Movie Rating Prediction Project Report

Paul He

Abstract

This project is based on a movie rating dataset from github. The main goal is two-fold: Reveal some intuitive understanding by EDA, and build several machine learning models to discover a proper way to predict the IMDB rating of a movie.

Data Preprocessing

1. First of all, there are many missing values in the original dataset:

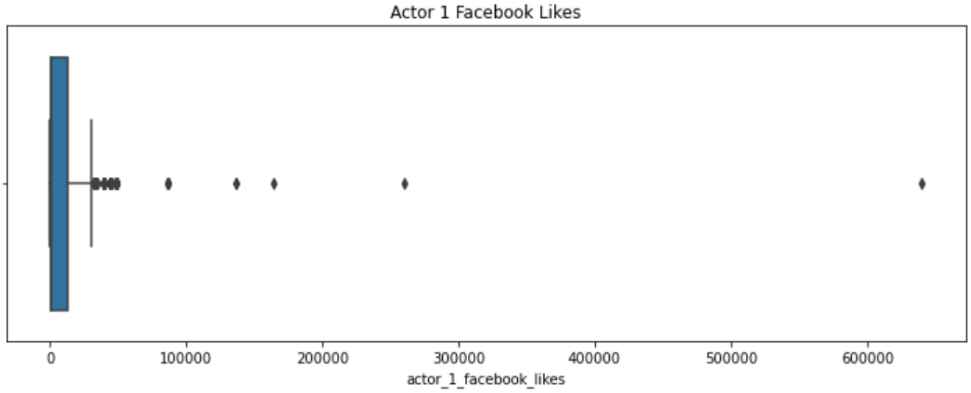
```
color                      19
director_name             104
num_critic_for_reviews     50
duration                   15
director_facebook_likes   104
actor_3_facebook_likes     23
actor_2_name               13
actor_1_facebook_likes      7
gross                     884
genres                      0
actor_1_name                7
movie_title                 0
num_voted_users             0
cast_total_facebook_likes   0
actor_3_name               23
facenumber_in_poster       13
plot_keywords             153
movie_imdb_link             0
num_user_for_reviews       21
language                   12
country                     5
content_rating            303
budget                    492
title_year                108
actor_2_facebook_likes     13
imdb_score                  0
aspect_ratio              329
movie_facebook_likes        0
```

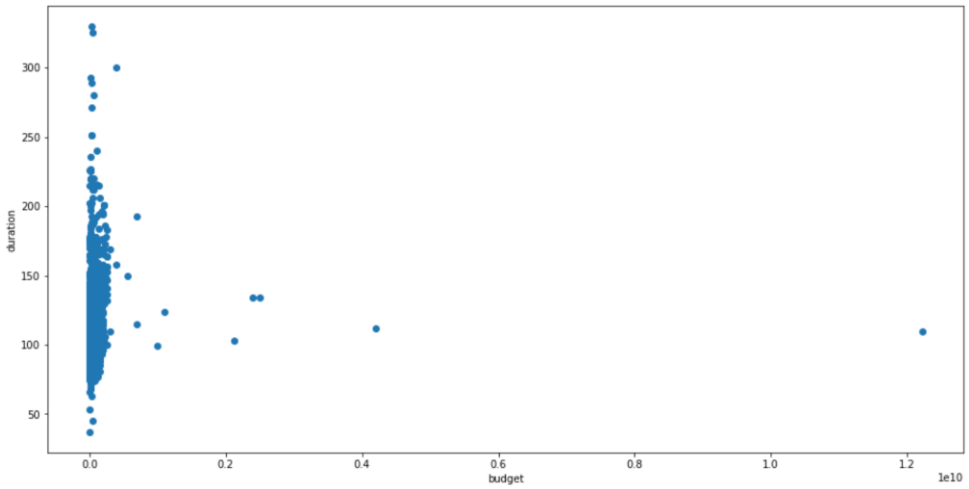
It may take quite some efforts to scrape missing data from the internet and integrate it with existing one, until now I choose to drop all instances with missing value to facilitate the whole progress.

2. After several data cleaning process, including dropping duplicates and trivial variables in this project, correcting data type, transforming some original features into a more operatable way (e.g. genres), the new dataset has 3723 instances, 24 features, and one label ("imdb_score") and looks in good shape.

