IMDB DATA ANALYSIS

Team

Hiral Patel, Ishita Agrawal, Pooja Mutreja, Sanjana Shashikanth

CSP 571 Data Preparation & Analysis

INTRODUCTION & OBJECTIVE

In today’s world, the film industry is becoming increasingly competitive. According to IMDB, the average number of films has been produced every year is 2577. In 2016, there were 9500 films produced in the world. It is impossible for all those movies to survive in this fierce competitive market. Hence, very few movies taste success and are ranked high. Given the low success rate, models, and mechanisms to predict reliably the ranking and / or box office collections of a movie can help de-risk the business significantly and increase average returns. It is hard to know the audience’s liking before seeing their reviews. Various stakeholders such as actors, financiers, directors etc. can use these predictions to make more informed decisions. Some of the questions that can be answered using prediction models are given below:

1. Does the cast or director matter in the success or ranking of a movie?

2. Is the genre of the movie a key determinant of rank or success?

3. Does duration matter?

Movie studios make many decisions both pre-production and post-production. One of the major decisions they must take is finding an optimal release date for the movie to maximize its performance by gaining as much market share as possible.

Since predicting exact release date is a complex task, we have also worked on predicting release season.

IMDB RATING

IMDB offers a rating / ranking scale that allows users to rate films by choosing one of ten categories in the range 1–10, with each user able to submit one rating. The points of reference given to users of these categories are the descriptions "1 (awful)" and 3 "10 (excellent)"; and these are the only descriptions of categories. Though the current formula is not disclosed, IMDb originally used the following formula to calculate their weighted rating:[1]

$$W = \frac{(R.V + C.M)}{(V + M)}$$

where:

W = weighted rating

R = average for the movie as a number from 1 to 10 (mean) = (Rating)

V = number of votes for the movie = (votes)

M = minimum votes required to be listed in the Top 250 (currently 25,000)

C = the mean vote across the whole report (currently 7.0)

DATA

We have scraped www.imdb.com website to fetch data required for our analysis like rating, content rating, duration, title, director/actor information, plot keywords etc. To perform web scraping we have used Python3 and BeautifulSoup library.

BeautifulSoup: BeautifulSoup provides a few simple methods and Pythonic idioms for navigating, searching, and modifying a parse tree: a toolkit for dissecting a document and extracting what you need. It doesn't take much code to write an application. Beautiful Soup sits on top of popular Python parsers like lxml and html5lib, allowing you to try out different parsing strategies or trade speed for flexibility.

Firstly, we scraped movie titles and their links. Then scraped information for those movie titles using their respective links.

Final Variables that we have use are:

Numerical variables:

movie_facebook_likes	Number of facebook likes for the movie
actor_1_facebook_likes	Number of facebook likes for actor 1
actor_2_facebook_likes	Number of facebook likes for actor 2
actor_3_facebook_likes	Number of facebook likes for actor 3
director_facebook_likes	Number of facebook likes for the director
Gross	Total earnings of the movie
Budget	Total budget of the movie
Duration	Total running time of the movie
Movie_Age	How old is the movie (calculated using title_year)
num_critic_for_reviews	Number of critics who have given their reviews
num_voted_users	Number of users who have given ratings for the movies
num_user_for_reviews	Number of users who have given their reviews

Categorical variables:

content_rating	Movie rating based on its content given by MPAA
genres	Category of the movie i.e., horror, romance, drama etc.
plot_keywords	Keywords related to plot of the movie

DATA CLEANING

Since the scraped data was incoherent to read and use, a lot of data cleaning was required. To clean the scraped data, we used "**Open-Refine**" tool. Open-Refine is a free, open source power tool for working with messy data and improving it.

Steps in Cleaning:

1. Retaining only one column of directors' information from the scraped data.
2. Removing characters like ',' , []
3. Creating new columns for the array of Actors and their respective facebook likes.

4. Converting the Content Rating from old format to new format. NOT RATED and UNRATED are combined and are given a common name.

