Prepared By: Parthasarathi Samantaray

Project: Predicting Movies User Ratings with IMDb Attributes
(with special attention to preprocessing of text and categorical attributes)

Data Source:
https://github.com/sundeepblue/movie_rating_prediction/blob/master/movie_metadata.csv

Code Link: https://colab.research.google.com/drive/1-ggHgk8uf1RC7KrUqYU1jMc-vfkRa_I1?usp=sharing

Approach:

- Develop overall understanding about distribution, datatypes, columns, type of preprocessing required
    - Visual inspection using charts and plots
    - Using function head (), describe (), info ()
- Feature Engineering: Develop strategies to handle the Numerical, Categorical and Text/String features in the data
    - Numerical Attributes: Impute and Scale the parameters
    - Categorical Attributes: Apply One-Hot encoding
    - Text/String Attributes- Split the text field and apply One-Hot / word frequency matrix approach
- Parameter Selection
    - Numerical Attributes: Presence of Collinearity
    - Categorical and Text attributes: after the transformations, the dataset became a wide detective. there are more no. of columns than the records. As most of the columns are sparse columns, most the models (including Neural network) failed product any result. So, 20 highly corelated features are selected
- Scalar Transformation and Split the data for Training, Validation and Testing.

Note: *Training data should be used to fit StandardScaler and all the 3 datasets are transformed with the StandardScaler object*

- Model creation & Evaluation: Use Randomized search and Validation details to tune the models. And performances during Training and Testing are used to compare the model performances.

Prepared By: Parthasarathi Samantaray

Data Dictionary:

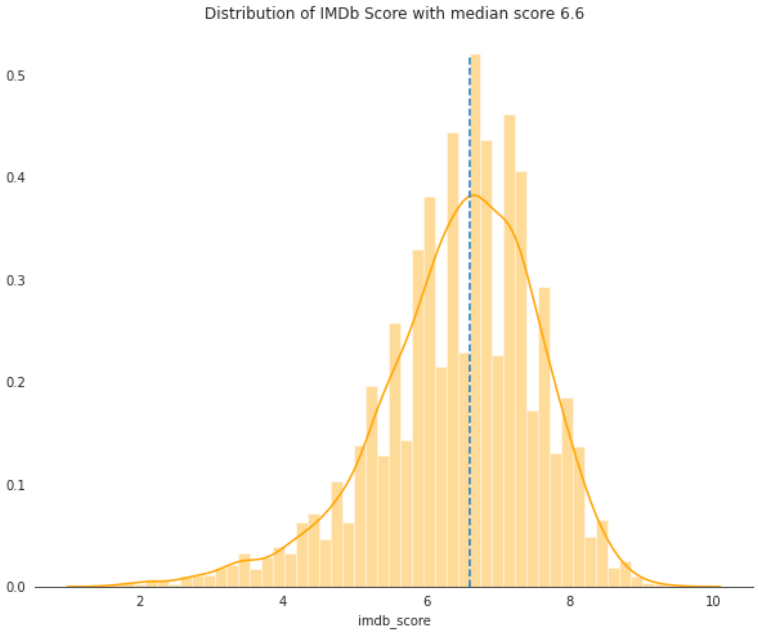
'color': Color /Black and white movie
'director_name': Name of director
'num_critic_for_reviews': Number of critic reviews
'**duration**': Duration of movie in minutes
'director_facebook_likes': Likes in Director Facebook page
'actor_3_facebook_likes': Likes in 3rd actor Facebook page
'actor_2_name': Name of 2nd Actor
'actor_1_facebook_likes': Likes in Director Facebook page
'gross': Gross amount in $
'genres': All Genre of movie separated by '|'
'actor_1_name': Name of actor 1
'movie_title': Movie Title
'num_voted_users': No. of users voted
'cast_total_facebook_likes': Likes in Director Facebook page
'actor_3_name': Name of actor 3
'facenumber_in_poster': No. of faces in movie poster
'plot_keywords': Keyword of plot separated by '|'
'movie_imdb_link': URL of IMDb page of the movie
'num_user_for_reviews': No. of user reviews
'language': Language
'country': Country of Movie
'content_rating': Rating (TV-MA, etc.)
'budget': Budget in $
'title_year': Year of Release
'actor_2_facebook_likes': Likes in Director Facebook page
'imdb_score': IMDB score (between 0 and 10)