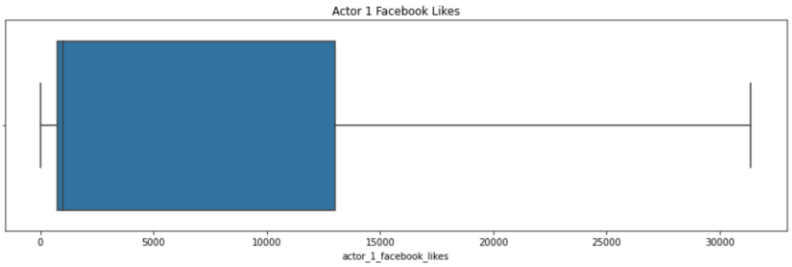
	actor_1_facebook_likes	actor_1_name	actor_2_facebook_likes	actor_2_name	actor_3_facebook_likes	actor_3_name	aspect_ratio	budget	cast_total_facebook_likes	color
0	1000.0	CCH Pounder	936.0	Joel David Moore	855.0	Wes Studi	1.78	237000000.0	4834	Color
1	40000.0	Johnny Depp	5000.0	Orlando Bloom	1000.0	Jack Davenport	2.35	300000000.0	48350	Color

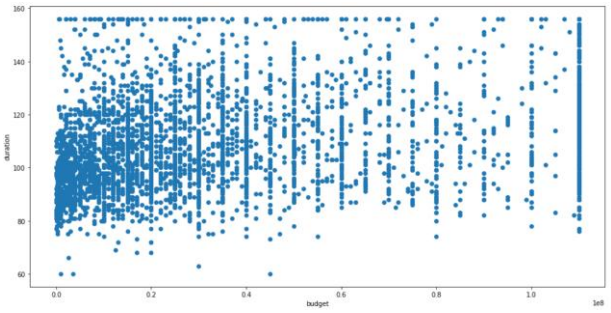
3. As doubted, this dataset has outliers in many features. For example:





I implement IQR quantile method to replace those outliers with the IQR cap. Below are the boxplots after the transformation:

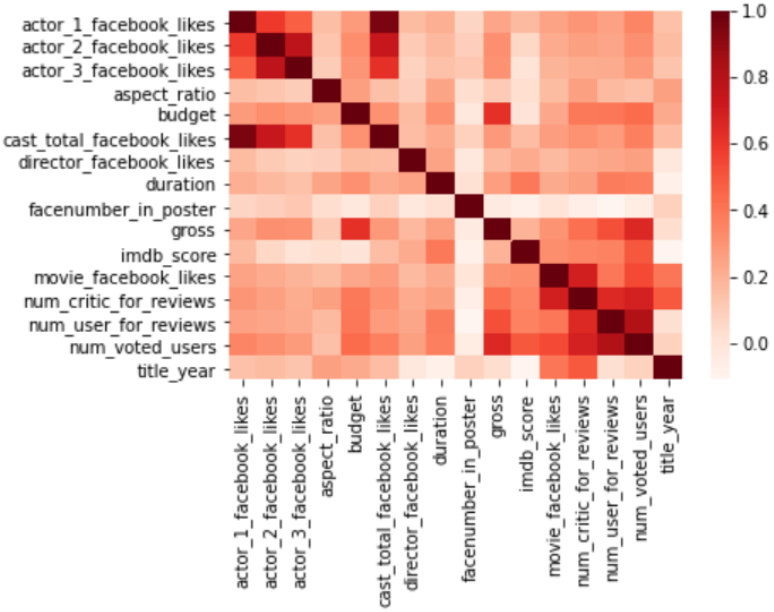




Exploratory Data Analysis

1. Correlation

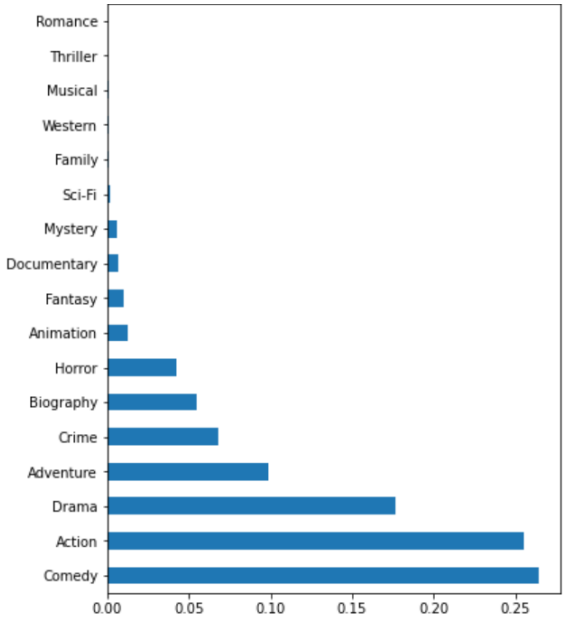
According to the heatmap of variable correlations, some variables have high collinearity (e.g., actor_1_facebook_likes and cast_total_facebook_likes):