Old format	Retained	Meaning
Rated G	G	General Audiences – all ages admitted
Rated GP, M, M/PG	PG	Parental Guidance Suggested – some material may not be suitable for children
Rated PG-13	PG-13	Parents Strongly Cautioned – some material may be inappropriate for children under 13
Rated R	R	Restricted – under 17 requires accompanying parent or adult guardian
Rated NC-17	NC-17	No children under 17 admitted [1990–1996] / No one 17 and under admitted [1996–present]

5. Removing data related to TV series and Video Series

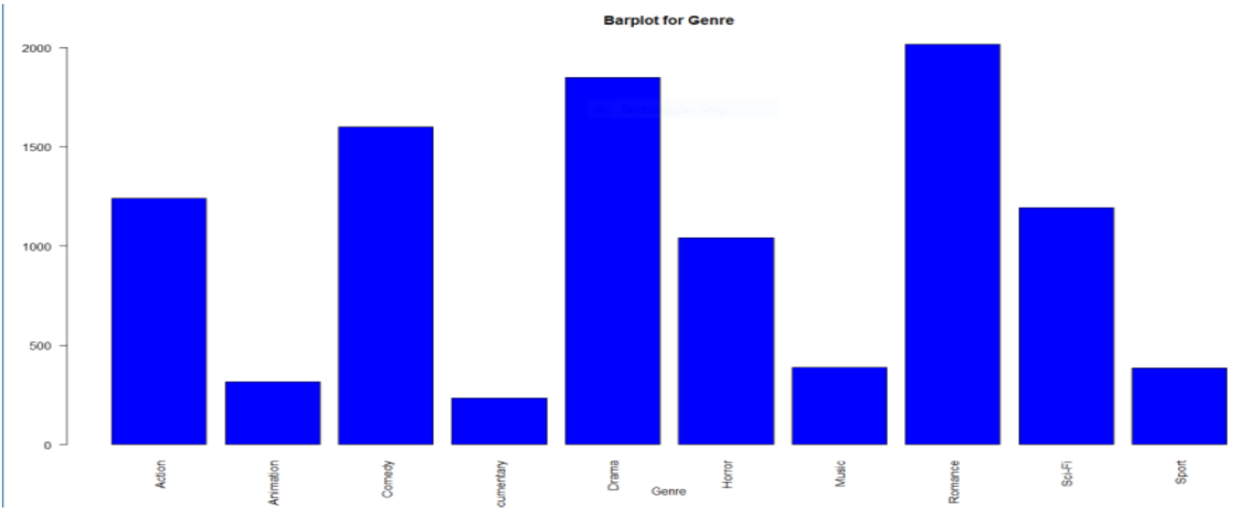
6. Converting the gross to one common currency US $.

7. Changed K to 1000 in gross and budget.

EXPLORATORY DATA ANALYSIS

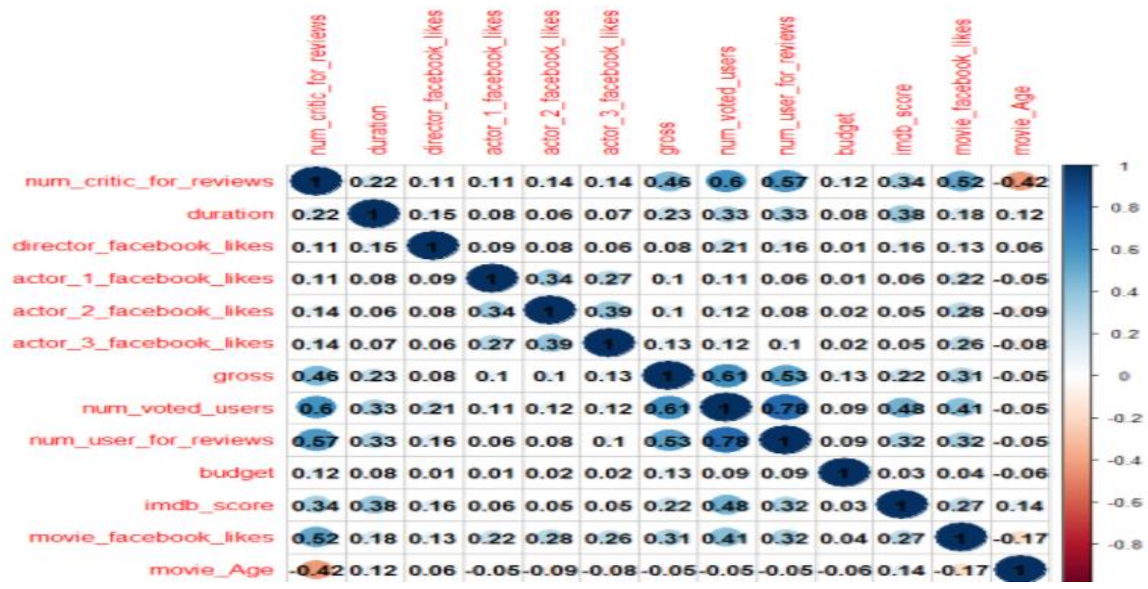
Performed EDA on the data set for the visualizations and better understanding of how the data is distributed and how variables are affecting the imdb_score.