'aspect_ratio': Aspect Ratio of movie poster

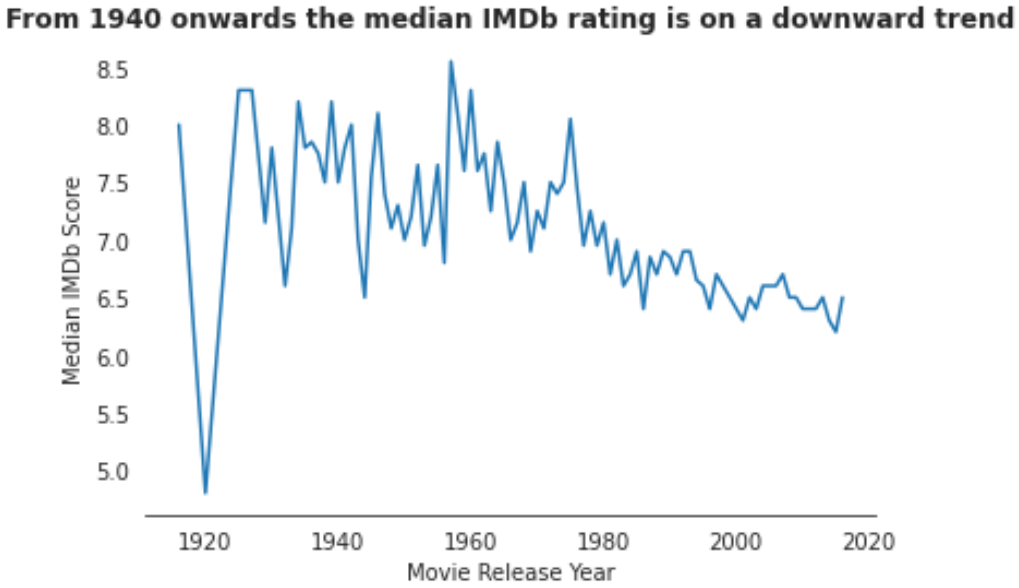
'movie_facebook_likes': Likes in Director Facebook page

Data Exploration for Insight generation

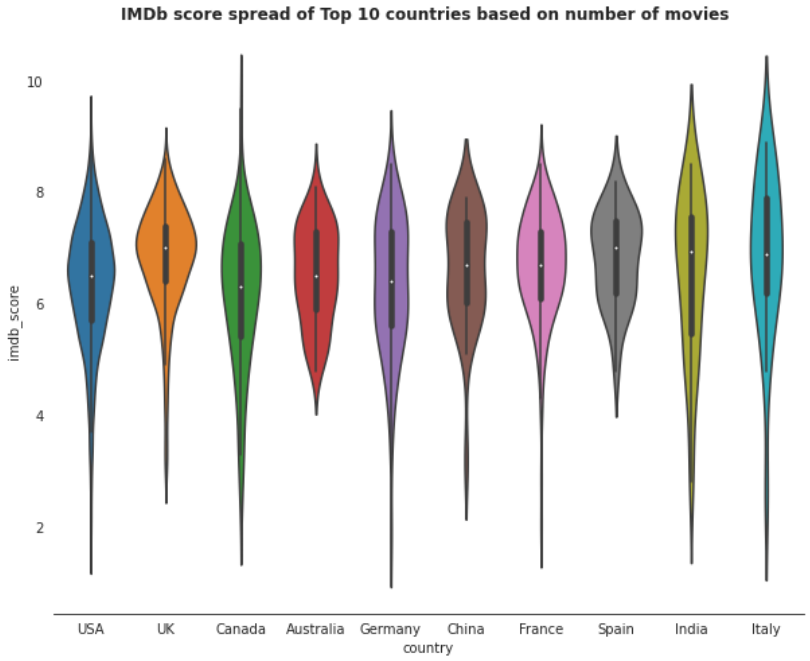
Distribution of IMDb Scores: The user reviews are almost normally spread with left tailed and centered(median) around 6.6