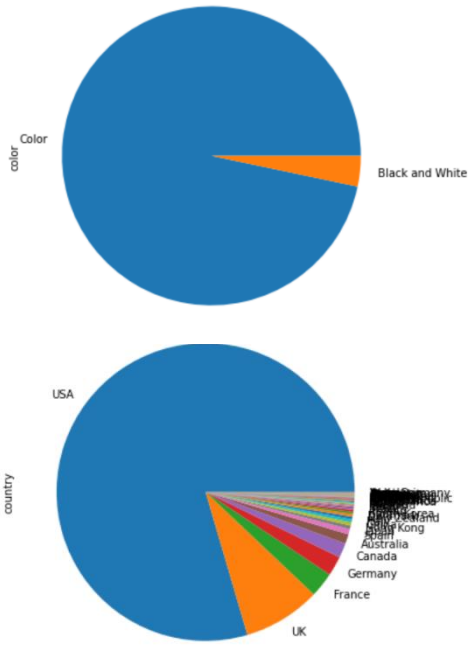


However, since the primary goal of this project is to predict, not interpret the coefficients which are severely influenced by multicollinearity, it is okay to leave those variables with high correlation.

I also find some interesting patterns from the data. Below are some examples.

2. Univariate Analysis

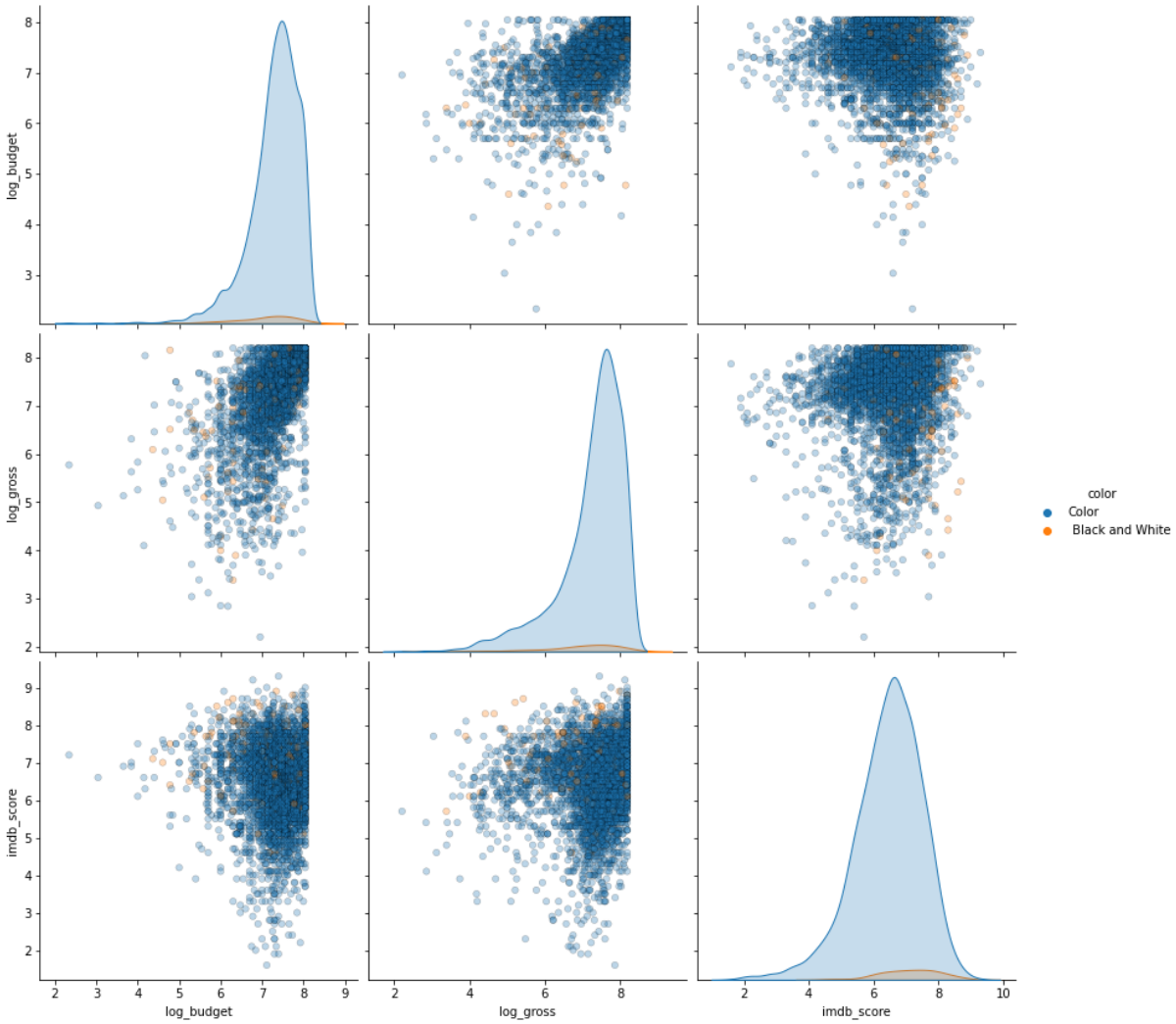




The barplot on the left side shows the ratio of different genres occurring in the dataset. It implies that the hottest categories are comedy, action, and drama. On the other hand, the two pie charts on the right side mainly support some common senses, like most movies are in color and the most productive country is USA. However, they also reveal some interesting findings, e.g., the most productive countries in filming are all in the western world.

