IMDb Top 1000 Movies List – Predicting Movie Critic Scores

Topic: I have always been one to check movie ratings before watching. It wasn't until I came to college that I realized that some of my new favorite movies had all types of reviews. The IMDb Top 1000 movies list is a ranking that many will look to for movie ideas on a regular basis. On IMDb, films are given both an "IMDb Rating" and a "Meta Score". The former comes as a result of one-time votes submitted by IMDb users on a 1–10 score. The latter is a score given by movie critics on a 0–100 scale. This particular list ranks the movies by their IMDb rating. With about a hundred years' worth of movies to consider, I was curious as to how movies received their IMDb rating. Furthermore, I wanted to see what movie feature has the greatest importance for predicting a movie's rating.

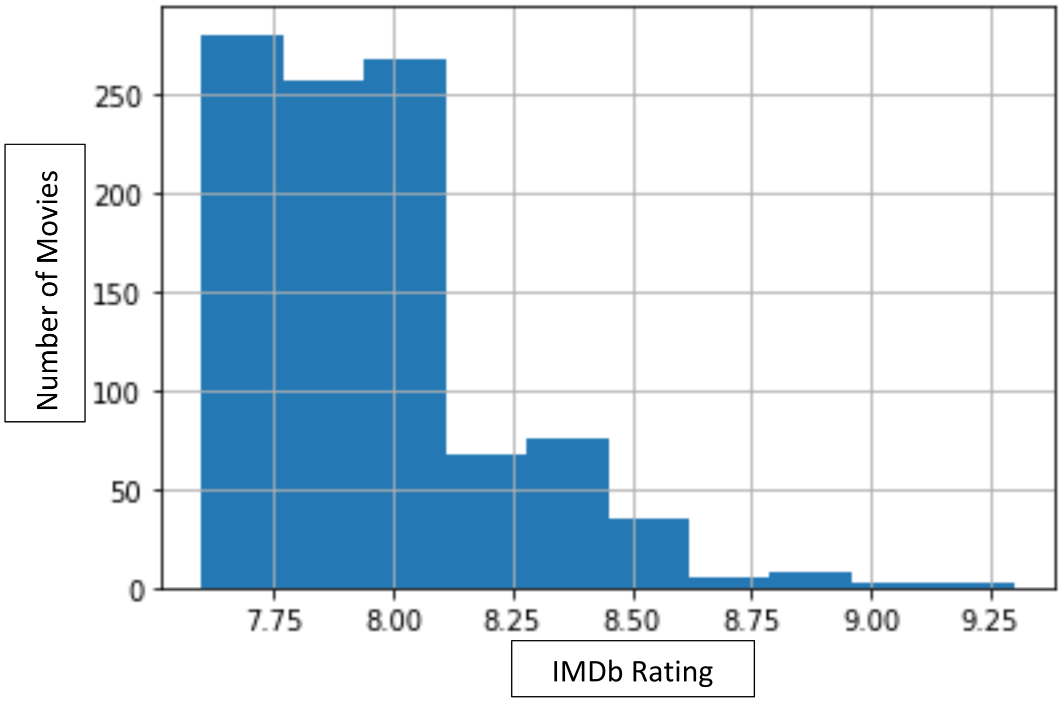
Dataset: The dataset contains each movie's year of release, Meta Score, IMDb Score, Gross Revenue, and runtime. In terms of missing values, the dataset was missing 157 Meta Scores, most of which are for the really old movies. In order to quickly check over the data, I printed descriptive statistics for each column in the dataset. I counted the number of movies from each year, took the average Meta Score and IMDb Rating from each year. Click Here to view the data source. Click Here to view the source code.

Guiding Questions:

1. To what degree can my model predict how critics rate a movie?
2. What movie features have the most importance for how a movie is rated?

Number of Movies by Year

In order to predict what kind of IMDb rating a movie was going to have I first had to get a description of the IMDb rating column in the dataset. Since a specific movie rating would be too difficult to predict with the information at hand, I broke the data into four evenly distributed quartiles: lower quartile, lower middle quartile, upper middle quartile and upper quartile.



Breaking Up the Genre Feature

After I applied new columns to the data set that broke Critic Score(IMDb Rating) and Audience Score (Meta Score) into four categories, I ran into a problem in that most of the movies on the list fell under multiple different genres. As a solution, I divided all the genres into a list in which I could add to the dataset and then create dummy variables from this list.

Attaching Movie Certificate and runtime Dummies

Whether a movie is made for kids or adults or is long or short, neither should have an influence on how the critics review a movie, I felt the need to turn these features into dummies just in case the movie critics had a tendency to favor longer movies, or PG, PG-13, or R rated movies over one another. Once the "certificate" column was converted into dummies, I concatenated it with the main data frame. Before I could attach the runtime however, I had to remove the commas so the model could process the values as integers.

Director as a Feature within the Model

At first, I thought that if I one-hot encoded the director, I could improve my accuracy score based off those directors that made the list multiple times and likely received great critic scores. Unfortunately, adding a new column for every director on the list was not only making the data frame too expansive to visualize on my computer but it actually made the accuracy score worse. As a s result, I removed the director feature from the model.

