

# Movie Recommender System

## Capstone Project One Final Report

12-22-2018

---

Samin Rastgoufard

Springboard

samin.rastgoufard@gmail.com

## Overview

Nowadays, we need recommender systems almost everywhere in our lives. Therefore, retailers are becoming more interested in recommender systems to analyze patterns of user interest in products and provide personalized recommendations. The first goal of this project is understanding, analyzing, and correlating the trend in average rating movies of different genres. The second goal is building recommender engines to provide recommendations to different users and build different machine learning models to predict the rating of each movie.

## Business Objective

The recommender system is useful to any business that makes money via recommendations. Since we want to work with movie dataset, the client of this project could be Amazon, Netflix, Hulu, HBO, etc.

Giving good recommendations will help users spend less time searching for their type of movie and having a recommender platform. This will help the customer to continue with the service and having a good experience.

## Data

I will use two datasets for this project as:

1. The first dataset is collected from a free, noncommercial movie recommender as MovieLens.org. (<https://grouplens.org/datasets/movielens>). This recommender is like Netflix, minus the ability to watch movies. The dataset which was released in April 2015 contains 20 million ratings and 465,000 tag applications applied to 27,000 movies by 138,000 users (movies.csv, ratings.csv, link.csv). The data has been created in October 2016. Users were selected at random for inclusion. All selected users had rated at least 20 movies.
2. The second dataset is collected from (<https://www.kaggle.com/karrimba/movie-metadatacsv>). We have more features in this dataset like information about revenue, budget, release date, popularity,...

## Solution

### I. How to get the insight about how people's rating change for different genres over the years?

Getting different distributions of the movie-ratings and movie-relevancy across title, genre, year, and tag of each movie can give the insight about how people's taste vary over the years. Different inferential statistics and visualization techniques can be implemented to deliver a better understanding of the trends.

### II. How to recommend movies?

In this work different Collaborative filtering (CF) and Content-based filtering techniques will be used to analyze relationships between users and interdependencies among products to identify new user-item associations. Most CF algorithms are based on a user-item rating matrix where each row represents a user, each column an item. The entries of this matrix are ratings given by users to items. One of the primary areas of collaborative filtering is the latent factor models which are based on matrix factorization (Singular value decomposition (SVD)).

### III. How to predict movie ratings?

In this work different machine learning techniques like Random Forest, Support Vector Machine, K-Nearest Neighbors,... will be implemented to predict the rating of the test set.

## Workflow

- Collecting data and applying data wrangling methods
- Starting exploratory data analysis to find trends and storytelling
- Conduct further data analysis to identify relationships between different variables
- Perform in-depth analysis using collaborative filtering and machine learning techniques to recommend and predict

## First Dataset (MovieLens)

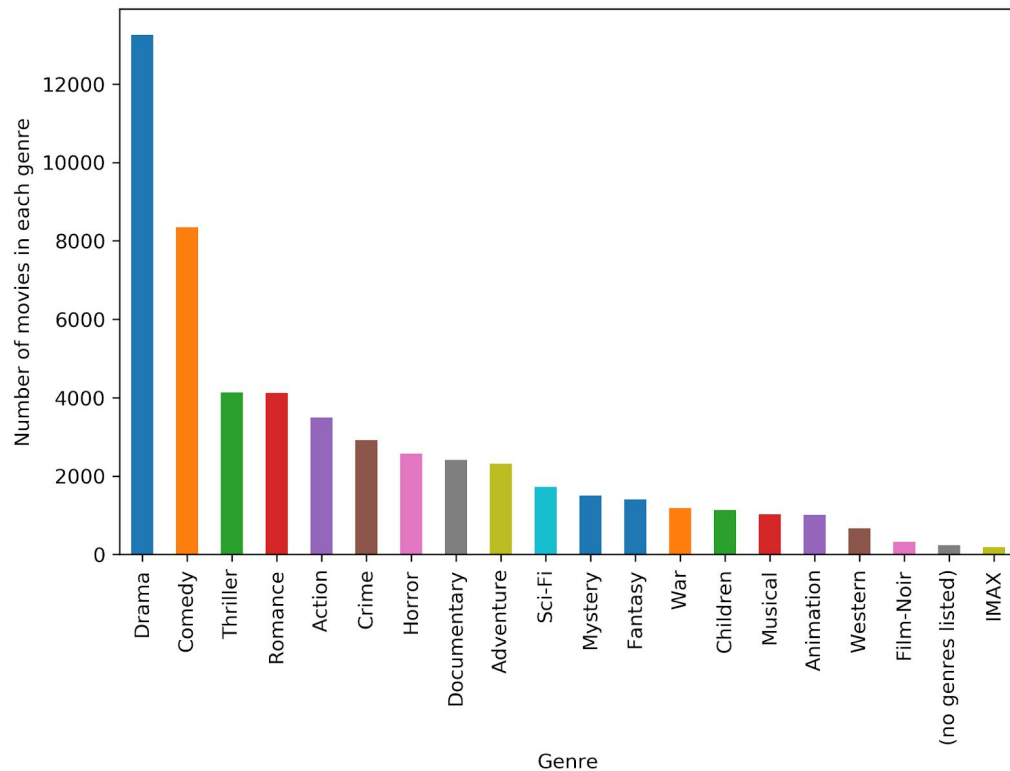
### Data Wrangling

This dataset was clean. The 'rating.csv' has 20,000,263 rows and 4 columns. The 'movie.csv' dataset has 27,278 rows and 3 columns. I just removed the year from the titles and made 'year' and 'decade' as separate columns. I also prepared the data for EDA by separating different genres and merge it with the rating dataset for better EDA.

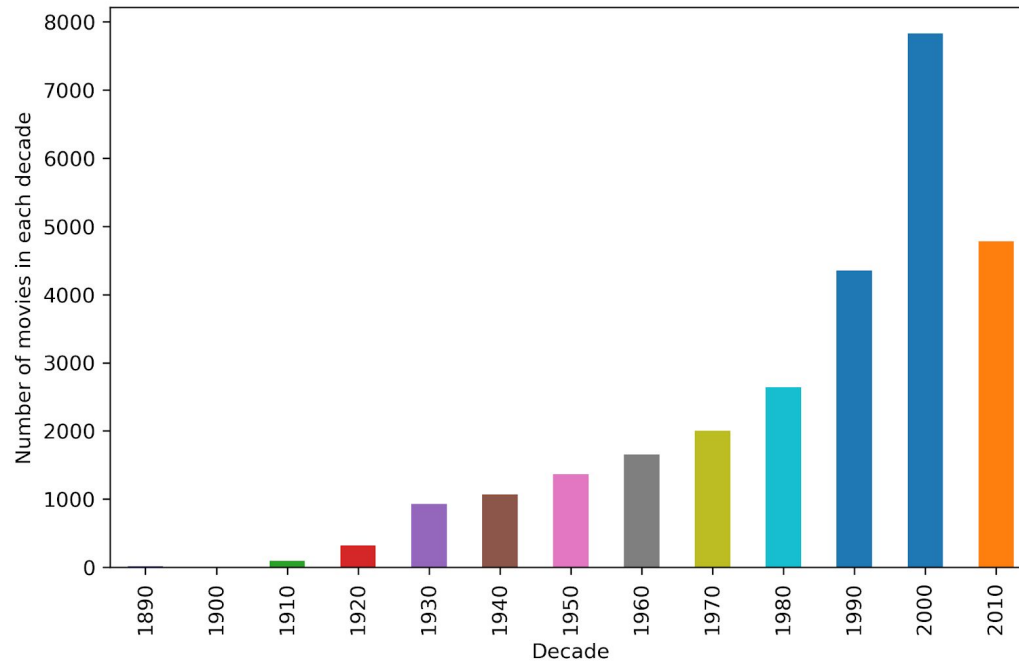
### Exploratory Data Analysis (EDA) and Storytelling

In this section, the various insights produced through descriptive statistics and data visualization is presented.

Genres: **Drama**, **Comedy**, and **Thriller** have the most production.

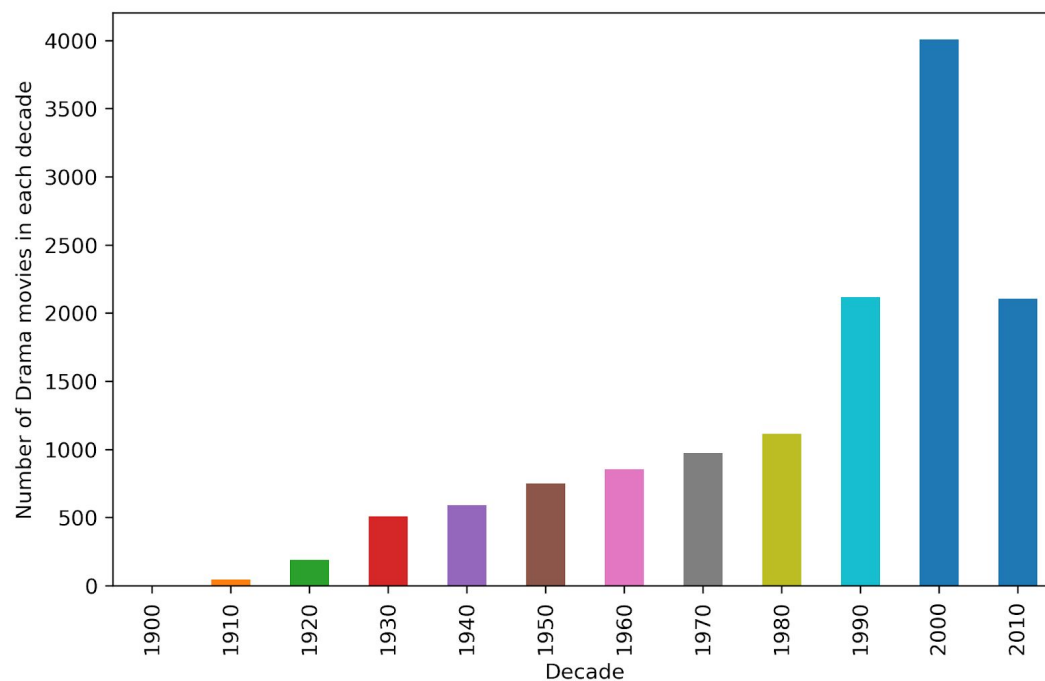


Total number of movies in each decade is presented below. Most production was in the decade **2000**.

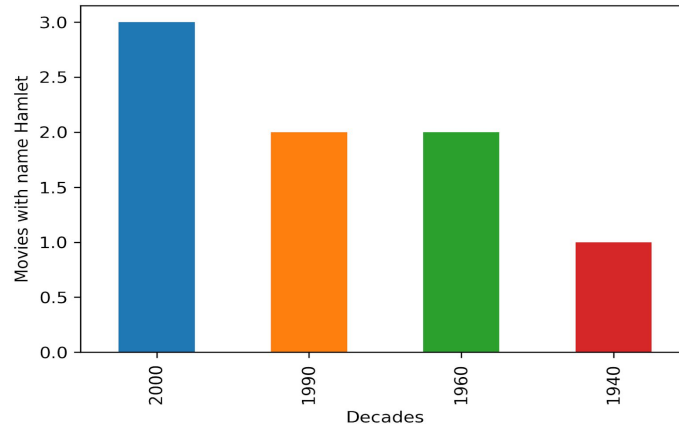


Most **drama** movies has been produced in years **2009**, **2007**, and **2008** respectively.