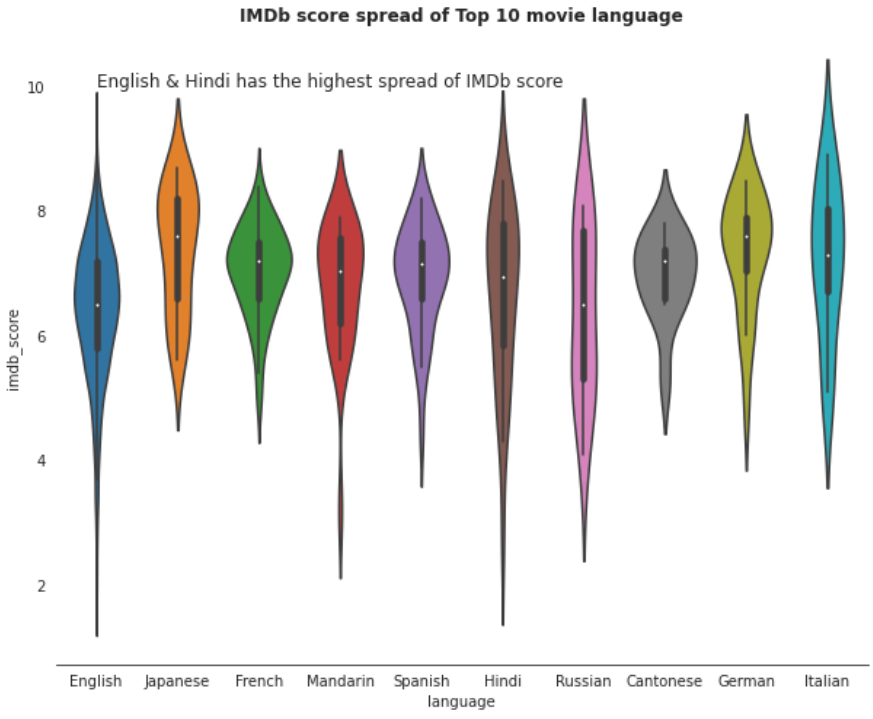
Trend of Median user reviews: The median IMDb scores have a downward trend as time progresses.



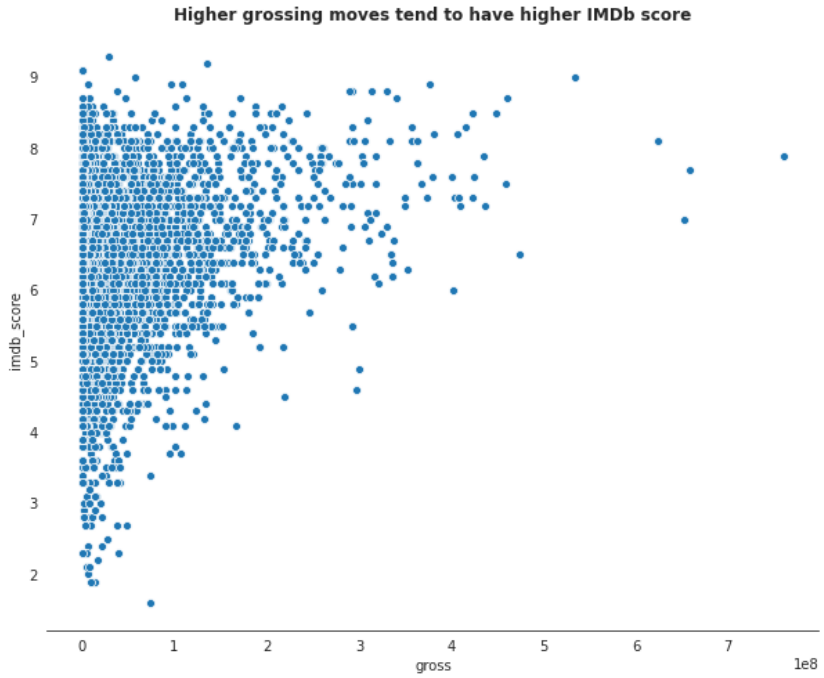
Top 10 countries ($) vs IMDB Score: Highest number of movies are from the US &UK. However, the spread f US movies are much bigger than UK. Few movies form Canada and Italy has got the highest scores closer to 10.