- Below is the Barplot for Genre. We can see that “Romance” genre movies are more in our dataset followed by Drama, Comedy, Action etc.



- The Correlation Plot is as shown below:

  I. The highest positive correlation of 0.78 is seen between the number of voted users and the number of users who have given reviews. Meaning, the ratings by users rely largely on reviews given by the audience.
  II. We can also see high positive correlation between gross and number of voted users, meaning that earnings of the movie is largely impacted by the ratings given by the users.
  III. There are also some interesting pairwise correlations that are counter-intuitive. For example, director and actor facebook likes have a less positive correlation with imdb_score meaning that the popularity of a director or an actor on social media does not have an impact on how much money his/her movie will earn.



Note: More visualizations can be found in appendix.

TREATING MISSING VALUES & OUTLIERS

Missing value count:

Duration	290
Director FB Likes	154
Actor FB Likes	407
Gross	2651
Budget	2258

KNN Imputation: We have use this technique to replace the missing values for all our variables. KNN imputation helps in replacing the missing values with the data corresponding value from the nearest-neighbor column. The nearest-neighbor column is the closest column in Euclidean distance. If the corresponding value from the nearest-neighbor column is also Nan, the next nearest column is used. The advantage is that you could impute all the missing values in all variables with one call to the function. It takes the whole data frame as the argument and you don't even have to specify which variable you want to impute.

Usage:

```
knnImputation(data, k = 10, scale = T, meth = "weighAvg",distData = NULL)
```

Outliers: The dataset that we have has various values in it across the variables. For some of the movies the gross is around 460K and for other is it as high as 652177K. Though the difference is huge we can't consider them as outliers because there are few hit movies and few flops. Thus, in this data set we don't have any outliers as such. The values across the variables solely depends on how well or how bad the movie did.

APPLYING MODELS

We have divided our dataset into 80:20 as train and test set. As our target variable is to predict the imdb_score of movie which is continuous thus, we have used **regression**. We have used **R-Square** as the measure to compare performance across the models.

Various models and their respective parameter settings:

- Random Forest

  randomForest(imdb_score~.,data=training, mtry=6,importance=TRUE, type = "regression")

  Used **tuneRF()** to find the best mtry value

  Parameters:

  mtry: Number of variables randomly sampled as candidates at each split.

  importance: Should importance of predictors be assessed?

  type: one of regression, classification, or unsupervised.

- Logistic Regression

  For this model, we scaled our target variable imdb_score in the range 0 to 1.

  glm(modelForm, family=binomial(link='logit'),data=trainData)

  Parameters:

  family: A description of the error distribution and link function to be used in the model.

- Gradient Boosting

  gbm(imdb_score~.,data=training,distribution="gaussian",n.trees=5000,interaction.depth=4)

  Parameters:

  distribution: either a character string specifying the name of the distribution to use or a list with a component name specifying the distribution and any additional parameters needed

  n.trees: the total number of trees to fit.

  interaction.depth: The maximum depth of variable interactions. 1 implies an additive model, 2 implies a model with up to 2-way interactions, etc.

- Linear Regression (Baseline Model)

  linearModel = lm(modelForm, data=trainData)

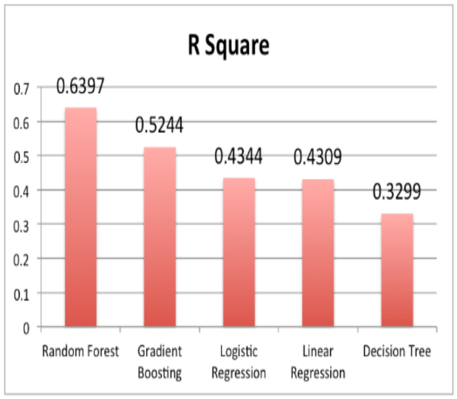
- Decision Tree

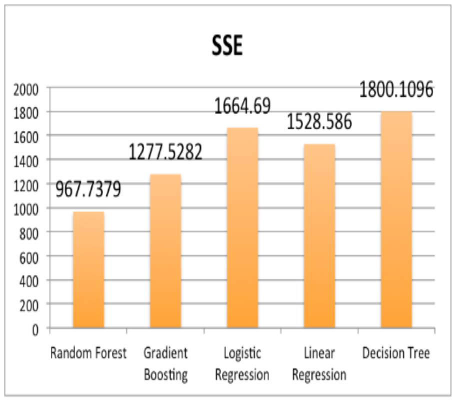
  rpart(imdb_score~.-imdb_score_01, data=trainData,method="anova")