3. Bivariate Analysis

By putting two variables together, there are more insights between those pairs. As I zoom in the relationships among budget, gross, and IMDB rating, categorized by color or not, there variables are more or less related. (The values of budget and gross are represented in logarithmic form due to the large number)



4. Variable Transformation

In order to prevent the future modeling being dominated by certain large-scale variables, numerical variables are standardized with the mean of 0 and standard deviation of 1. As for the categorical

variables, I choose to perform categorical encoding instead of one-hot encoding, since some variables have thousands of unique values, which make one-hot encoding impossible.

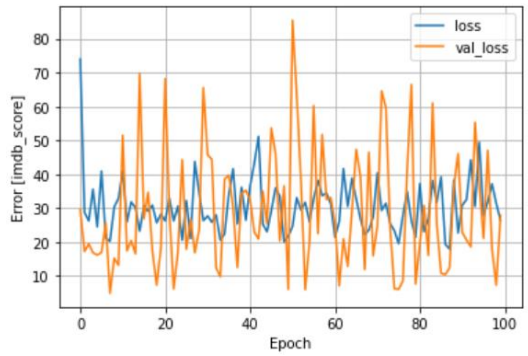
Modeling

The dataset is split into training set and testing set with 80/20 ratio by sampling, and the cross validation rate of all model is set to 0.2.

By implementing the sequential function from Keras, it is easy to build and compile several machine learning models and train them.

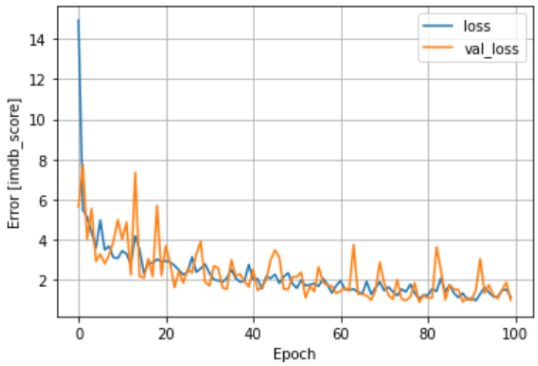
Parameter epoch is set to 100 without early stop. Performance of the models are measured by mean absolute error.

1. Multivariate Linear Model



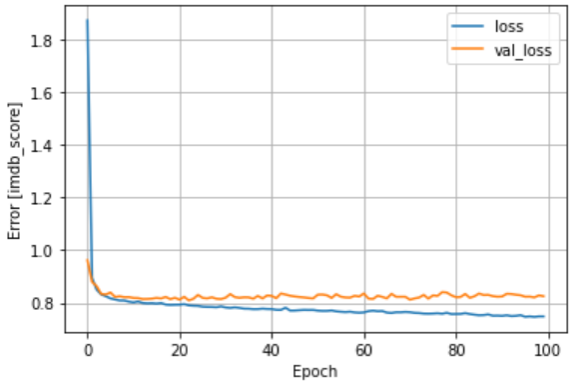
The performance of linear model is quite poor, probably due to its high sensitivity to large coefficients. Besides, the categorical variables are encoding in an ordinal way, which may also contribute to the bad result.

2. DNN Model with Rectified Linear Unit Activation Function and Adam Algorithm Optimizer



The result is much more ideal compared to the former one. This may imply that neural network models can indeed learn by themselves and produce the output that is not limited to the input provided.

3. DNN Model Applying Hyperbolic Tangent Activation Function and Adamax Algorithm Optimizer



This model's performance is slightly inferior to the second one. Since other variables are controlled, the reason might be the application of activation function and optimizer.

4. Performance Evaluation

All the models are tested by feeding the testing set.

	Mean absolute error (imdb_score)
linear_model	29.511837
dnn_model_1	0.966076
dnn_model_2	0.846006

Conclusion

According to the result, linear model has a quite poor performance, while both deep neural network models work relatively better. In that case, DNN should be the more useful algorithm tool to deploy into further prediction.

However, this is a quite naïve approach to solve this regression problem. At far as I can see, there are some aspects could be improved, e.g., fill the missing value in the original dataset, try principal component analysis to ease the dimension problem, and use grid-search to tune the parameter of models.