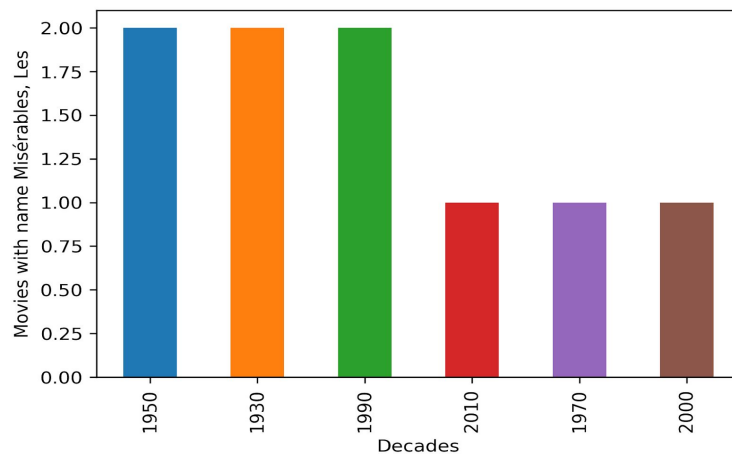
Most **drama** movies produced in the decade **2000**.



8 movies have been produced with the title '**Hamlet**', The following barplot shows the number of 'Hamlet' movies in each decade.



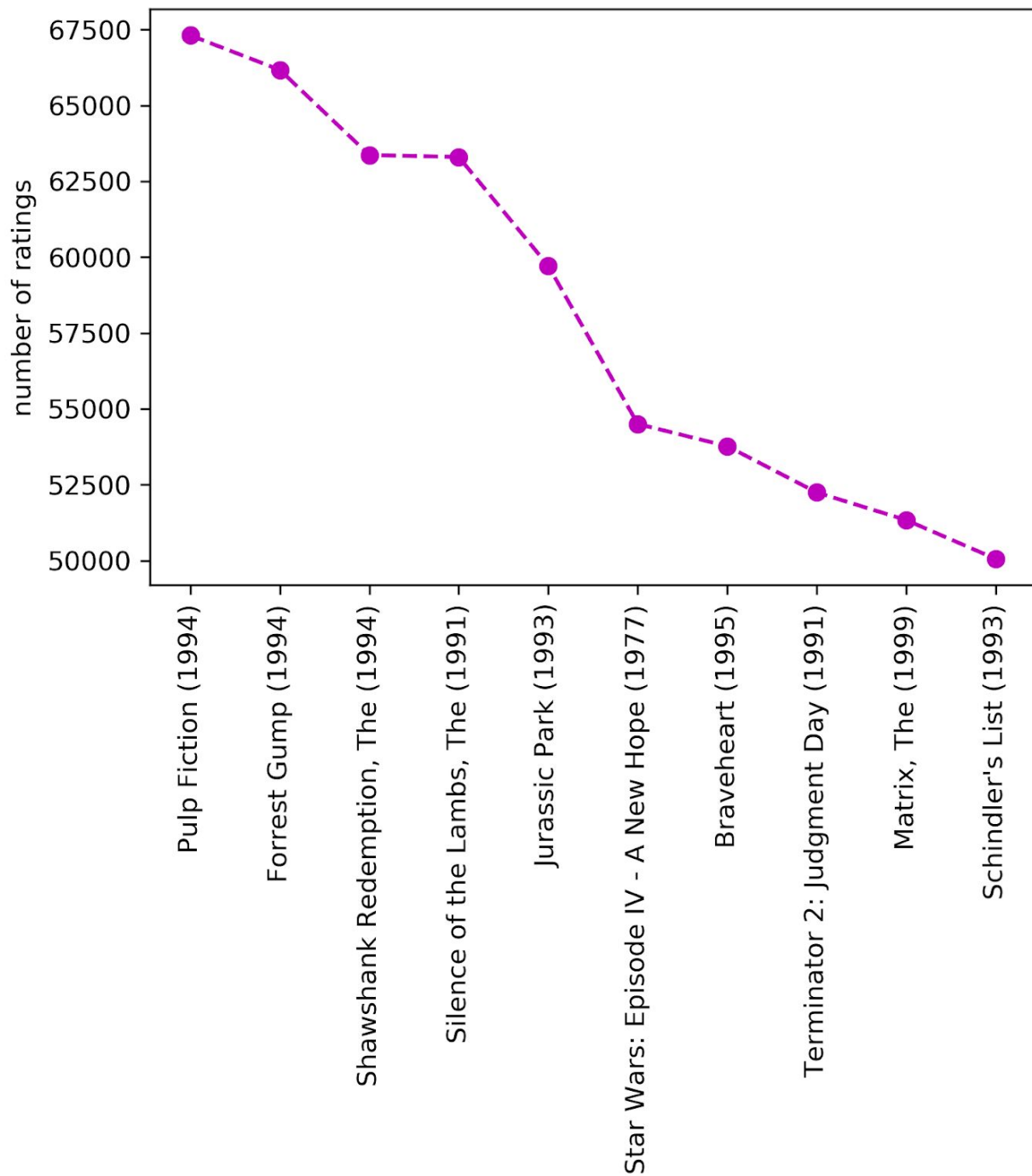
9 movies have been produced with title '**Misérables, Les**'. The following barplot shows the number of 'Misérables, Les' movies in each decade.



Most favorable movies are:

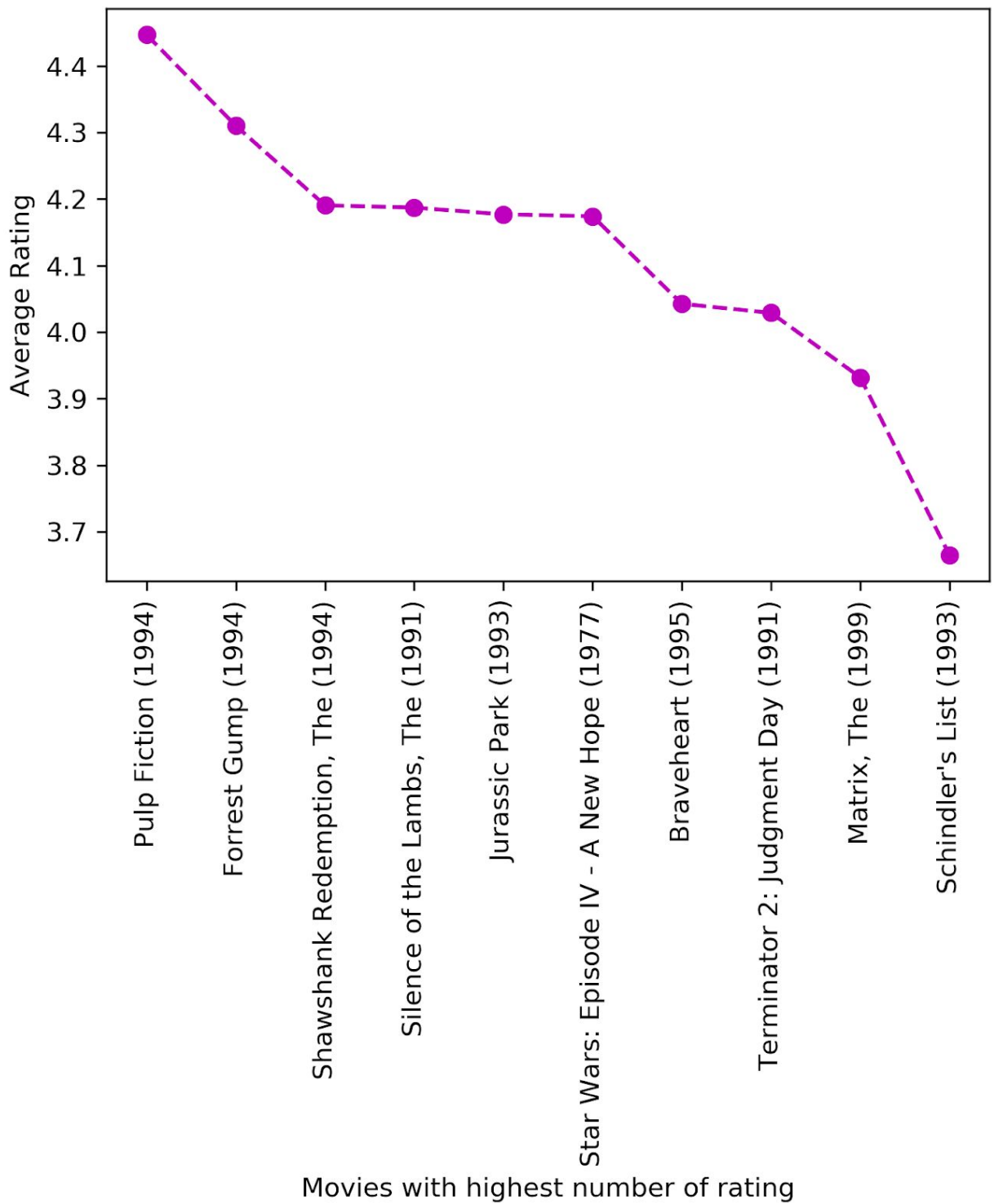
- As it was expected, **Shawshank Redemption**, **Pulp Fiction**, and **Silence of the Lambs** are the first three with most ranking equal to 5 respectively.
- **Shawshank Redemption**, **Matrix**, and **Pulp Fiction** are the first three with most ranking equal to 4.5 respectively.
- The first three movies with most ranks equal to 4 are **Silence of the Lambs**, **Fugitive**, and **Jurassic**.
- The average rating for the most favorable movie **Shawshank Redemption** is: 4.447
- The average rating for the second favorable movie **Pulp Fiction** is: 4.174
- The movies with the max number of ratings are **Pulp Fiction**, **Forrest Gump**, **Shawshank redemption** respectively.

- The graph for the ten movies with the maximum number of the rating is presented below.



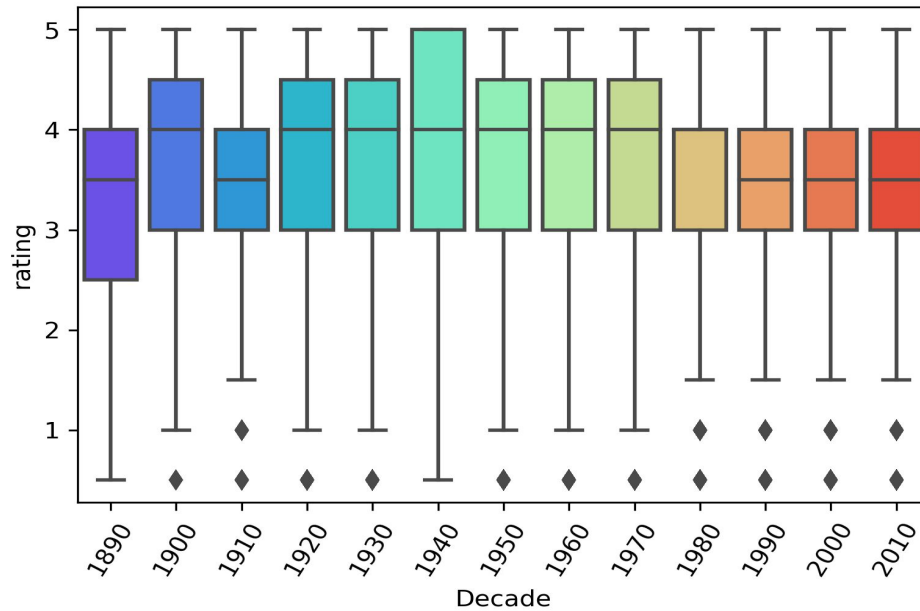
10 movies with the max number of rating

- The graph of the movies with the largest average rating is presented below.