  Parameters:

  method: one of "anova", "poisson", "class" or "exp". ‘anova’ is for regression’

Model Comparison based on R-Square:





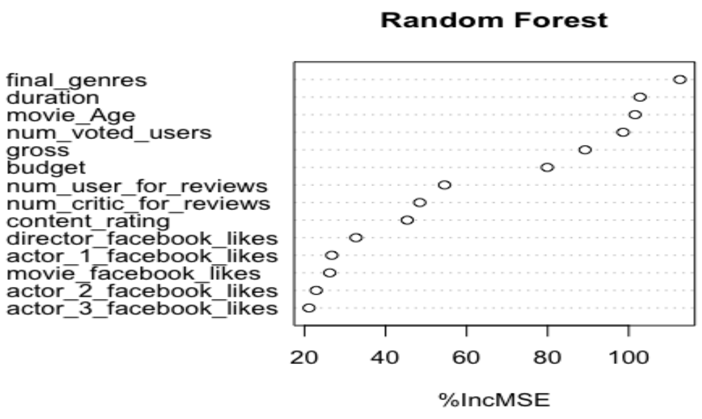
R-Square can be calculated using given formula :

$$R^2 = \frac{SS_R}{SS_T}.$$

SS_R : Explained Variance (variance of the model's predictions)

SS_T : Total Variance (sample variance of the dependent variable)

We can see that **Random Forest** was the best model with a higher **R-square** value of **0.6397** and a low Sum of Square of Errors of 967.73.



Feature ranking using Random Forest

RELEASE SEASON PREDICTION

Release seasons were defined as follows:

Release Months	*Release Season*
1,2	Winter
3,4	Spring
5,6,7	Summer
8,9,10	Fall
11, 12	Holiday

We fitted decision tree and random forest to predict release season using the following features:

- Genre
- Budget
- Content Rate
- Duration
- Director and actor Facebook likes

Unfortunately, both models gave us an accuracy around 0.3. Thus, we tried another approach.

PREDICTING RELEASE SEASON USING TOPIC MODELING

The assumption behind this approach is that in past movies have been released considering all factors such as not producing similar movies together, not producing movies of same actors/ directors/ production house together.

Thus, if we can imitate the process in which movies have been released in past we should be successful.

Following are the steps involved in the process of predicting release season of a movie:

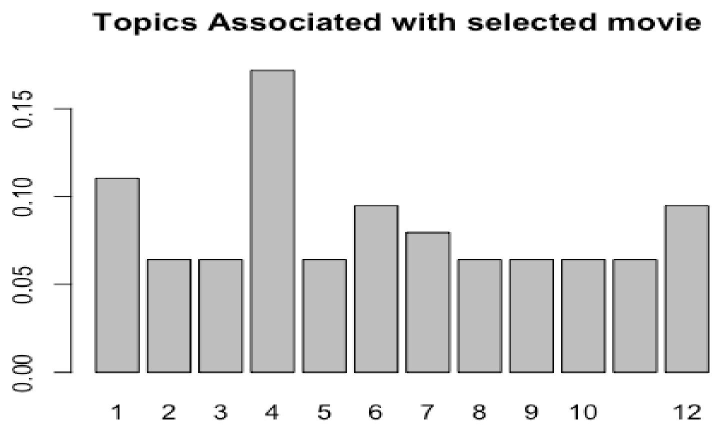
1. Representing movies in a topic space using **Latent Dirichlet Allocation** on plot keywords.
2. Finding top 10 movies most similar to the given movie.
3. Find the frequency of each release season in the top 10 similar movies
4. Predict the most frequent release season as the optimal release season.

LDA: Latent Dirichlet Allocation is a very popular approach to do topic modeling. Topics in LDA are a group of words with their probability of occurrence in the topic and documents in LDA are represented as distribution of topics with different probabilities.