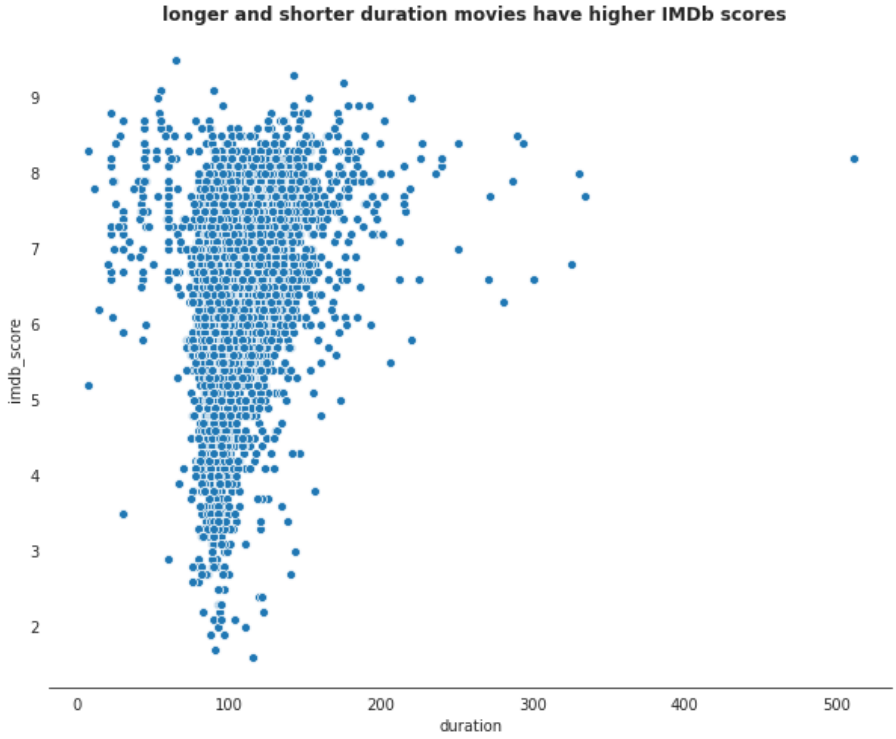
Top 10 Language ($) vs IMDB Score: Most of the movies are in English and Japanese, but average rating of movies in Japanese is much higher rating than movies in English US. Movies in Hindi and English have the highest spread of rating