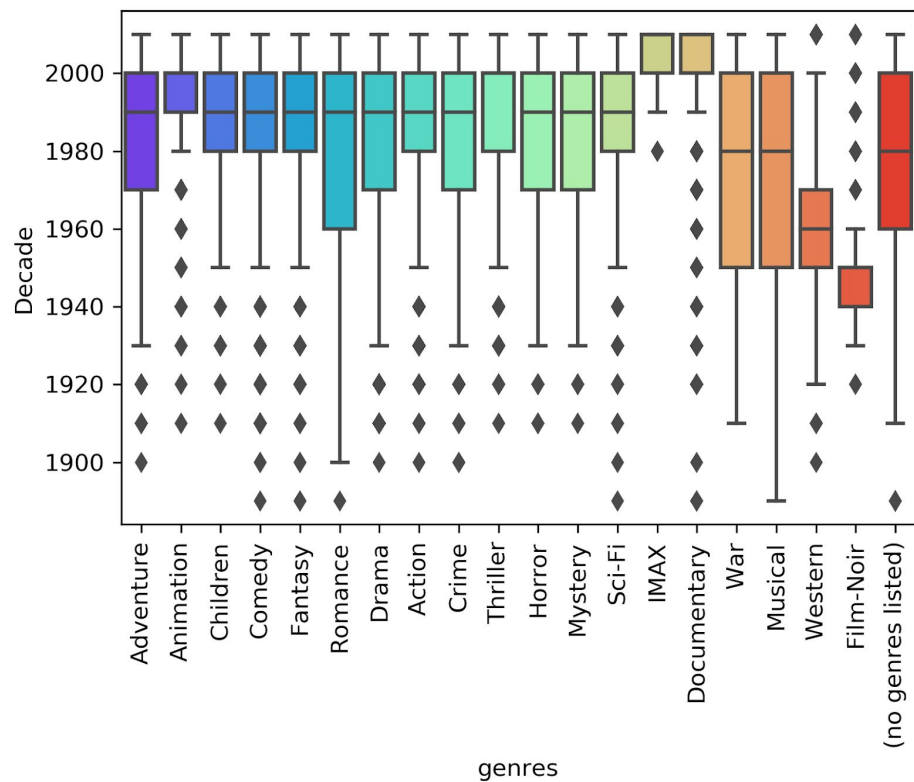




- The Boxplot of the '**Decade**' vs '**Rating**' is illustrated below. The movies of the **40th** decade have the highest rating.



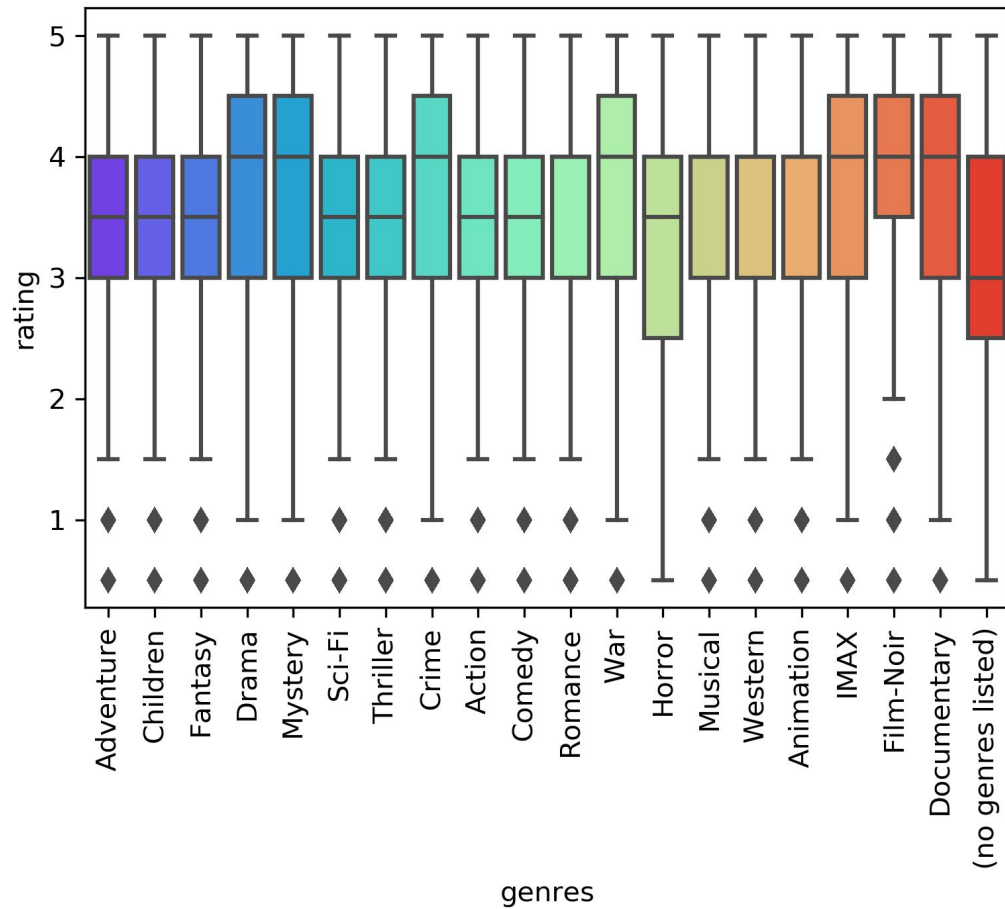
- Boxplot of the '**Decade**' vs '**Genres**' is presented below:



- Boxplot of the 'Genre' vs 'Rating'

**Drama, Mystery, Crime, War, Imax, documentary** have the highest ratings.

**Horror** is the least favorite.



## Title Word Clouds

There are certain words that use more often in titles. I use the WordCloud library to find out what are these words. The word **Love** is the most commonly used word in movie titles. **Girl**, **Day** and **Man** are also among the most commonly occurring words.



## Recommender Systems

Before applying different recommender engines, I filtered the movies with more than 100 vote counts. Because we don't want movies with small number of votes, affect our recommender. Also, I use the sklearn library to set the train and test data.

## 1. Simple Recommender

The Simple Recommender offers generalized recommendations to every user based on movie popularity, ratings, and genre. The basic idea behind this recommender is that movies that are more popular and more critically acclaimed will have a higher probability of being liked by the average audience. This model does not give personalized recommendations based on the user.

So I sort the movies based on their genre, and I recommend the ones with the highest number of votes and the higher average rankings to the new audience.



The list of the **10 best Drama** movies to recommend are:

- 1 : Pulp Fiction (1994)
- 2 : Shawshank Redemption, The (1994)
- 3 : Schindler's List (1993)
- 4 : American Beauty (1999)
- 5 : Fargo (1996)
- 6 : Godfather, The (1972)
- 7 : Fight Club (1999)
- 8 : Lord of the Rings: The Return of the King, The (2003)
- 9 : One Flew Over the Cuckoo's Nest (1975)
- 10 : Godfather: Part II, The (1974)

The list of the **10 best Romance** movies to recommend are:

- 1 : Forrest Gump (1994)
- 2 : Beauty and the Beast (1991)
- 3 : Princess Bride, The (1987)
- 4 : Groundhog Day (1993)
- 5 : Shrek (2001)
- 6 : Sleepless in Seattle (1993)
- 7 : Good Will Hunting (1997)
- 8 : Four Weddings and a Funeral (1994)
- 9 : Crouching Tiger, Hidden Dragon (Wo hu cang long) (2000)
- 10 : There's Something About Mary (1998)

The list of the **10 best Action** movies to recommend are:

- 1 : Star Wars: Episode IV - A New Hope (1977)
- 2 : Braveheart (1995)
- 3 : Matrix, The (1999)
- 4 : Star Wars: Episode VI - Return of the Jedi (1983)
- 5 : Star Wars: Episode V - The Empire Strikes Back (1980)
- 6 : Raiders of the Lost Ark (Indiana Jones and the Raiders of the Lost Ark) (1981)
- 7 : Fight Club (1999)
- 8 : Saving Private Ryan (1998)
- 9 : Princess Bride, The (1987)
- 10 : Lord of the Rings: The Return of the King, The (2003)

## 2. IMDB Weighted Rating Formula

Another technique could be IMDB's **weighted rating** formula which is mathematically represented as follows:

$$WR = \frac{v}{v+m}R + \frac{m}{v+m}C$$

where,

$v$  is the number of votes for the movie

$m$  is the minimum votes required to be listed in the chart

$R$  is the average rating of the movie

$C$  is the mean vote across the whole report

The next step is to determine an appropriate value for  $m$ , the minimum votes required to be listed in the chart. We will use **95th percentile** as our cutoff. In other words, for a movie to feature in the charts, it must have more votes than at least 95% of the movies in the list.

- mean vote across the whole report (C): **3.13**
- minimum votes required to be listed in the chart (m): **8463.75**
- **1061** movies are qualified

The list of the 30 best movies to recommend are:

|    |   |    |   |
|----|---|----|---|
| 0  | Shawshank Redemption, The (1994)                  | 15 | City of God (Cidade de Deus) (2002)               |
| 1  | Usual Suspects, The (1995)                        | 16 | Monty Python and the Holy Grail (1975)            |
| 2  | Godfather, The (1972)                             | 17 | Casablanca (1942)                                 |
| 3  | Schindler's List (1993)                           | 18 | Lord of the Rings: The Return of the King, The... |
| 4  | Fight Club (1999)                                 | 19 | Dr. Strangelove or: How I Learned to Stop Worr... |
| 5  | Pulp Fiction (1994)                               | 20 | Inception (2010)                                  |
| 6  | Star Wars: Episode IV - A New Hope (1977)         | 21 | Chinatown (1974)                                  |
| 7  | Silence of the Lambs, The (1991)                  | 22 | American Beauty (1999)                            |
| 8  | Matrix, The (1999)                                | 23 | Life Is Beautiful (La Vita è bella) (1997)        |
| 9  | Star Wars: Episode V - The Empire Strikes Back... | 24 | Fargo (1996)                                      |
| 10 | North by Northwest (1959)                         | 25 | Wallace & Gromit: The Wrong Trousers (1993)       |
| 11 | Princess Bride, The (1987)                        | 26 | Seven Samurai (Shichinin no samurai) (1954)       |
| 12 | Godfather: Part II, The (1974)                    | 27 | Memento (2000)                                    |
| 13 | Raiders of the Lost Ark (Indiana Jones and the... | 28 | Rear Window (1954)                                |
| 14 | Dark Knight, The (2008)                           | 29 | Blade Runner (1982)                               |

### 3. Content-Based Filtering

The recommender we built in the previous section suffers some severe limitations. For one, it gives the same recommendation to everyone, regardless of the user's personal taste. If a person who loves romantic movies (and hates action) had to look at our Top 10 Chart, s/he wouldn't probably like most of the movies. If s/he were to go one step further and look at our charts by genre, s/he wouldn't still be getting the best recommendations.

So in this section, I set an engine to recommend based on the user's rating history. A simple example is using the mean as an aggregation function. For implementation step check the lpython notebook.

The RMSE for content-based filtering is 0.9969 which is a acceptable value for this large dataset.

The total elapsed time for this evaluation was 61.474 sec.

### 4. Collaborative Filtering

Collaborative Filtering is based on the idea that users similar to a me can be used to predict how much I will like a particular product or service those users have used/experienced but I have not.