We used "text mining" package for cleaning and processing plot keywords and "topic models" package for applying LDA. We chose 12 topics as the appropriate number of topics for our data as log likelihood increases slowly after that. A better estimate for number of topics can be found out if LDA is run on larger data set and on more number of plot keywords for each movie.

Example of release season prediction:

Let's try to predict release date for "Harry Potter and Deathly Hallows: Part 2".
Following is topic distribution for "Harry Potter and Deathly Hallows: Part 2":



We can see that highest probability is of topic 4, which means topic 4 is most dominant in this movie. Below is the frequency of each release season in 10 most similar movies to harry potter:

Season	Number of Movies
Spring	3
Summer	4
Fall	2
Holiday	1

Here we can see that summer was the most popular release season for similar movies. Thus, our **prediction** will be **summer**.

Actual release date of this movie was 7/15/2011.

CONCLUSION

- Random forest is the best model for predicting rating. Though the R-square is still 0.6379, there is still scope of improvement.
- Topic modeling is an interesting approach towards predict release season and can be explored further.
- We learnt about scarping. Since we collected data there was a lot of cleaning involved, we explored OpenRefine for this.
- This was our first time extensively working on regression problem; we got exposure to regression version of various models.

FUTURE WORK

- R-square of predicting rating can be increased by considering various other features such as number of countries and languages movie was released in, the production house of movie, how widely movie was released etc. Such details are not available on IMDB but can be obtained from sited like Opus data by paying.
- We could use other social media like tweets while predicting movie rating.
- We can try to predict exact release date instead of just season.
- We can run LDA on a larger dataset considering more number of keywords for each movie.

REFERENCES

[1] http://www.pythonforbeginners.com/python-on-the-web/web-scraping-with-beautifulsoup

[2] https://www.datascienceplus.com/missing-value-treatment/

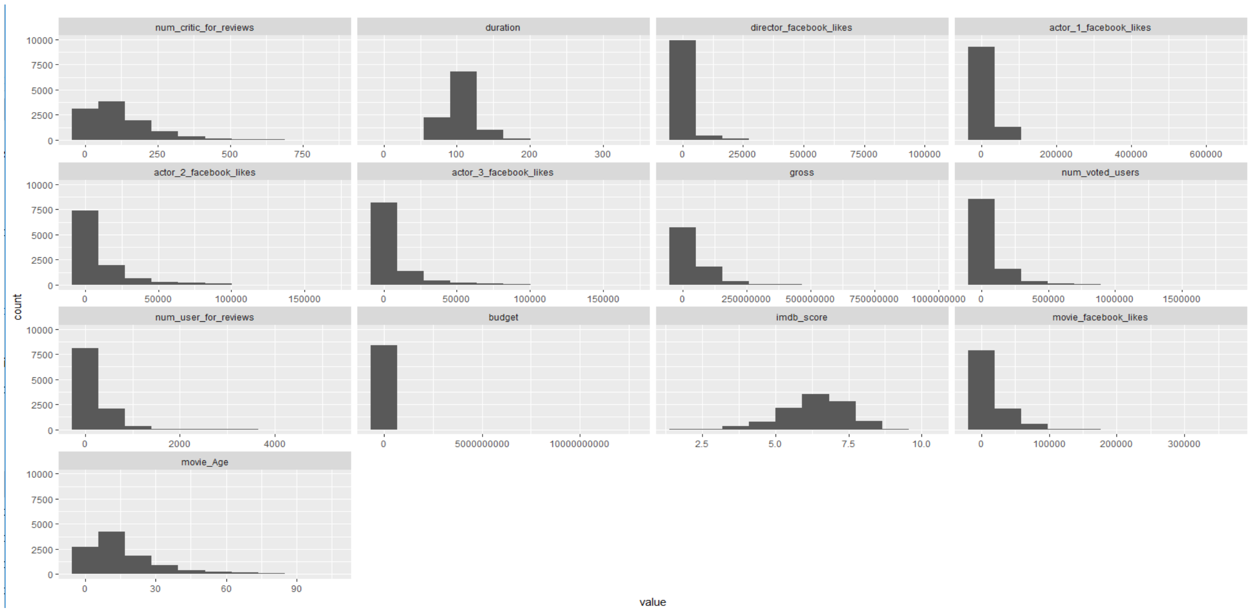
[3]http://stackoverflow.com/questions/21355156/topic-models-cross-validation-with-loglikelihood-or-perplexity/21394092#21394092

