**Predicting Movie Scores**

**IT6773 – Practical Data Analytics**

**Fall 2020**

### **Group3**

### Caitlin McOsker

### Edward Moses

### Jason Smith

### Fikko Soenanta

**Abstract**

The purpose of this project was to take a large data set with various movie data and apply machine learning techniques learned throughout the Fall 2020 semester to perform various data analytics tasks. The dataset contained ~10k entries that contained movie metrics and the goal was to apply regression models to predict the score of the movies based on those metrics.

**Introduction**

The data for this project was obtained from the following links:

Movies Dataset: <https://www.kaggle.com/michau96/are-popular-movies-good?select=movies.csv>

Budget Dataset: <https://www.kaggle.com/deepak525/investigate-tmdb-movie-dataset?select=tmdb_movies_data.csv>

As noted above, we are looking to utilize the metrics in this data to predict the expected score of how good a movie is on a scale of 1-10. While there was a lot of entries in the columns provided, there were several troublesome attributes that we did not deem important for predicting the score:

* Movie Title
* Year Movie was released
* Total Votes
* Runtime
* Movie Description
* Director

These columns were removed from the data prior to any preprocessing and training/testing was completed on the data. We then added in the Budget column from the budget dataset included above

The models we chose to use for this project were regression algorithms fit for determining the score:

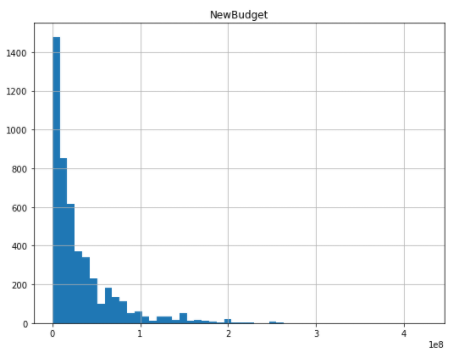
* Linear Regression
* Logistic Regression
* Support Vector Machine
* Decision Tree

**Preliminary Analysis**

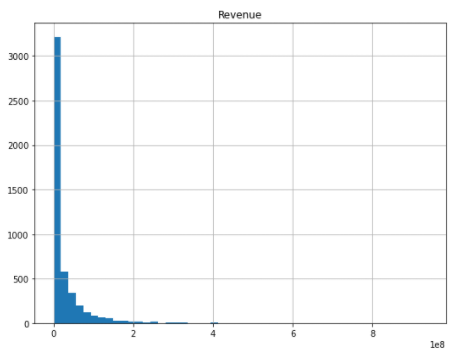
The final data file in the project contained the following data columns.

**Numeric Columns:**

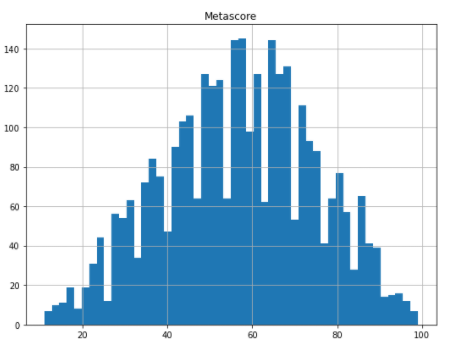
***NewBudget*** – Budget for the film



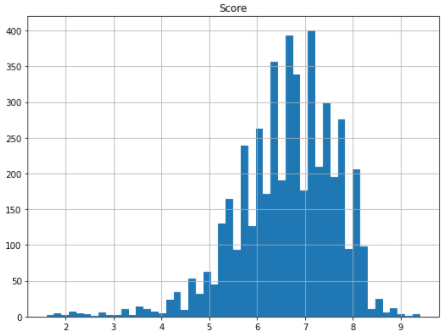
***Revenue*** – Total revenue the film made



***Metascore*** – Score from critics

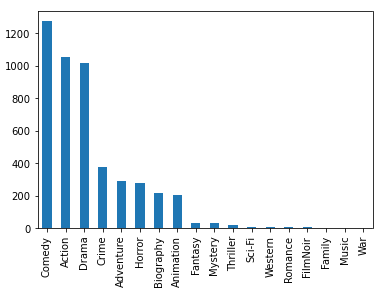


***Score*** [**target**] - Average score of the movie based on reviews



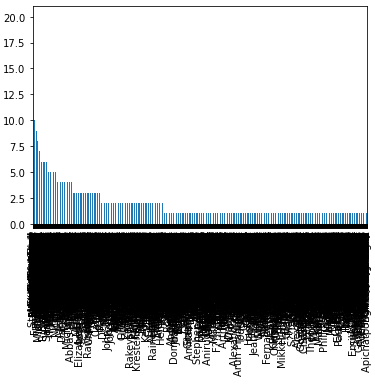
**Class Columns:**

***Genre*** – Movies have multiple genres associated to them and this was the primary genre associated



Originally, we tried to include all of the genres for a given movie, but we ended up going with the first main genre listed for each movie.