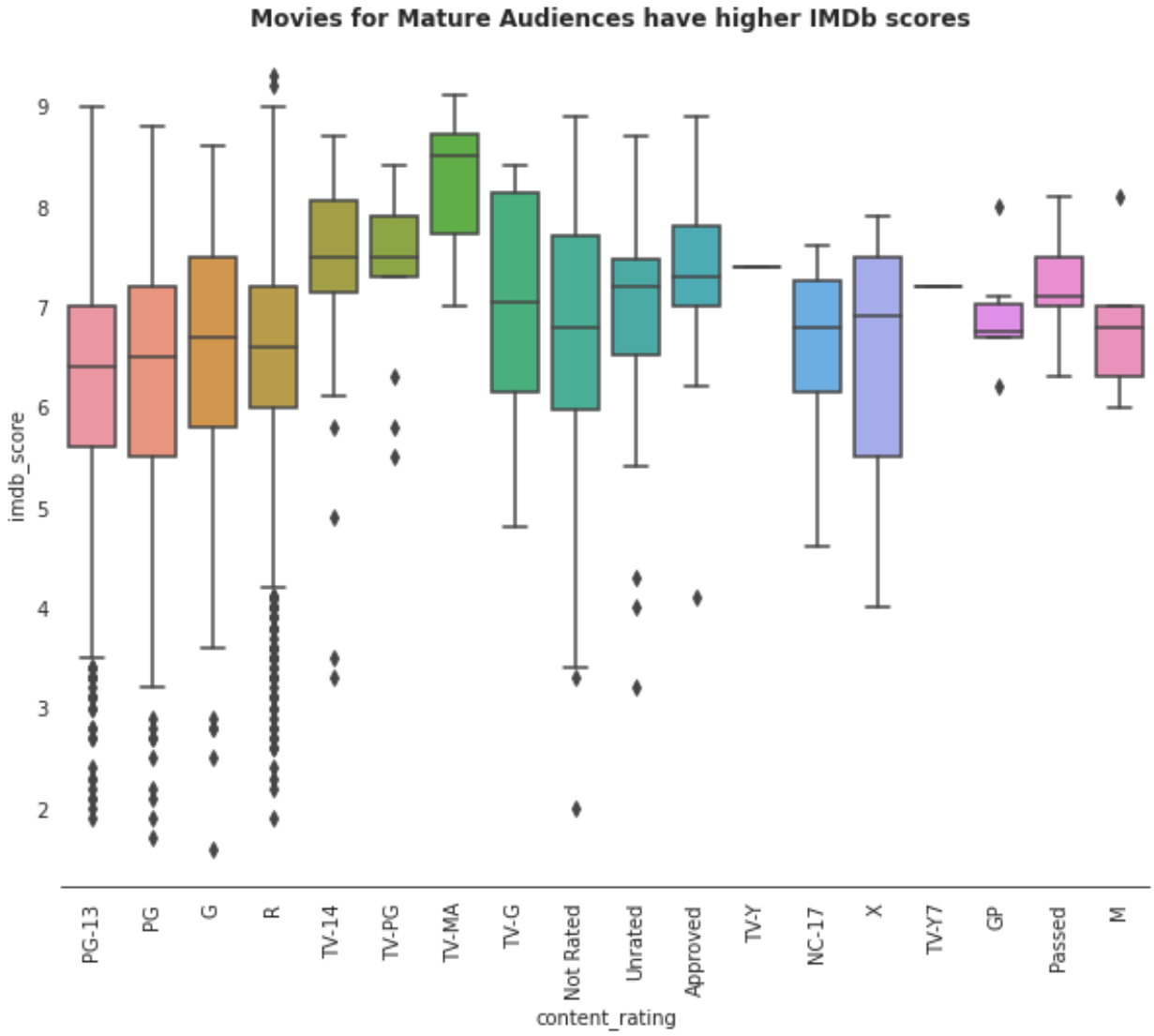
Gross ($) vs IMDB Score: Higher grossed movies tend to have higher rating. As highest rated movies do much better in box-office collections, and it increases the gross amount.



Duration vs IMDb_Score : Majority of movies with smaller or larger duration had higher ratings than average length movies.



Content Rating vs IMDb Scores: Mature contents(TV-MA) has higher rating than other categories of rating



Preprocessing & Feature Engineering

There 3 type of independent features are there in the dataset.

