar in the metadata, however, maybe one of the movies has a very bad rating because of the story line or other reasons. Therefore, besides using the metadata information, we can also consider movies’ overall ratings and build them together.

4. Recommendation results from Content Based method using Movie Metadata and vote scores

Through this method, the movie’s meta information like “director”, “genres”, “actors, actress”, “production companies” together with “vote scores” are considered together since we certainly don’t want to recommend users with similar metadata but has bad reviews movie.

The results show that this method works pretty well, all movies recommended for “The Dark Knight Rises” are very similar to that comic style but also has high vote scores like: “Interstellar”, “Batman Begins”. For the movie “The Avengers”, through this method, our result does not show “Captain America: The First Avenger” to users compared with the above method, although the movie is very similar to “The Avengers”, it does not have a high enough vote score for our users.

Actually more factor could put in consideration for the data mining process, which could be quite helpful for improving accuracy.

5. Revenue prediction

Interpretation of first model:
For the first model, we have a r2 score of 0.35, which means over 30% of our predicted data can be correctly estimated by our model. Because of the large amount of our data, we have a squared error of 2.55. According to our model, we can say budget is positively and greatly affecting our revenue. In addition, films tend to earn 2289 dollars

more every day, which corresponds to the fact that most recent movies earn more than older movies.

Interpretation of second model:
For the second model, we have a r2 score of 0.30, which means over 30% of our predicted data can be correctly estimated by our model. Because of the large amount of our data, we have a squared error of 1.09. According to our model, we can say budget is positively and greatly affecting our revenue. In addition, unlike our first model films tend to earn 10344 dollars less every day.

Comparison and key observations:

1.Release date has more explanatory power than year in our prediction: There is another data field in our original dataset called year. If we replace our release date with year in our model, we have a lower R2 score and a higher standard error. After studying our dataset, we believe the reason could be that the year field is more in vogue than actual release date. For famous movie like Avatar, the data seem to be collected from year 2010 which is not the original release date. In this way, the revenue is shown less during the year 2010 because we all know Avatar have over $2 billion revenue globally.

2.Surprisingly, drama films are earning less in recent year (>2010). As we all may believe that movies released today will earn more than old movies because the world economy is overall growing and our movie industry is prospering, the regression on drama movies shows the opposite. For drama movie, we have a negative coefficient for release date which shows people are less interested in this genre. If we want to attract more audience, we shouldn't be sitting on our old idea that drama is the most popular genre and explore other kinds of movies.

In the fictitious study we used to illustrate the results section, it makes sense that most famous movie genres produce the highest revenues. Everyone knows the popular movie and enjoys to watch the movie. Huge ticket sales are usually matched by huge revenues.

Future Work

There are many ways to expand the work on this project, which there couple tips we'd like to point out with this project.

First, we should try to combine the advantage of demographic-based filtering and content-based filtering techniques into the same movie recommendation system.

Secondly, in our second content-based approach, more weights can be added towards the director perspective since people normally would like to endorse a particular director's film series.

Third, movies recommendation using metadata can be improved, since we only focus on the metadata similarity, however, some movies with bad ratings should not be recommended although they are similar to the user's likeness. More future work can be added here to recommend the right movies to the users.

Furthermore, as the number of users on the website increases, we could use collaborative filtering techniques to build a model from the user's past behavior as well as similar attributes from other users. This will use to predict movies to the users who may have an interest in that kind of movie.

At last, one thing we took in concern is that all the data information all provided with users' feedback, and later on, we would like to learn more data and resources from actors, directors and investors to improve the completion of model, so that the accuracy of recommendation and prediction would be better.

Conclusion

We create movie recommendation system using demographic-based filtering and content-based filtering techniques. While general data mining technique is very fundamental and cannot be used to predict the trends. However, demographic-based filtering and content-based filtering techniques are proved to be complementary.

Recommendation to users can have a great impact for the movies industry.

References

1. S Bhatnagar, 2014, Chapter 5, Data Mining Applications with R

2. K Meenakshi , G Maragatham , Neha Agarwal and Ishitha Ghosh,2018, A Data mining Technique for Analyzing and Predicting the success of Movie

3.https://research-methodology.net/research-methods/quantitative-research/regression-analysis/

https://www.kaggle.com/tmdb/tmdb-movie-metadata/kernels