### 1- Simple Collaborative filtering based on mean

Recommend based on other user's rating histories. A simple example is using the mean as an aggregation function.

The results for some random user and movies Id is presented below:

```
Collaborative mean filter for userId: 1 and movieId: 151 is: [3.31060606]  
Actual rating value is: 4.0
```

```
Collaborative mean filter for userId: 1 and movieId: 29 is: [3.80232558]  
Actual rating value is: 3.5
```

```
Collaborative mean filter for userId: 138493 and movieId: 55269 is:  
[4.06666667]  
Actual rating value is: 5.0
```

### 2-Matrix Factorization-based algorithms

The idea is basically to take a large (or potentially huge) matrix and factor it into some smaller representation of the original matrix. You can think of it in the same way as we would take a large number and factor it into two much smaller primes. We end up with two or more lower dimensional matrices whose product equals the original one.

When we talk about collaborative filtering for recommender systems we want to solve the problem of our original matrix having millions of different dimensions, but our “tastes” not being nearly as complex. Even if i’ve viewed hundreds of items they might just express a couple of different tastes. Here we can actually use matrix factorization to mathematically reduce the dimensionality of our original “all users by all items” matrix into something much smaller that represents “all items by some taste dimensions” and “all users by some taste dimensions”. These dimensions are called latent or hidden features and we learn them from our data.

Doing this reduction and working with fewer dimensions makes it both much more computationally efficient and but also gives us better results since we can reason about items in this more compact “taste space”.

If we can express each user as a vector of their taste values, and at the same time express each item as a vector of what tastes they represent. You can see we can quite easily make a

recommendation. This also gives us the ability to find connections between users who have no specific items in common but share common tastes.

**SVD** is a matrix factorization technique that is usually used to reduce the number of features of a data set by reducing space dimensions from  $N$  to  $K$  where  $K < N$ . For the purpose of the recommendation systems however, we are only interested in the matrix factorization part keeping same dimensionality. The matrix factorization is done on the user-item ratings matrix. From a high level, matrix factorization can be thought of as finding 2 matrices whose product is the original matrix.

([https://medium.com/@m\\_n\\_malaeb/singular-value-decomposition-svd-in-recommender-systems-for-non-math-statistics-programming-4a622de653e9](https://medium.com/@m_n_malaeb/singular-value-decomposition-svd-in-recommender-systems-for-non-math-statistics-programming-4a622de653e9))

I use **Surprise** library which has several powerful algorithms like **Singular Value Decomposition (SVD)**, **Non-negative Matrix Factorization (NMF)**, **K Nearest Neighbor (KNN)**, and **Co-Clustering** to minimize RMSE (Root Mean Square Error) and give recommendations.

Evaluating RMSE, MAE of algorithm **SVD** on 5 split(s).

|                | Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5        | Mean   | Std    |
|----------------|--------|--------|--------|--------|---------------|--------|--------|
| RMSE (testset) | 0.9165 | 0.9119 | 0.9194 | 0.9091 | <b>0.9220</b> | 0.9158 | 0.0047 |
| MAE (testset)  | 0.7058 | 0.7010 | 0.7093 | 0.7029 | 0.7101        | 0.7058 | 0.0035 |
| Fit time       | 16.00  | 15.48  | 14.18  | 17.57  | 15.86         | 15.82  | 1.09   |
| Test time      | 0.25   | 0.25   | 0.25   | 0.23   | 0.36          | 0.27   | 0.05   |

Total Elapsed time with the model is: 81.90

Evaluating RMSE, MAE of algorithm **NMF** on 5 split(s).

|                | Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5        | Mean   | Std    |
|----------------|--------|--------|--------|--------|---------------|--------|--------|
| RMSE (testset) | 0.9560 | 0.9636 | 0.9502 | 0.9585 | <b>0.9592</b> | 0.9575 | 0.0044 |
| MAE (testset)  | 0.7364 | 0.7387 | 0.7284 | 0.7336 | 0.7390        | 0.7352 | 0.0039 |
| Fit time       | 15.98  | 15.27  | 15.72  | 12.91  | 13.61         | 14.70  | 1.22   |
| Test time      | 0.27   | 0.22   | 0.34   | 0.20   | 0.31          | 0.27   | 0.05   |

Total Elapsed time with the model is: 76.32 sec

Evaluating RMSE, MAE of algorithm **SlopeOne** on 5 split(s).



|                | Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5        | Mean   | Std    |
|----------------|--------|--------|--------|--------|---------------|--------|--------|
| RMSE (testset) | 0.9487 | 0.9463 | 0.9371 | 0.9449 | <b>0.9491</b> | 0.9452 | 0.0043 |
| MAE (testset)  | 0.7264 | 0.7261 | 0.7208 | 0.7232 | 0.7278        | 0.7248 | 0.0025 |
| Fit time       | 9.84   | 10.11  | 11.68  | 10.86  | 11.21         | 10.74  | 0.68   |
| Test time      | 11.49  | 13.06  | 12.45  | 10.91  | 8.92          | 11.37  | 1.43   |

Total Elapsed time with the model is: 112.29 sec

Evaluating RMSE, MAE of algorithm **KNNBaseline** on 5 split(s).

|                | Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5        | Mean   | Std    |
|----------------|--------|--------|--------|--------|---------------|--------|--------|
| RMSE (testset) | 0.9148 | 0.9180 | 0.9152 | 0.9191 | <b>0.9149</b> | 0.9164 | 0.0018 |
| MAE (testset)  | 0.7050 | 0.7066 | 0.7044 | 0.7064 | 0.7034        | 0.7052 | 0.0012 |
| Fit time       | 1.50   | 1.56   | 1.59   | 1.61   | 1.58          | 1.57   | 0.04   |
| Test time      | 5.85   | 5.47   | 5.25   | 6.03   | 5.73          | 5.67   | 0.28   |

Total Elapsed time with the model is: 37.747 sec

Evaluating RMSE, MAE of algorithm CoClustering on 5 split(s).

|                | Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5        | Mean   | Std    |
|----------------|--------|--------|--------|--------|---------------|--------|--------|
| RMSE (testset) | 0.9785 | 0.9831 | 0.9782 | 0.9906 | <b>0.9765</b> | 0.9814 | 0.0051 |
| MAE (testset)  | 0.7587 | 0.7585 | 0.7542 | 0.7660 | 0.7567        | 0.7588 | 0.0039 |
| Fit time       | 12.22  | 6.37   | 6.58   | 6.59   | 6.59          | 7.67   | 2.28   |
| Test time      | 0.44   | 0.17   | 0.47   | 0.17   | 0.47          | 0.34   | 0.14   |

Total Elapsed time with the model is: 41.57

From the results, **SVD** and **KNN** provide the lowest RMSE for the test set.

Now let's compare the predicted rating values of some samples with their actual values:

| UserId<br>MovieId       | SVD  | NMF | Slope One | KNN  | CoClustering | Actual |
|-------------------------|------|-----|-----------|------|--------------|--------|
| userId=1<br>movieId=151 | 3.68 | 3.4 | 3.49      | 3.77 | 3.28         | 4      |

|                                |      |      |      |      |      |     |
|--------------------------------|------|------|------|------|------|-----|
| userId=1<br>movieId=29         | 3.87 | 3.86 | 3.72 | 3.94 | 3.97 | 3.5 |
| userId=138493<br>movieId=55269 | 3.84 | 3.48 | 3.48 | 3.82 | 3.48 | 5   |

## 5. ALS Implicit Collaborative Filtering

Alternating Least Squares (ALS) is the model we'll use to fit our data and find similarities.

ALS is an iterative optimization process where we for every iteration try to arrive closer and closer to a factorized representation of our original data.

We have our original matrix  $R$  of size  $u \times i$  with our users, items and some type of feedback data. We then want to find a way to turn that into one matrix with users and hidden features of size  $u \times f$  and one with items and hidden features of size  $f \times i$ . In  $U$  and  $V$  we have weights for how each user/item relates to each feature. What we do is we calculate  $U$  and  $V$  so that their product approximates  $R$  as closely as possible:  $R \approx U \times V$ .

By randomly assigning the values in  $U$  and  $V$  and using least squares iteratively we can arrive at what weights yield the best approximation of  $R$ . The least squares approach in its basic forms means fitting some line to the data, measuring the sum of squared distances from all points to the line and trying to get an optimal fit by minimising this value.

With the alternating least squares approach we use the same idea but iteratively alternate between optimizing  $U$  and fixing  $V$  and vice versa. We do this for each iteration to arrive closer to  $R = U \times V$ .

For more info read article:

<https://medium.com/radon-dev/als-implicit-collaborative-filtering-5ed653ba39fe>

So now let's check ALS for some well known movies, and see how does it recommend:

**Similar items to 'Shawshank Redemption':**

|   | MovieId | Score    | Title                             |
|---|---------|----------|-----------------------------------|
| 0 | 318     | 0.310394 | Shawshank Redemption, The (1994)  |
| 1 | 527     | 0.309371 | Schindler's List (1993)           |
| 2 | 593     | 0.308900 | Silence of the Lambs, The (1991)  |
| 3 | 296     | 0.308599 | Pulp Fiction (1994)               |
| 4 | 50      | 0.308179 | Usual Suspects, The (1995)        |
| 5 | 356     | 0.307320 | Forrest Gump (1994)               |
| 6 | 110     | 0.307277 | Braveheart (1995)                 |
| 7 | 47      | 0.306583 | Seven (a.k.a. Se7en) (1995)       |
| 8 | 480     | 0.305257 | Jurassic Park (1993)              |
| 9 | 589     | 0.303026 | Terminator 2: Judgment Day (1991) |

### Find similar items to 'Pulp Fiction':

|   | MovieId | Score    | Title                             |
|---|---------|----------|-----------------------------------|
| 0 | 296     | 0.306645 | Pulp Fiction (1994)               |
| 1 | 50      | 0.305682 | Usual Suspects, The (1995)        |
| 2 | 593     | 0.305319 | Silence of the Lambs, The (1991)  |
| 3 | 318     | 0.304871 | Shawshank Redemption, The (1994)  |
| 4 | 527     | 0.304766 | Schindler's List (1993)           |
| 5 | 47      | 0.304278 | Seven (a.k.a. Se7en) (1995)       |
| 6 | 110     | 0.303050 | Braveheart (1995)                 |
| 7 | 356     | 0.302907 | Forrest Gump (1994)               |
| 8 | 480     | 0.302016 | Jurassic Park (1993)              |
| 9 | 589     | 0.300631 | Terminator 2: Judgment Day (1991) |

### Recommend movies to userId=1:

|   | Movielid | Score    | Title   |
|---|----------|----------|---|
| 0 | 1274     | 1.341337 | Akira (1988)                                      |
| 1 | 3703     | 1.298482 | Road Warrior, The (Mad Max 2) (1981)              |
| 2 | 3508     | 1.277379 | Outlaw Josey Wales, The (1976)                    |
| 3 | 3681     | 1.274659 | For a Few Dollars More (Per qualche dollaro in... |
| 4 | 2105     | 1.272464 | Tron (1982)                                       |
| 5 | 1748     | 1.266311 | Dark City (1998)                                  |
| 6 | 2788     | 1.213520 | Monty Python's And Now for Something Completel... |
| 7 | 2951     | 1.205464 | Fistful of Dollars, A (Per un pugno di dollari... |
| 8 | 3702     | 1.189130 | Mad Max (1979)                                    |
| 9 | 2657     | 1.173790 | Rocky Horror Picture Show, The (1975)             |