[4] https://stats.stackexchange.com/questions/25113/the-input-parameters-for-using-latent-dirichlet-allocation

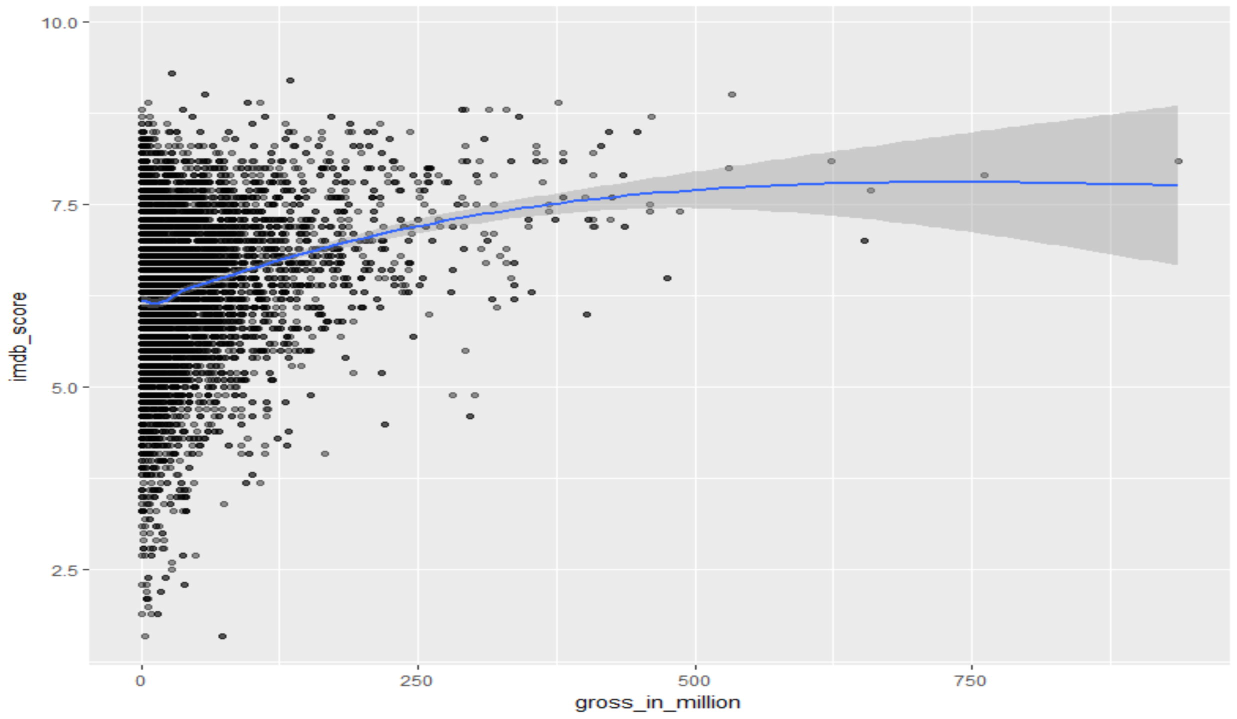
[5] https://eight2late.wordpress.com/2015/09/29/a-gentle-introduction-to-topic-modeling-using-r/