Error in IMDb Dataset Found

As I tried to train and test the data, I kept getting an error message stating that the vale "PG" could not converted into an integer. This stumped me because I knew I had already converted all the movie certificates to dummy variables. So I searched the data and found a random entry in Released Year as "PG". In an attempt to avoid any more errors like this, I dropped all unnecessary features from the data frame before training and testing.

Baseline Accuracy = 27.05%.

Bagging Mode Accuracy Score = 35.11%

Gradient Boosting Accuracy Score = 32.89%

Logistic Regression Accuracy Score = 24.44%

Logistic Regression with Cross Validation Score = 32.44%

Random Forest Classifier Accuracy Score = 39.56%

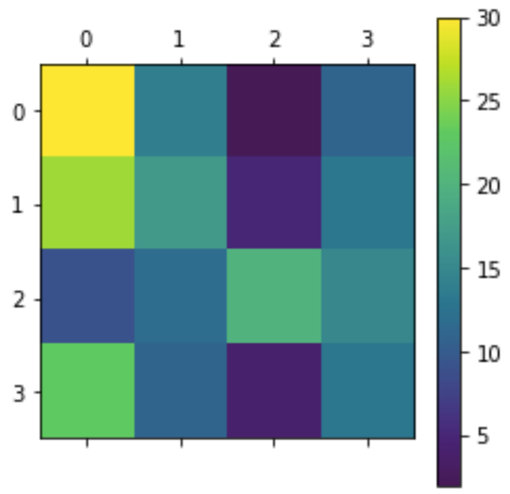
RF Classifier with Randomized Search CV Accuracy Score = 36.89

Choosing the Best Model

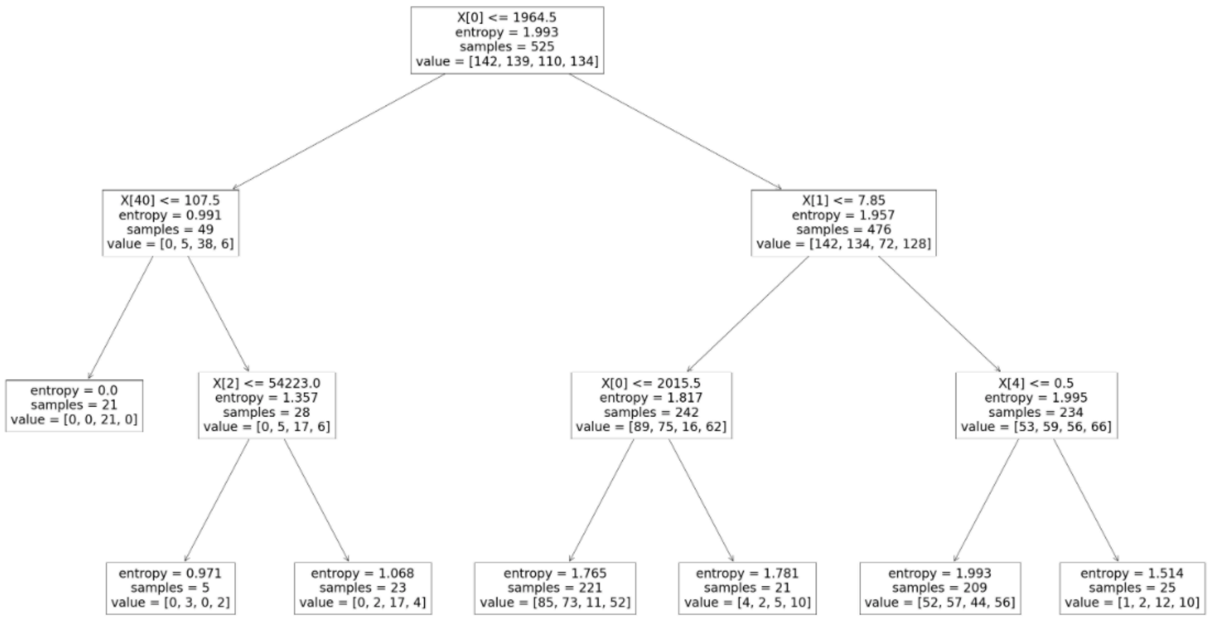
Of the models I ran, the random forest classifier had the highest accuracy score. However, when I ran the cross validation for the RF classifier, the accuracy score decreased almost 3%. I did notice that if increased the values of the parameters that were being searched, the accuracy would improve but the model would take longer to finish running.

Confusion Matrix

In this chart, the yellow corner represents movies that were accurately predicted with this model.



Decision Tree



Conclusion

According to the model, the best way to get an idea for how great a movie is, is to watch the movie itself. With such a low accuracy score, it is clear that one cannot accurately predict a movie's IMDb rating despite knowing its key features. In a future analysis, I would break the IMDb Rating into halves instead of quartiles in order to get a higher accuracy score.