### Recommend movies to userId=903:

|   | Movielid | Score    | Title  |
|---|----------|----------|--|
| 0 | 2594     | 1.369689 | Open Your Eyes (Abre los ojos) (1997)          |
| 1 | 1938     | 1.316449 | Lost Weekend, The (1945)                       |
| 2 | 2025     | 1.265688 | Lolita (1997)                                  |
| 3 | 213      | 1.247192 | Burnt by the Sun (Utomlyonnye solntsem) (1994) |
| 4 | 2874     | 1.231940 | Pajama Game, The (1957)                        |
| 5 | 760      | 1.229541 | Stalingrad (1993)                              |
| 6 | 2940     | 1.226364 | Gilda (1946)                                   |
| 7 | 5151     | 1.220052 | 40 Days and 40 Nights (2002)                   |
| 8 | 2559     | 1.191427 | King and I, The (1999)                         |
| 9 | 2774     | 1.188486 | Better Than Chocolate (1999)                   |

## 6. Deep Learning

## Second Dataset (MetaData)

The second dataset which is from Kaggle competition, provides 24 features and 45,466 samples. The information about each feature is provided below:

1. **adult:** Indicates if the movie is X-Rated or Adult.
2. **belongs\_to\_collection:** A stringified dictionary that gives information on the movie series the particular film belongs to.
3. **budget:** The budget of the movie in dollars.
4. **genres:** A stringified list of dictionaries that list out all the genres associated with the movie.
5. **homepage:** The Official Homepage of the movie.
6. **id:** The ID of the movie.
7. **imdb\_id:** The IMDB ID of the movie.
8. **original\_language:** The language in which the movie was originally shot in.
9. **original\_title:** The original title of the movie.
10. **overview:** A brief blurb of the movie.
11. **popularity:** The Popularity Score assigned by TMDB.
12. **poster\_path:** The URL of the poster image.
13. **production\_companies:** A stringified list of production companies involved with the making of the movie.
14. **production\_countries:** A stringified list of countries where the movie was shot/produced in.
15. **release\_date:** Theatrical Release Date of the movie.
16. **revenue:** The total revenue of the movie in dollars.
17. **runtime:** The runtime of the movie in minutes.
18. **spoken\_languages:** A stringified list of spoken languages in the film.
19. **status:** The status of the movie (Released, To Be Released, Announced, etc.)
20. **tagline:** The tagline of the movie.
21. **title:** The Official Title of the movie.
22. **video:** Indicates if there is a video presentation of the movie with TMDB.
23. **vote\_average:** The average rating of the movie.
24. **vote\_count:** The number of votes by users, as counted by TMDB.

## Data Wrangling

For cleaning the data, I started with:

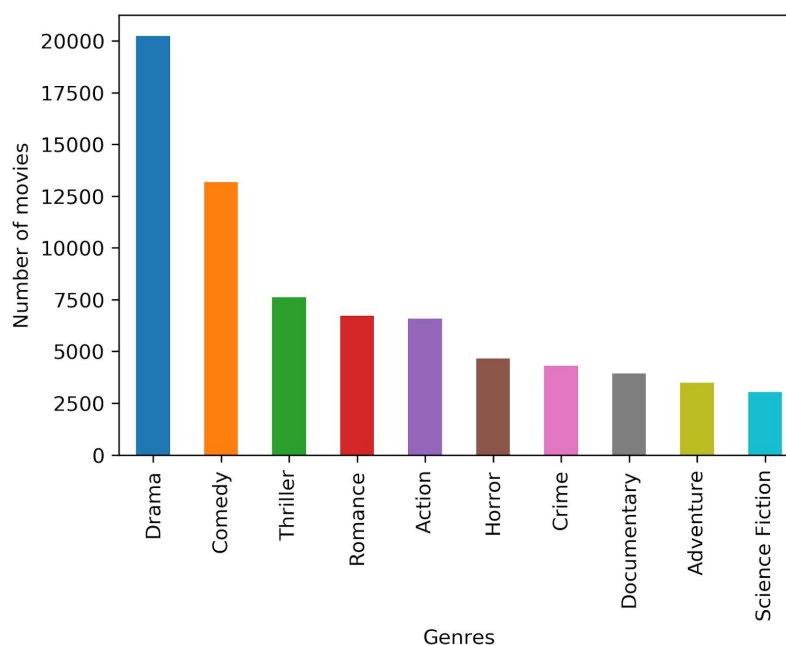
- dropping the duplicated rows.
- Just kept the rows with False and True in the adult column.
- I dropped the columns 'status', 'adult', 'homepage', 'imdb\_id', 'original\_title', 'poster\_path', 'tagline', 'video', 'spoken\_languages', since they are not providing important information for this project.
- The 'budget' and 'popularity' columns have been changed to numeric.
- The 'release\_date' has been changed to DateTime format.
- 'year' column is added for future use.
- I used the literal\_eval, apply and lambda to change the format of the 'production\_companies', 'genres', and 'production\_countries' columns.

Please check the second ipython notebook.

## Exploratory Data Analysis (EDA) and Storytelling

In this section, the various insights produced through descriptive statistics and data visualization is presented.

- Total number of movies of each genre:

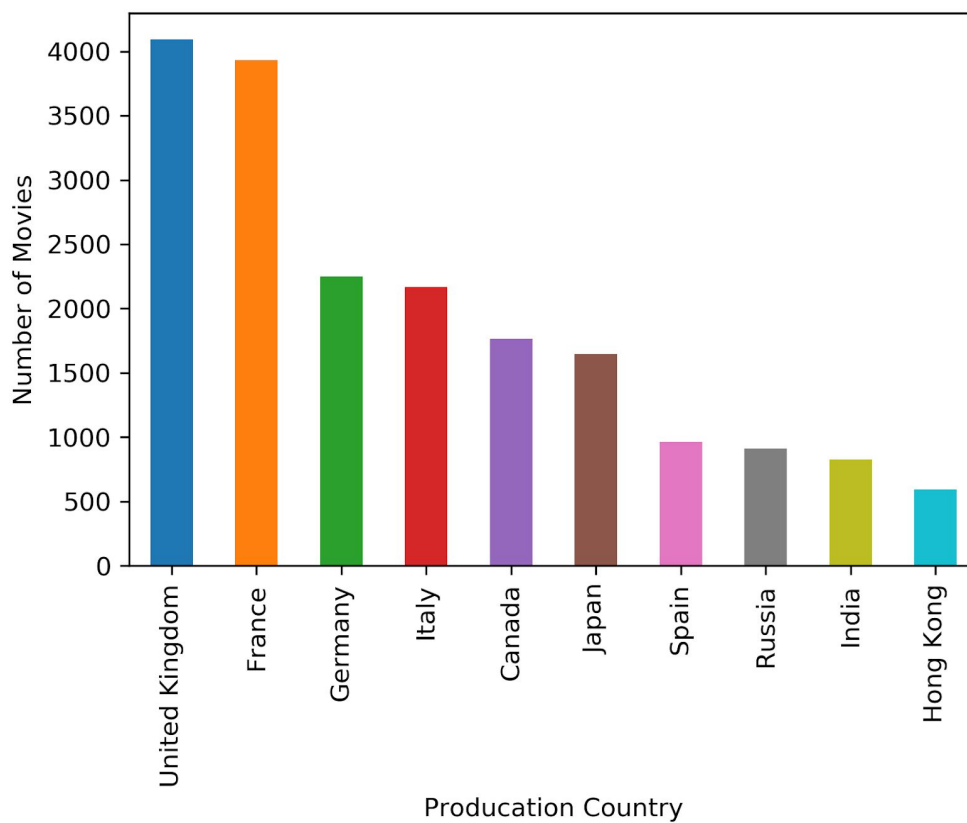


- Total production of each country is:

|                          |       |
|--------------------------|-------|
| United States of America | 21140 |
| United Kingdom           | 4091  |
| France                   | 3932  |
| Germany                  | 2249  |
| Italy                    | 2166  |
| Canada                   | 1765  |
| Japan                    | 1645  |
| Spain                    | 964   |
| Russia                   | 912   |
| India                    | 827   |

To get a better bar plot, I have ignored the USA from the list, since the USA has produced 5 times more than the United Kingdom.

After the **USA, UK, France, and Germany** have the highest number of production.



- **Revenue**

USA average revenue is: 22,534,331.039

UK average revenue is: 17,949,958.81

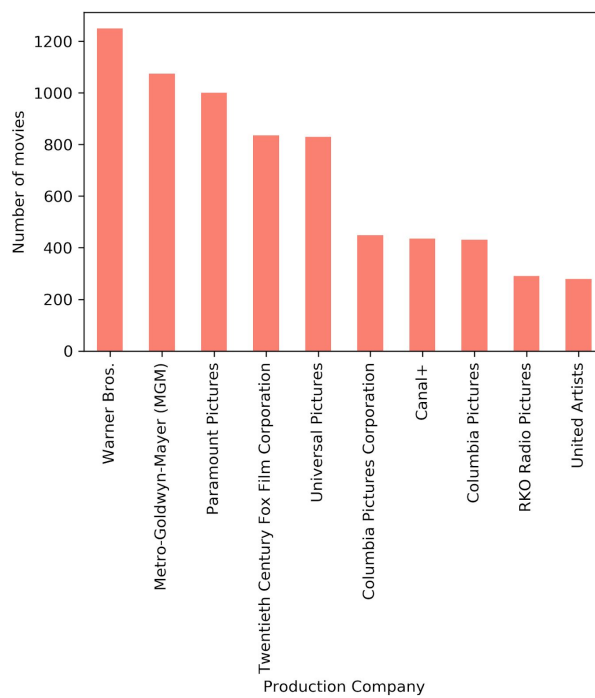
France average revenue is: 5,156,112.68

- **Production companies**

Since some movies have more than one production company, I used stack function to get the right value counts of each company.

**Warner Bros., Metro-Goldwyn-Mayer (MGM), and Paramount Pictures** have the highest number of production.

|  |      |
|--|------|
| Warner Bros.                           | 1250 |
| Metro-Goldwyn-Mayer (MGM)              | 1074 |
| Paramount Pictures                     | 1001 |
| Twentieth Century Fox Film Corporation | 836  |
| Universal Pictures                     | 830  |
| Columbia Pictures Corporation          | 448  |
| Canal+                                 | 436  |
| Columbia Pictures                      | 431  |
| RKO Radio Pictures                     | 290  |
| United Artists                         | 279  |





- **Warner Bros., Universal Pictures, and Paramount Pictures** have the highest total revenue respectively.

|   | Total        | Average      | Number |
|---|--------------|--------------|--------|
| <b>Warner Bros.</b>                           | 6.352519e+10 | 5.082015e+07 | 1250   |
| <b>Universal Pictures</b>                     | 5.525919e+10 | 6.657734e+07 | 830    |
| <b>Paramount Pictures</b>                     | 4.876940e+10 | 4.872068e+07 | 1001   |
| <b>Twentieth Century Fox Film Corporation</b> | 4.768775e+10 | 5.704276e+07 | 836    |
| <b>Walt Disney Pictures</b>                   | 4.083727e+10 | 1.552748e+08 | 263    |
| <b>Columbia Pictures</b>                      | 3.227974e+10 | 7.489498e+07 | 431    |
| <b>New Line Cinema</b>                        | 2.217339e+10 | 8.004834e+07 | 277    |
| <b>Amblin Entertainment</b>                   | 1.734372e+10 | 2.282068e+08 | 76     |
| <b>DreamWorks SKG</b>                         | 1.547575e+10 | 1.629027e+08 | 95     |
| <b>Dune Entertainment</b>                     | 1.500379e+10 | 2.308275e+08 | 65     |