Numerical Features:

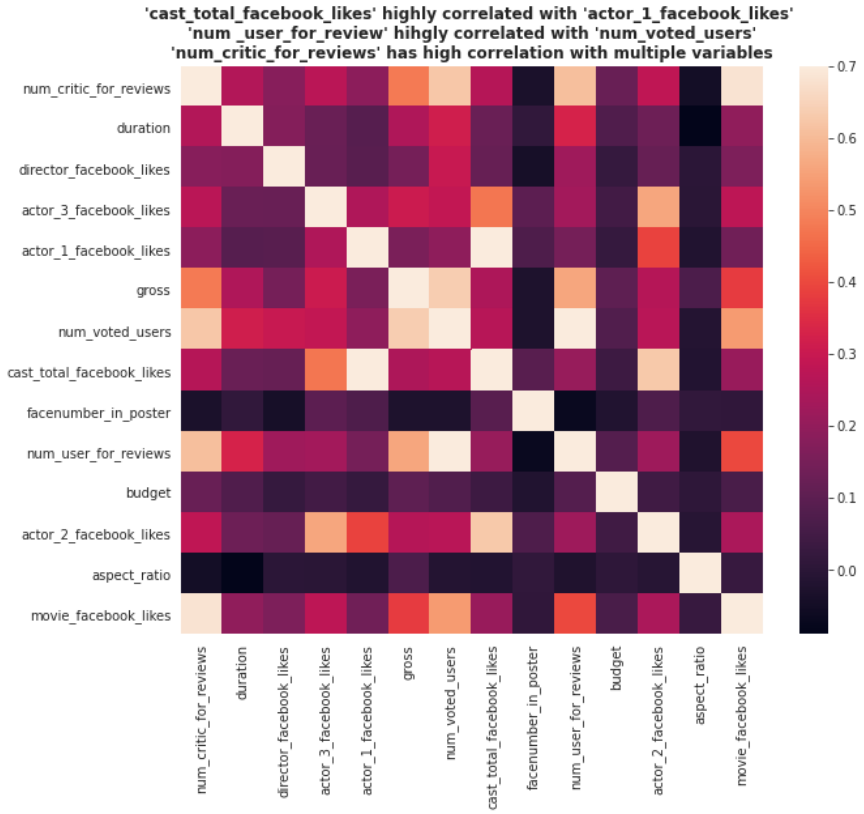
Features -['num_critic_for_reviews', 'duration', 'director_facebook_likes', 'actor_3_facebook_likes', 'actor_1_facebook_likes', 'gross', 'num_voted_users', 'cast_total_facebook_likes', 'facenumber_in_poster', 'num_user_for_reviews', 'budget', 'actor_2_facebook_likes', 'aspect_ratio', 'movie_facebook_likes']

Collinearity Test: Correlation matrix is used find the collinearity among the features. 'cast_total_facebook_likes' is highly correlated with 'actor_1_facebook_likes' and dropped from the dataset.

Similarly, 'num _user_for_review' highly correlated with 'num_voted_users' , 'movie_facebook_likes'. 'num_critics_for_review' is highly correlated with 'num_user_for review'. To accommodate all the dependencies, average of 'num_voted_users' and 'num_user_for_reviews' is used to create a new variable and all the 4 features('num_voted_users', 'num_user_for_reviews', 'num_critic_for_reviews', 'cast_total_facebook_likes') are dropped.

```
dataset['num_user_voted_reviews']= (dataset.num_voted_users+dataset.num_user_for_reviews)/2
numeric_features.append("num_user_voted_reviews")

removelist=['num_voted_users','num_user_for_reviews','num_critic_for_reviews','cast_total_facebook_likes']
dataset.drop(removelist, axis=1, inplace=True)
```