***Director*** – This is the name of the director of each movie



This is why we decided to not use “Director,” because we felt like we would not get meaningful results. Most directors only have one movie, with the top one only having 10. To use OneHotEncoder on this data would result in having too many rows and did not seem meaningful.

**Data Preprocessing**

First, we added in our budget data. We did this by comparing our budget dataset to our movie dataset and matching up budgets based on movie title. We then added this column into the movies dataframe.

Then we removed movies that did not have a budget and revenue listed. We ran into this issue because some movies were too old to have a recorded budget and revenue. It was over 5000 records, so we decided to not impute this data. If it had one and not the other, we kept this record and used imputation.

After removing some columns that were not pertinent to our prediction models, the data still needed to be tweaked and preprocessed due to either missing values, classes that need to be encoded, or wide-ranging numeric values that required standardization. The preprocessing steps taken included the following:

***Imputation*** -Required to fill in missing numeric and class values in the following columns:

* Numeric
  + NewBudget
  + Revenue
  + Metascore
* Class
  + Genre1

***One Hot Encoding*** - Required to encode class attributes for genre into numerical values as part of the model.

***Standardization*** - Required to standardize numeric columns due to value ranges that were inconsistent across attributes. We did this for the three numeric attributes. We also did this for the target data for our three regression models: random forest, linear, and SVR.

For the logistic regression model, we also did some preprocessing to convert the target to binary data.

**Modeling and Results**

For all of our models, we used a cross-validation number of 5.

**Linear Regression**:

Linear Reg Train R2: 0.3512681644674207

Linear Reg CV MSE: 0.6547177447302073

Linear Reg CV R2: 0.3451619869941007

R2 had a rather low (34%) score, due to a large variation in the data. We concluded that this was not a good model to fit our data with.

**Logistic Regression**:

Training Accuracy: 0.737410071942446

Cross-Validation Accuracy: 0.7354748808719668

Cross-Validation F1: 0.6640899508081519

Training accuracy and Cross-Validation accuracy resulted in similar values. Cross-Validation F1 was slightly lower. These tests do not appear to have significant overfitting.

After regularization, we got the following MSE and R2 scores:

Best Parameter: {'C': 5}

Best Score: -0.08100611669492416

Logistic Reg Train R2: 0.737410071942446

Logistic Reg Train MSE: 0.9349372592538624

Which were much better results than with linear regression. The R2 score was likely better because there was less variation due to the L2 regularization.

**SVR:**

{'C': 1, 'gamma': 0.1, 'kernel': 'rbf'}

0.3621112101234983

SVM MSE: 0.6414267441219439

SVM R2: 0.3583495553993789

This model did okay, though not as well as our logistic regression model. It had a similar R2 score to linear regression, likely again to the variation.

**Random Forest**:

Random Forest MSE: 0.7546849822803375

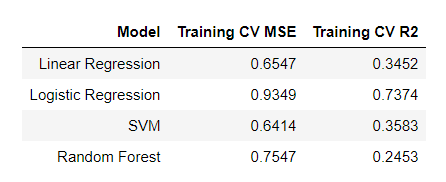
Random Forest R2: 0.24530149916762473

The Random Forest model was our worst performing model. It produced an abnormally low R2 value, despite a decent MSE score.

**Conclusion**

Of all of our models, our best models were logistic regression and SVR. Logistic looked at the target after it had been converted into binary data, with all scores over 7 being encodes as “1” and below 7 as “0.” SVR looked at the target as numeric data. The biggest issue with this model was the low R2 value.

Below is a chart of the MSE and R2 scores for each model:



Based on logistic regression being our best model, it is better to predict a good v bad ranking than to predict the actual score that a movie might receive. That resulted in a better fitting model with better scores.