- From the companies with a higher production number of 500, **Universal Pictures, Twentieth Century Fox Film Corporation, and Warner Bros.** have the highest average revenue respectively.

|   | Total        | Average      | Number |
|---|--------------|--------------|--------|
| <b>Universal Pictures</b>                     | 5.525919e+10 | 6.657734e+07 | 830    |
| <b>Twentieth Century Fox Film Corporation</b> | 4.768775e+10 | 5.704276e+07 | 836    |
| <b>Warner Bros.</b>                           | 6.352519e+10 | 5.082015e+07 | 1250   |
| <b>Paramount Pictures</b>                     | 4.876940e+10 | 4.872068e+07 | 1001   |
| <b>Metro-Goldwyn-Mayer (MGM)</b>              | 1.237679e+10 | 1.152401e+07 | 1074   |

- Comparing revenue with budget  
5775 Movies have higher revenue than budget  
5108 Movies have lower revenue than budget  
34547 movies have equal revenue and budget
- **Avatar, Star Wars: The force awakens, and Titanic are** the movies with the maximum revenue.
- The movies with the highest revenue do not have the highest popularity and vote\_average. Though the total number of votes for each movie is important.

|  | title  | popularity | vote_average | revenue      |
|--|--|------------|--------------|--------------|
|  | Avatar                                       | 185.070892 | 7.2          | 2.787965e+09 |
|  | Star Wars: The Force Awakens                 | 31.626013  | 7.5          | 2.068224e+09 |
|  | Titanic                                      | 26.889070  | 7.5          | 1.845034e+09 |
|  | The Avengers                                 | 89.887648  | 7.4          | 1.519558e+09 |
|  | Jurassic World                               | 32.790475  | 6.5          | 1.513529e+09 |
|  | Furious 7                                    | 27.275687  | 7.3          | 1.506249e+09 |
|  | Avengers: Age of Ultron                      | 37.379420  | 7.3          | 1.405404e+09 |
|  | Harry Potter and the Deathly Hallows: Part 2 | 24.990737  | 7.9          | 1.342000e+09 |
|  | Frozen                                       | 24.248243  | 7.3          | 1.274219e+09 |
|  | Beauty and the Beast                         | 287.253654 | 6.8          | 1.262886e+09 |

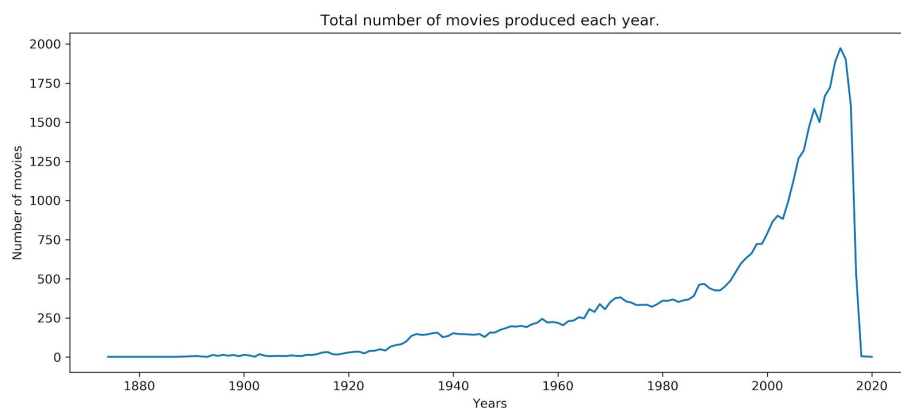
- The most voted movies are **Inception**, **The Dark Knight**, and **Avatar**.

|  | title                   | vote_count | year   |
|--|-------------------------|------------|--------|
|  | Inception               | 14075.0    | 2010.0 |
|  | The Dark Knight         | 12269.0    | 2008.0 |
|  | Avatar                  | 12114.0    | 2009.0 |
|  | The Avengers            | 12000.0    | 2012.0 |
|  | Deadpool                | 11444.0    | 2016.0 |
|  | Interstellar            | 11187.0    | 2014.0 |
|  | Django Unchained        | 10297.0    | 2012.0 |
|  | Guardians of the Galaxy | 10014.0    | 2014.0 |
|  | Fight Club              | 9678.0     | 1999.0 |
|  | The Hunger Games        | 9634.0     | 2012.0 |

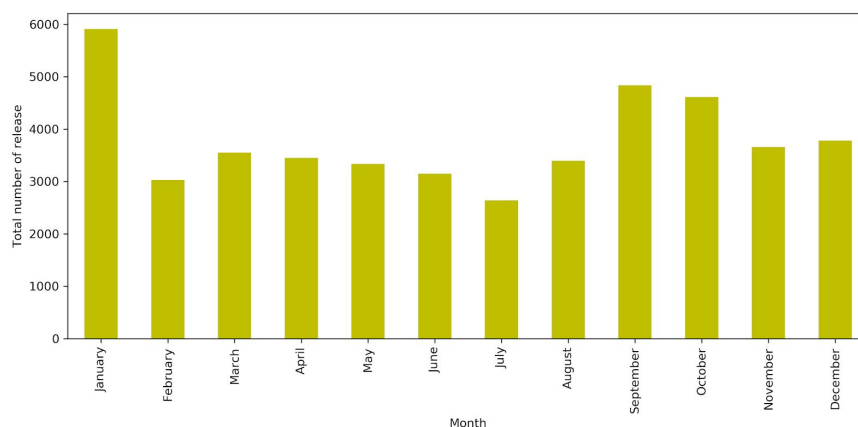
- The first three most popular movies are **Minions**, **Wonder Woman**, and **Beauty and the beast**.

|  | title                          | popularity | vote_average | revenue      |
|--|--------------------------------|------------|--------------|--------------|
|  | Minions                        | 547.488298 | 6.4          | 1.156731e+09 |
|  | Wonder Woman                   | 294.337037 | 7.2          | 8.205804e+08 |
|  | Beauty and the Beast           | 287.253654 | 6.8          | 1.262886e+09 |
|  | Baby Driver                    | 228.032744 | 7.2          | 2.245113e+08 |
|  | Big Hero 6                     | 213.849907 | 7.8          | 6.521054e+08 |
|  | Deadpool                       | 187.860492 | 7.4          | 7.831130e+08 |
|  | Guardians of the Galaxy Vol. 2 | 185.330992 | 7.6          | 8.634161e+08 |
|  | Avatar                         | 185.070892 | 7.2          | 2.787965e+09 |
|  | John Wick                      | 183.870374 | 7.0          | 8.876166e+07 |
|  | Gone Girl                      | 154.801009 | 7.9          | 3.693304e+08 |

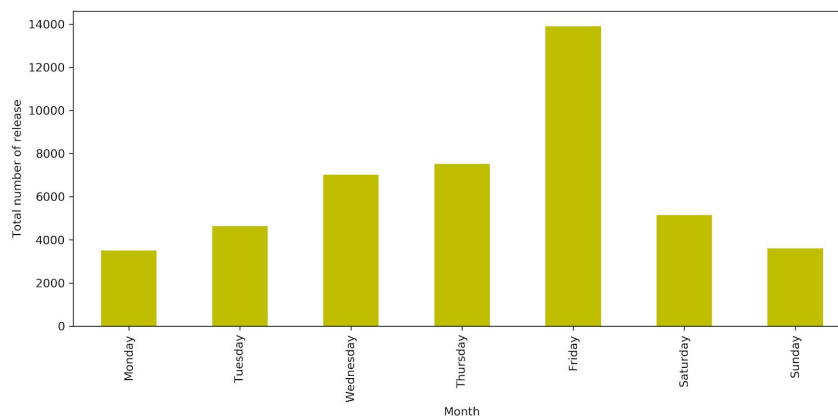
- Movies with more than 2000 votes and higher vote average are:  
**The Shawshank Redemption, Godfather, and Life is beautiful.**
- Plotting the total number of movies produced each year.



- Total number of movies release each month. (Most movies released in **January**)



- Most movies are released on **Friday**. Most movies release on the first day of each month.