The significant correlations are shown below:

	num_critic_for_reviews	num_voted_users
num_critic_for_reviews	1.000000	0.624943
duration	NaN	NaN
director_facebook_likes	NaN	NaN
actor_3_facebook_likes	NaN	NaN
actor_1_facebook_likes	NaN	NaN
gross	NaN	0.637271
num_voted_users	0.624943	1.000000
cast_total_facebook_likes	NaN	NaN
facenumber_in_poster	NaN	NaN
num_user_for_reviews	0.609387	0.798406
budget	NaN	NaN
actor_2_facebook_likes	NaN	NaN
aspect_ratio	NaN	NaN
movie_facebook_likes	0.683176	NaN

	num_user_for_reviews	movie_facebook_likes
num_critic_for_reviews	0.609387	0.683176
duration	NaN	NaN
director_facebook_likes	NaN	NaN
actor_3_facebook_likes	NaN	NaN
actor_1_facebook_likes	NaN	NaN
gross	NaN	NaN
num_voted_users	0.798406	NaN
cast_total_facebook_likes	NaN	NaN
facenumber_in_poster	NaN	NaN
num_user_for_reviews	1.000000	NaN
budget	NaN	NaN
actor_2_facebook_likes	NaN	NaN
aspect_ratio	NaN	NaN
movie_facebook_likes	NaN	1.000000

Categorical Features:

Features - ['color', 'director_name', 'actor_2_name', 'actor_1_name', 'actor_3_name', 'language', 'country', 'content_rating', 'title_year']

One-Hot encoding is used for all the categorical features including separate column for nan(i.e. absence of data)

String Features:

Features - ['movie_title', 'plot_keywords', 'genres']

For Genres and Plot keywords, each field had multiple values separated by '|' . for i.e. `Action|Adventure|Fantasy|Sci-Fi` or `avatar|future|marine|native|paraplegic` . the texts are split and a word matrix is used to capture the presence of the word or category.

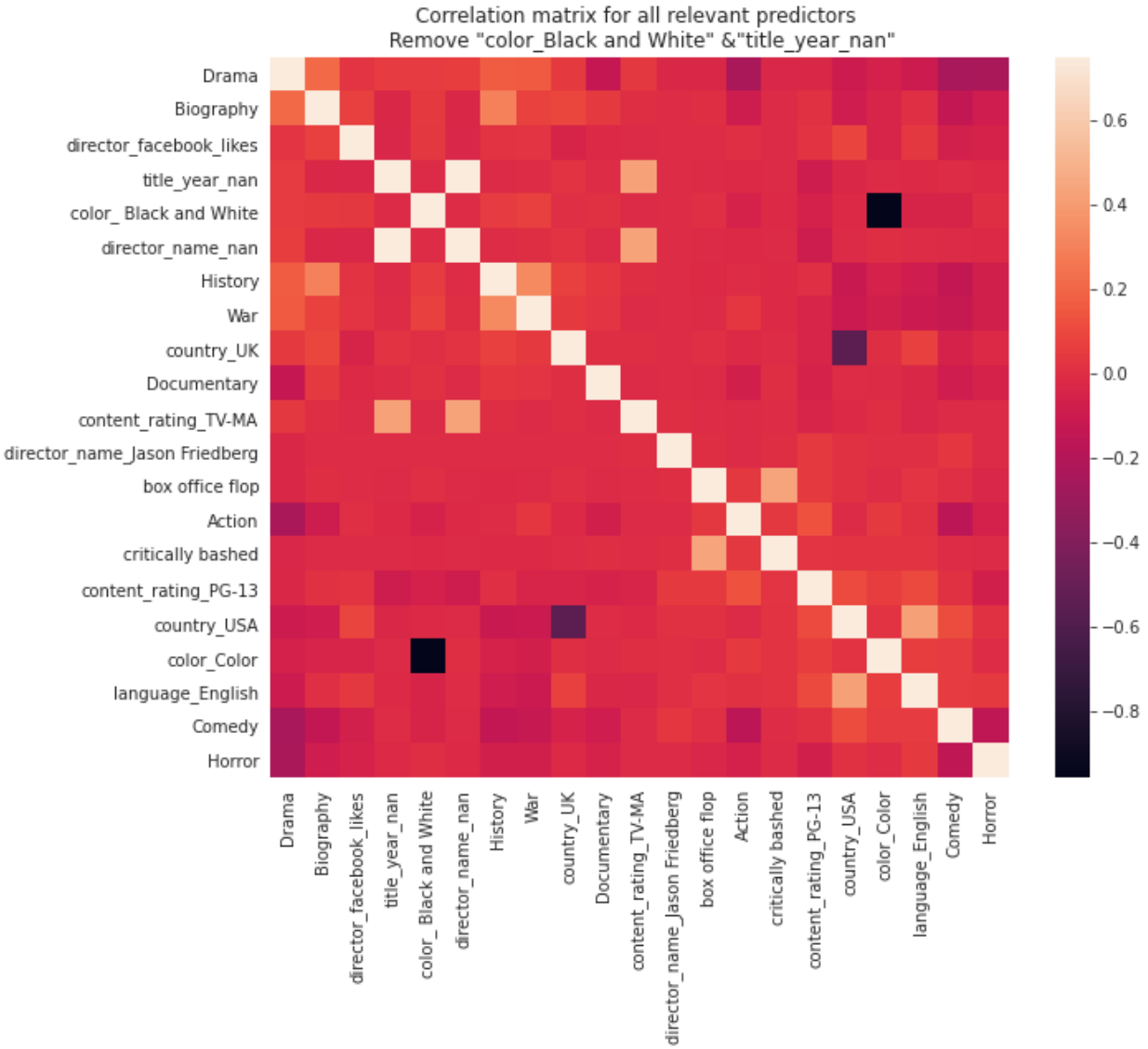
Movie titles had a colon ":" to capture the film series. `i.e. Avengers: Age of Ultron.` The text is split at ':' and only the initial part is used for the movie. Then the above method of breaking the text into a matrix of all words, and presence 1 denote the presence of the word is used.