APPENDIX

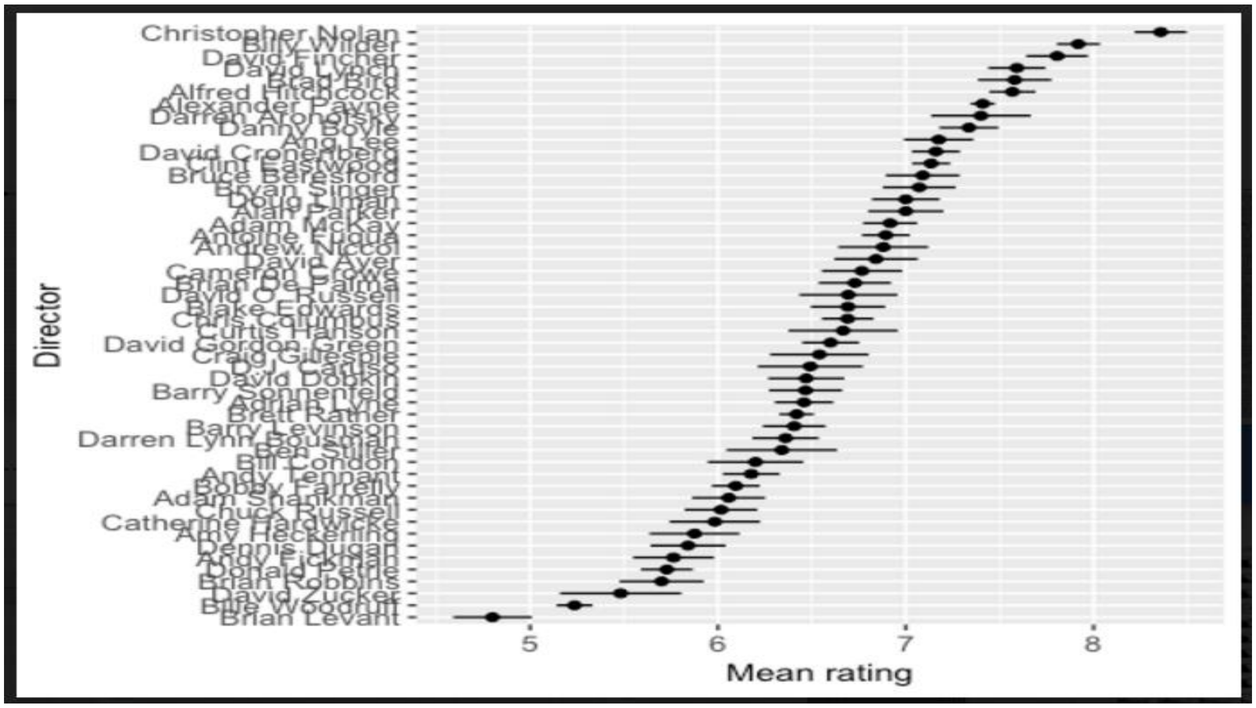
Histograms of numerical variables



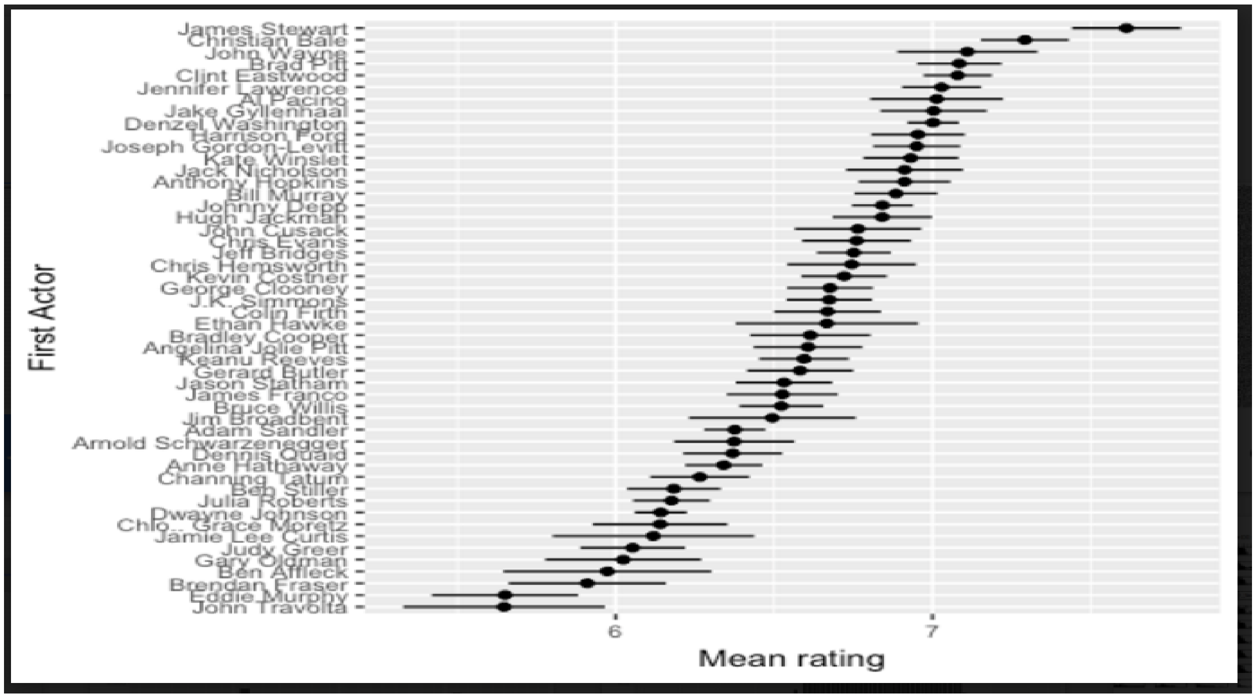
Plot between imdb_score and gross



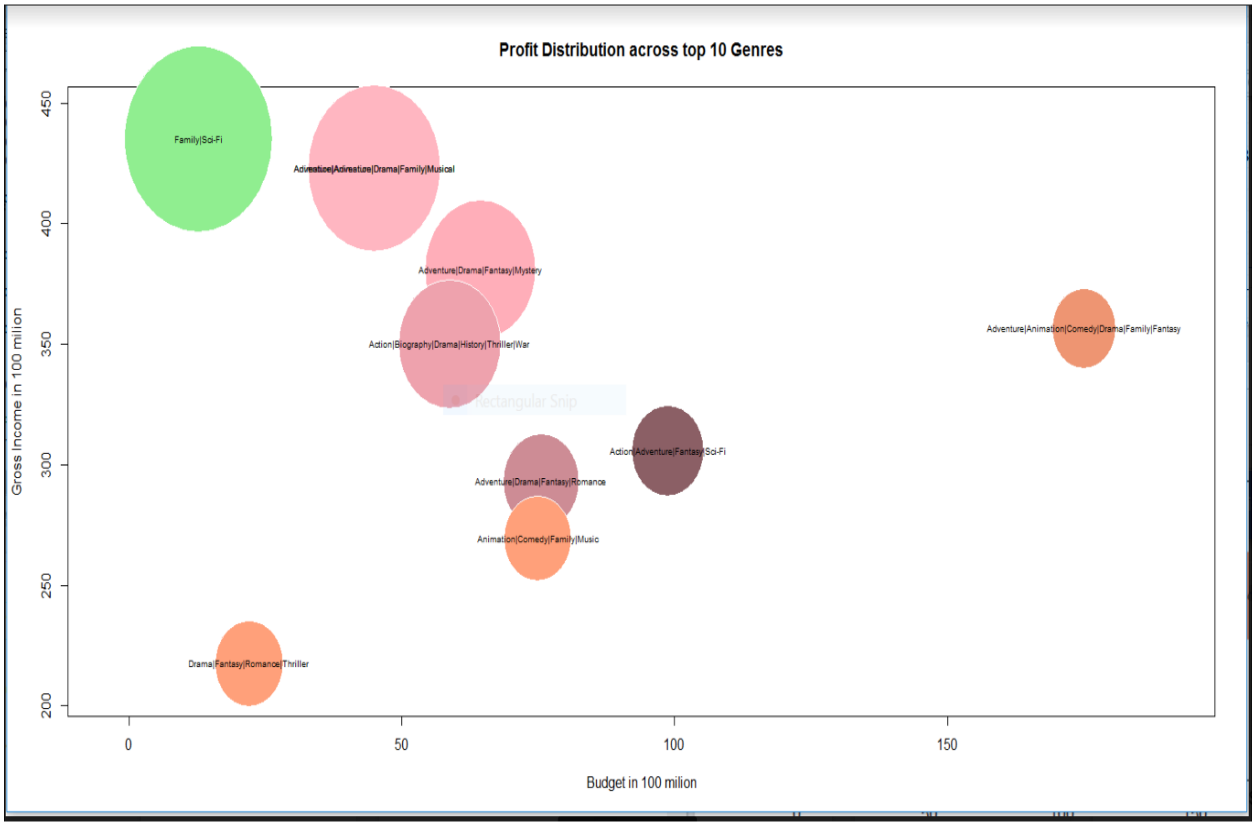
Director Ranking based on mean movie ratings



Acotor1 Ranking based on mean movie ratings



Profit distribution across top 10 Genres


Profit Distribution across top 10 Genres
Gross Income in 100 milion
Budget in 100 milion
450
400
350
300
250
200
0
50
100
150
Family|Sci-Fi
Adventure|Adventure|Drama|Family|Musical
Adventure|Drama|Fantasy|Mystery
Action|Biography|Drama|History|Thriller|War
Adventure|Animation|Comedy|Drama|Family|Fantasy
Action|Adventure|Fantasy|Sci-Fi
Adventure|Drama|Fantasy|Romance
Animation|Comedy|Family|Music
Drama|Fantasy|Romance|Thriller
Rectangular Snip