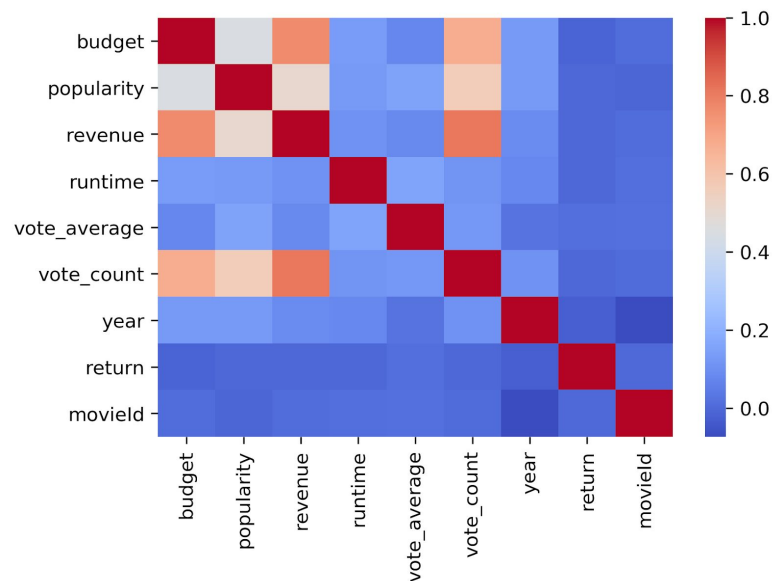


Budget and vote count are around 67% correlated

popularity and revenue are 50% correlated

popularity and vote count are 56% correlated

Revenue and vote count are 81% correlated

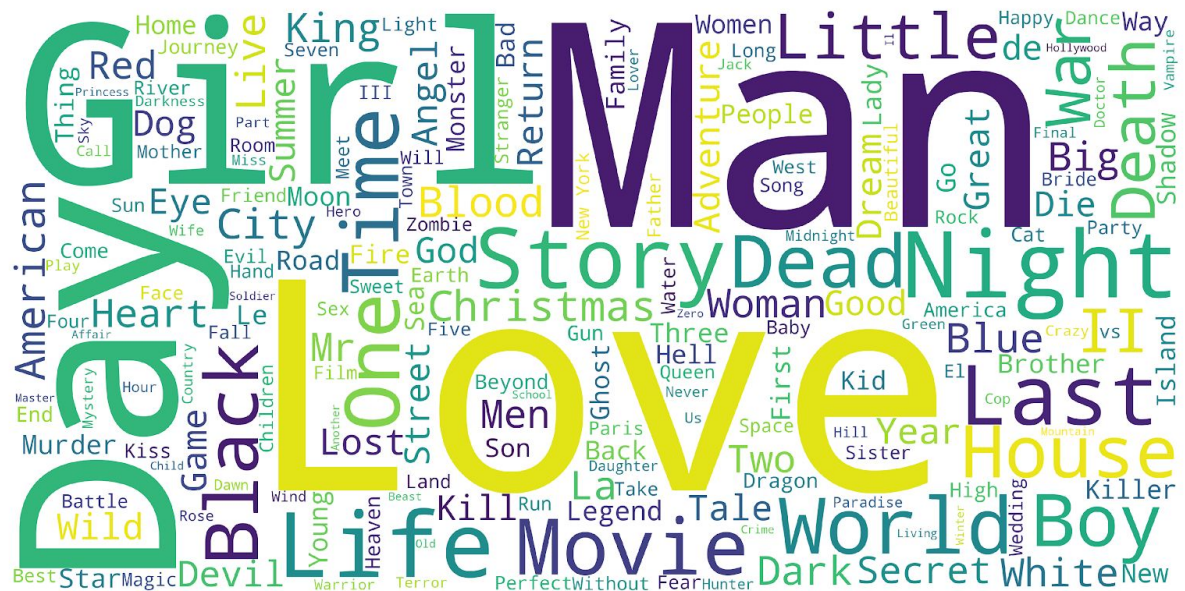


|              | budget    | popularity | revenue   | runtime   | vote_average | vote_count | year      | return    | moviedl   |
|--------------|-----------|------------|-----------|-----------|--------------|------------|-----------|-----------|-----------|
| budget       | 1.000000  | 0.449682   | 0.768825  | 0.134700  | 0.073496     | 0.676699   | 0.131647  | -0.012572 | 0.010009  |
| popularity   | 0.449682  | 1.000000   | 0.506221  | 0.129912  | 0.154357     | 0.559995   | 0.131634  | -0.003946 | -0.009621 |
| revenue      | 0.768825  | 0.506221   | 1.000000  | 0.103948  | 0.083883     | 0.812031   | 0.088358  | -0.005515 | 0.007910  |
| runtime      | 0.134700  | 0.129912   | 0.103948  | 1.000000  | 0.158192     | 0.113555   | 0.078714  | -0.005189 | 0.012001  |
| vote_average | 0.073496  | 0.154357   | 0.083883  | 0.158192  | 1.000000     | 0.123611   | 0.025829  | 0.013161  | 0.015532  |
| vote_count   | 0.676699  | 0.559995   | 0.812031  | 0.113555  | 0.123611     | 1.000000   | 0.106797  | -0.003041 | 0.005362  |
| year         | 0.131647  | 0.131634   | 0.088358  | 0.078714  | 0.025829     | 0.106797   | 1.000000  | -0.024818 | -0.073434 |
| return       | -0.012572 | -0.003946  | -0.005515 | -0.005189 | 0.013161     | -0.003041  | -0.024818 | 1.000000  | 0.000810  |
| moviedl      | 0.010009  | -0.009621  | 0.007910  | 0.012001  | 0.015532     | 0.005362   | -0.073434 | 0.000810  | 1.000000  |

- **Title and Overview Word Clouds**

There are certain words that use more often in titles and overviews. I use WordCloud library to find out what are these words.

The word **Love** is the most commonly used word in movie titles. **Girl**, **Day** and **Man** are also among the most commonly occurring words.



**Life** is the most commonly used word in Movie titles. **One** and **Find** are also popular.





The Popularity score seems to be an extremely skewed quantity with a mean of only 2.9 but maximum values reaching as high as 547, which is almost 1800% greater than the mean. However, as can be seen from the distribution plot, almost all movies have a popularity score of less than 10 (the 75th percentile is at 3.678902).

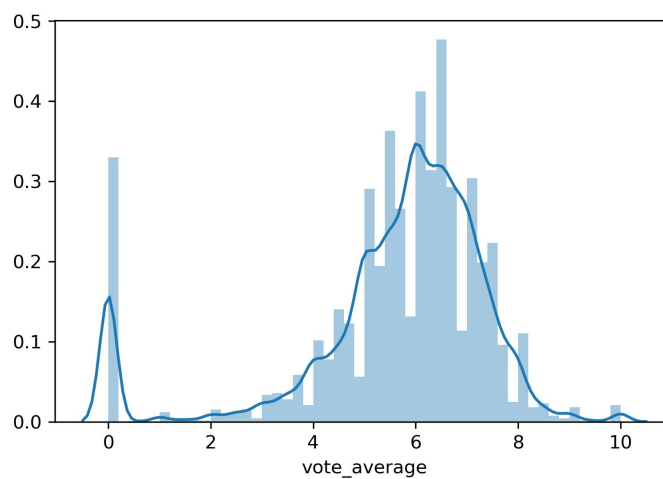
A histogram showing the frequency of the number of non-zero elements in the matrix. The x-axis is labeled 'Number of non-zero elements' and ranges from 0 to 500. The y-axis is labeled 'Frequency' and is on a logarithmic scale from  $10^0$  to  $10^4$ . The distribution is highly skewed, with a peak frequency of approximately 30,000 for matrices with 0 non-zero elements.

| Number of non-zero elements | Frequency |
|-----------------------------|-----------|
| 0                           | ~30,000   |
| 50                          | ~15       |
| 100                         | ~8        |
| 150                         | ~5        |
| 200                         | ~4        |
| 250                         | ~1        |
| 300                         | ~2        |
| 500                         | ~1        |

|       | title                          | popularity | year |
|-------|--------------------------------|------------|------|
| 30700 | Minions                        | 547.488298 | 2015 |
| 33356 | Wonder Woman                   | 294.337037 | 2017 |
| 42222 | Beauty and the Beast           | 287.253654 | 2017 |
| 43644 | Baby Driver                    | 228.032744 | 2017 |
| 24455 | Big Hero 6                     | 213.849907 | 2014 |
| 26564 | Deadpool                       | 187.860492 | 2016 |
| 26566 | Guardians of the Galaxy Vol. 2 | 185.330992 | 2017 |
| 14551 | Avatar                         | 185.070892 | 2009 |
| 24351 | John Wick                      | 183.870374 | 2014 |
| 23675 | Gone Girl                      | 154.801009 | 2014 |

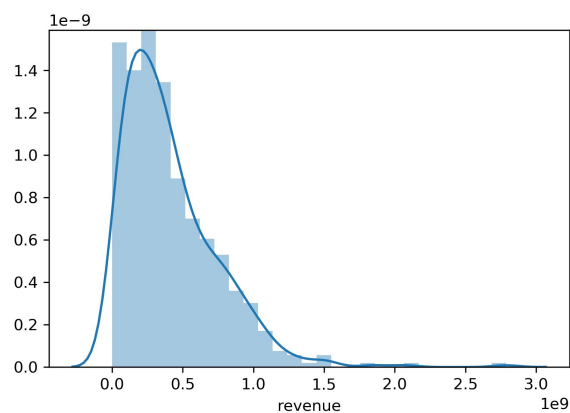
### Vote average

The mean rating is only a 5.6 on a scale of 10. Half the movies have a rating of less than or equal to 6.



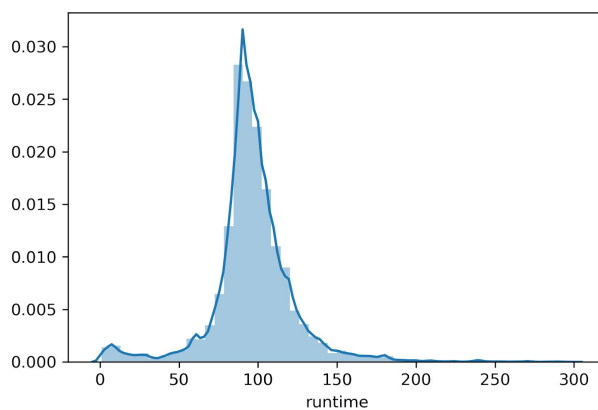
### Revenue

The revenue statistics for movies with more than 2000 votes. The distribution plot is skewed to the right.



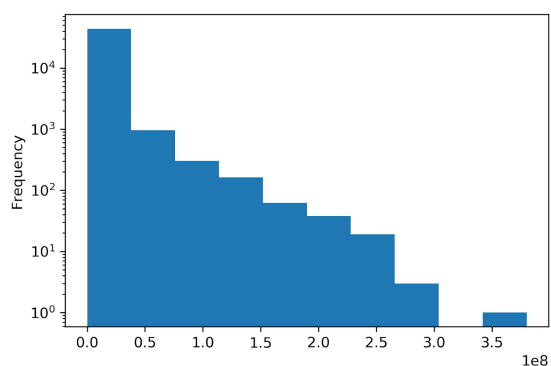
## Runtime

The average length of a movie is about 1 hour and 30 minutes. The longest movie on record in this dataset is a staggering 1256 minutes (or 20 hours) long.



## Budget

The distribution of movie budgets shows an exponential decay. More than 75% of the movies have a budget smaller than 25 million dollars.





## Shortest and longest movies

- We can see there is several one minute length movies.

| runtime | title   | year   |
|---------|---|--------|
| 1.0     | Mr. Edison at Work in His Chemical Laboratory     | 1897.0 |
| 1.0     | Grandma's Reading Glass                           | 1900.0 |
| 1.0     | What Happened on Twenty-Third Street, New York... | 1901.0 |
| 1.0     | The Magician                                      | 1898.0 |
| 1.0     | Panorama pris d'un train en marche                | 1898.0 |
| 1.0     | Divers at Work on the Wreck of the "Maine"        | 1898.0 |
| 1.0     | After the Ball                                    | 1897.0 |
| 1.0     | Between Calais and Dover                          | 1897.0 |
| 1.0     | The Surrender of Tournavos                        | 1897.0 |
| 1.0     | Blacksmith Scene                                  | 1893.0 |

- All these long time films are TV series and we don't have access to the runtime of each episode in this dataset.

| runtime | title                               | year   |
|---------|-------------------------------------|--------|
| 1256.0  | Centennial                          | 1978.0 |
| 1140.0  | Jazz                                | 2001.0 |
| 1140.0  | Baseball                            | 1994.0 |
| 931.0   | Berlin Alexanderplatz               | 1980.0 |
| 925.0   | Heimat: A Chronicle of Germany      | 1984.0 |
| 900.0   | The Story of Film: An Odyssey       | 2011.0 |
| 877.0   | Taken                               | 2002.0 |
| 874.0   | The War                             | 2007.0 |
| 840.0   | The Roosevelts: An Intimate History | 2014.0 |
| 840.0   | Seventeen Moments in Spring         | 1973.0 |

## Recommender Systems

### 1. Simple Recommender

The Simple Recommender offers generalized recommendations to every user based on movie popularity, ratings, number of votes and genre. The basic idea behind this recommender is that movies that are more popular and more critically acclaimed will have a higher probability of being liked by the average audience. This model does not give personalized recommendations based on the user.


So I sort the movies based on their genre, and I recommend the ones with the highest number of votes and the higher average rankings to the new audience.