Failed 1st Approach of taking all the features as Input:

All the above parameters (categorial, text and numerical) are used to create a regressor neural network. However, they were failed create any substantial model.

2nd Approach of Categorical feature selection:

Top 20 features with high Pearson's correlation are selected and features high collinearity are dropped. The correlation map of top features as follows:



Two features `"color_Black and White" &"title_year_nan"` are dropped from the dataset.

Final Feature list for Model input: 18 categorical variables and the numerical variables.

Train, Validation & Test split:

25% of data is reserved for test and 25% of the remaining 75% is used during model finetuning/validation. Rest are used for Training purpose.

Scaling of Numerical Variables:

Training data is used to fit the Standard Scaler and it is used to transform the rest. As this ensures the relative of test variables are maintained with respect to the training. Standard Scaler also maintains the variance of input data.

```
scaler=StandardScaler()
X_train[numeric_features]=scaler.fit_transform(X_train[numeric_features])
X_test[numeric_features]=scaler.transform(X_test[numeric_features])
X_valid[numeric_features]=scaler.transform(X_valid[numeric_features])
```

Neural Network (with Randomized search to finetune the hyperparameters)

The Sequential Neural Network with 'ReLU' activation function was failed to create any meaning full model.

Random Forest Regressor (with Randomized search to find optimal Model)

The model behavior of random forest is mentioned in the model comparison table and the algorithm was overfitting the training data.

Gradient Boosting (GB)

A simple GB with 2 max_depth has performed the best. The R2 of testing was very close to the R2 of training.

XGBoost

The XGboost , perfrmed relatively better than Random forest, but training R2 was little higher than the testing R2.

Model Comparison:

- RandomForest Training R2: 0.93| Testing R2: 0.56
- GradientBoosing Training R2: 0.58| Testing R2: 0.56
- XGBoost Training R2: 0.61| Testing R2: 0.56

Decision: All the 3 models have similar test R2, but for the simple Gradient boosting model, the training R2 is much closer to the testing R2 and not over or under fitting.