- The list of the **15 best Drama** movies to recommend are:

|    |                          |
|----|--------------------------|
| 0  | The Dark Knight          |
| 1  | Interstellar             |
| 2  | Fight Club               |
| 3  | The Shawshank Redemption |
| 4  | Forrest Gump             |
| 5  | The Godfather            |
| 6  | The Intouchables         |
| 7  | Schindler's List         |
| 8  | Whiplash                 |
| 9  | Leon: The Professional   |
| 10 | The Green Mile           |
| 11 | Life Is Beautiful        |
| 12 | The Godfather: Part II   |
| 13 | The Usual Suspects       |
| 14 | GoodFellas               |

- The list of the **15 best Romance** movies to recommend are:

|   |                                       |
|---|---------------------------------------|
| 0 | Forrest Gump                          |
| 1 | Titanic                               |
| 2 | La La Land                            |
| 3 | Her                                   |
| 4 | The Great Gatsby                      |
| 5 | The Fault in Our Stars                |
| 6 | Eternal Sunshine of the Spotless Mind |
| 7 | Edward Scissorhands                   |


- 
- 8 Aladdin
  - 9 Amélie
  - 10 The Theory of Everything
  - 11 The Curious Case of Benjamin Button
  - 12 The Notebook
  - 13 A Beautiful Mind
  - 14 The Perks of Being a Wallflower

- The list of the **15 best Action** movies to recommend are:

- 0 Inception
- 1 The Dark Knight
- 2 Avatar
- 3 The Avengers
- 4 Deadpool
- 5 Guardians of the Galaxy
- 6 Mad Max: Fury Road
- 7 The Dark Knight Rises
- 8 The Matrix
- 9 Iron Man
- 10 The Lord of the Rings: The Fellowship of the Ring
- 11 The Lord of the Rings: The Return of the King
- 12 Star Wars: The Force Awakens
- 13 The Lord of the Rings: The Two Towers
- 14 Batman Begins

- The list of the **15 best Fantasy** movies to recommend are:

- 0 Avatar
- 1 The Lord of the Rings: The Fellowship of the Ring
- 2 The Lord of the Rings: The Return of the King
- 3 Star Wars: The Force Awakens
- 4 The Lord of the Rings: The Two Towers
- 5 Pirates of the Caribbean: The Curse of the Bla...
- 6 Harry Potter and the Philosopher's Stone
- 7 X-Men: Days of Future Past
- 8 Harry Potter and the Deathly Hallows: Part 2
- 9 Harry Potter and the Prisoner of Azkaban
- 10 Harry Potter and the Chamber of Secrets



|    |  |
|----|--|
| 11 | Doctor Strange                               |
| 12 | Harry Potter and the Goblet of Fire          |
| 13 | Harry Potter and the Deathly Hallows: Part 1 |
| 14 | Harry Potter and the Order of the Phoenix    |

## 2. IMDB Weighted Rating Formula

Another technique could be IMDB's **weighted rating** formula which is mathematically represented as follows:

$$WR = \frac{v}{v+m}R + \frac{m}{v+m}C$$

where,

$v$  is the number of votes for the movie

$m$  is the minimum votes required to be listed in the chart

$R$  is the average rating of the movie


$C$  is the mean vote across the whole report

The next step is to determine an appropriate value for  $m$ , the minimum votes required to be listed in the chart. We will use **95th percentile** as our cutoff. In other words, for a movie to feature in the charts, it must have more votes than at least 95% of the movies in the list.

- mean vote across the whole report ( $C$ ): 5.245
- minimum votes required to be listed in the chart ( $m$ ): 434.0
- 2181 movies are qualified

The **best 30 movies** to recommend base on IMDB weighting are:

|       |   |
|-------|---|
| 15480 | Inception   |
| 12481 | The Dark Knight                                   |
| 22879 | Interstellar                                      |
| 2843  | Fight Club  |
| 4863  | The Lord of the Rings: The Fellowship of the Ring |
| 292   | Pulp Fiction                                      |
| 314   | The Shawshank Redemption                          |
| 7000  | The Lord of the Rings: The Return of the King     |
| 351   | Forrest Gump                                      |
| 5814  | The Lord of the Rings: The Two Towers             |
| 256   | Star Wars   |
| 1225  | Back to the Future                                |
| 834   | The Godfather                                     |
| 1154  | The Empire Strikes Back                           |
| 46    | Se7en   |



|       |                          |
|-------|--------------------------|
| 24860 | The Imitation Game       |
| 359   | The Lion King            |
| 18465 | The Intouchables         |
| 22841 | The Grand Budapest Hotel |
| 586   | The Silence of the Lambs |
| 11354 | The Prestige             |
| 522   | Schindler's List         |
| 23673 | Whiplash                 |
| 289   | Leon: The Professional   |
| 4099  | Memento                  |
| 3030  | The Green Mile           |
| 5481  | Spirited Away            |
| 1213  | The Shining              |
| 1057  | Reservoir Dogs           |
| 2211  | Life Is Beautiful        |

### 3. Matrix Factorization-based algorithms

The idea is basically to take a large (or potentially huge) matrix and factor it into some smaller representation of the original matrix. You can think of it in the same way as we would take a large number and factor it into two much smaller primes. We end up with two or more lower dimensional matrices whose product equals the original one.

When we talk about collaborative filtering for recommender systems we want to solve the problem of our original matrix having millions of different dimensions, but our “tastes” not being nearly as complex. Even if i’ve viewed hundreds of items they might just express a couple of different tastes. Here we can actually use matrix factorization to mathematically reduce the dimensionality of our original “all users by all items” matrix into something much smaller that represents “all items by some taste dimensions” and “all users by some taste dimensions”. These dimensions are called latent or hidden features and we learn them from our data.

Doing this reduction and working with fewer dimensions makes it both much more computationally efficient and but also gives us better results since we can reason about items in this more compact “taste space”.

If we can express each user as a vector of their taste values, and at the same time express each item as a vector of what tastes they represent. You can see we can quite easily make a recommendation. This also gives us the ability to find connections between users who have no specific items in common but share common tastes.

**SVD** is a matrix factorization technique that is usually used to reduce the number of features of a data set by reducing space dimensions from  $N$  to  $K$  where  $K < N$ . For the

([https://medium.com/@m\\_n\\_malaeb/singular-value-decomposition-svd-in-recommender-systems-for-non-math-statistics-programming-4a622de653e9](https://medium.com/@m_n_malaeb/singular-value-decomposition-svd-in-recommender-systems-for-non-math-statistics-programming-4a622de653e9))

In the first part, the **vote\_average** is predicted from the **vote\_average** and the **popularity**.

[illegible]

- Evaluating RMSE, MAE of algorithm SVD on 5 split(s).

[illegible]

## Machine Learning

In this section, I applied different machine learning techniques to find the best model with the lowest RMSE.

### 4. Random Forest

Random forest is a type of supervised machine learning algorithm based on ensemble learning. The random forest algorithm combines multiple algorithms of the same type i.e. multiple decision trees, resulting in a forest of trees, hence the name "Random Forest".

I check random forest regressor to find the vote\_average base on some features 'id', 'popularity', 'moviend', 'vote\_count', 'revenue', 'budget'.

I have checked a different number of estimators to get the minimum RMSE.

Total Elapsed time with the model is: 23.899521589279175

RMSE value for n\_estimator= 100 is: 1.230501506418753

Total Elapsed time with the model is: 32.894049882888794

RMSE value for n\_estimator= 110 is: 1.2297851910704445

Total Elapsed time with the model is: 34.64072108268738

RMSE value for n\_estimator= 120 is: 1.2291184320901682

Total Elapsed time with the model is: 36.02364206314087

RMSE value for n\_estimator= 130 is: 1.2285653671442673

Total Elapsed time with the model is: 36.642759799957275

RMSE value for n\_estimator= 140 is: 1.228152851883824

Total Elapsed time with the model is: 42.90695285797119

RMSE value for n\_estimator= 150 is: 1.2278656497690312

Total Elapsed time with the model is: 43.093995809555054

RMSE value for n\_estimator= 160 is: 1.2273117394335689

Total Elapsed time with the model is: 48.45515489578247

RMSE value for n\_estimator= 170 is: 1.2268291900261832

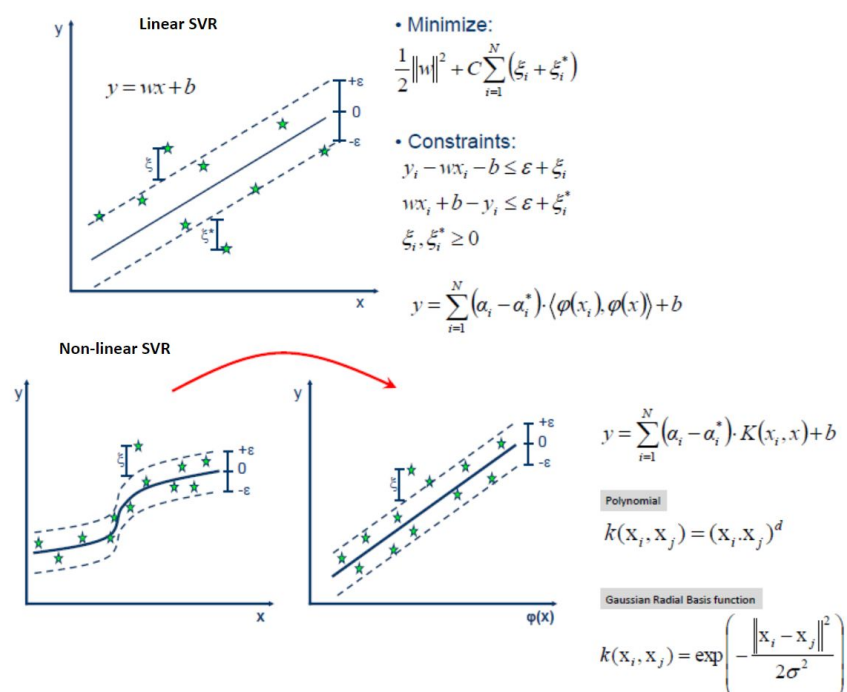
Total Elapsed time with the model is: 47.292954206466675  
 RMSE value for n\_estimator= 180 is: 1.2270316755432094

Total Elapsed time with the model is: 53.971041202545166  
 RMSE value for n\_estimator= 190 is: 1.22688515828762

The minimum RMSE (**1.268**) is achieved with n\_estimator=**170**.

## 5. Support Vector Regression

Support Vector Machine can also be used as a regression method, maintaining all the main features that characterize the algorithm (maximal margin). The Support Vector Regression (SVR) uses the same principles as the SVM for classification, with only a few minor differences. First of all, because output is a real number it becomes very difficult to predict the information at hand, which has infinite possibilities. In the case of regression, a margin of tolerance (epsilon) is set in approximation to the SVM which would have already requested from the problem. But besides this fact, there is also a more complicated reason, the algorithm is more complicated therefore to be taken in consideration. However, the main idea is always the same: to minimize error, individualizing the hyperplane which maximizes the margin, keeping in mind that part of the error is tolerated.





Total Elapsed time with model is: 84.37661409378052  
SVR RMSE (gamma=.01, C=100) is: 1.8604387694736897

Total Elapsed time with model is: 68.05094337463379  
SVR RMSE(gamma=0.001, C=100) is: 1.8819087974081985

Total Elapsed time with model is: 63.12529253959656  
SVR RMSE(gamma=0.001, C=10) is: 1.888942925254787

As it can see from the results, SVR is slow and the RMSE is not as small as random forest.

## 6. K-Nearest Neighbor

KNN is fast but the RMSE is not as small as random forest.


RMSE value for k= 11 is: 1.8038207834268334  
RMSE value for k= 13 is: 1.7934131089891299  
RMSE value for k= 15 is: 1.78986557123273  
RMSE value for k= 17 is: 1.786579572380617  
RMSE value for k= 19 is: 1.783226039661783

## Conclusion

This report highlighted the processes of data wrangling, storytelling, EDA, data visualization, inferential statistics, recommender systems and machine learning techniques to perform on two movie datasets.

In this work, several recommended systems are implemented as:

- 1. Simple Recommender** used the overall Vote Count and Vote Averages to build Top Movies Charts, in general and for a specific genre.
- 2. IMDB Weighted Rating** system was used to calculate ratings on which the sorting was finally performed and recommend the best 30 movies of the datasets.
- 3. Content Based Recommender** was built to recommend based on the user's rating history.



**4. Collaborative Filtering** in a very simple format could recommend based on other user's rating histories. It could also be based on matrix factorization techniques like SVD or ALS.

**5. Machine learning:** Random Forest is a fast and good technique to predict when we have several features in our dataset. Support Vector Machines are strong machine learning tools for classification and regression however tuning its parameters need optimization techniques and the training process is slow.