[[1]](#footnote-1)

Predicting movie rating

based on IMDb data

Neil Matin (nm735)  
Department of Computational Engineering  
Mississippi State University  
nm735@msstate.edu

*Abstract*— The Internet Movie Database (IMDb) is an online database that has a weighted mean-rating system for rating movies. Knowing the possible rating of a movie based on various factors like genre(s), actor(s), director(s), etc. would help producers determine the cost benefit of a movie before its substantiation. This paper talks about Film Analytics, a project aimed at predicting IMDb movie ratings using covariates like genre(s), actor(s), and director(s). The implemented predictor system uses and compares two regression algorithms—linear regression and multinomial logistic regression—for predicting the movie rating and the predictive improvements, bringing insights about the data. We have also analyzed some trends occurring in the IMDb movie data as well as the box office data from IMDb itself. Incorporating Naive Bayes classification method, the machine learning model also has the ability to include certain plot keywords as features for predictive analysis.

*Index Terms*—Data mining, Machine learning, Multiple Linear regression, Naïve Bayes classification, Ordered Logistic regression, Predictor.

# INTRODUCTION

Every year, hundreds of movies are made only in the United States of America. It is a multi billion-dollar industry where revenues generated contribute to a major part of the economy. A lot of movies also incur heavy losses for the production houses, and hence the possibility of pre-determining the success of a movie could be of considerable importance for the makers. Of course, success is a relative term and there could be more than one aspect to it, so here we focus on determining the rating of a movie and base its success on it. The Internet Movie Database (IMDb) allows users to rate a movie on a scale of 1 to 10, and the totals are converted into a weighted mean-rating that is considered as the rating of the movie. In general, people are more inclined to watch a movie with a higher rating than one with a lower rating, and so we aim to measure success of a movie by its IMDb rating. Knowing the approximate IMDb rating of a movie before its conception could give the producers an idea about how economically viable the movie was going to be.

Numerous studies have tried exploring and developing models which aim at predicting movie ratings. Film Analytics is a project in which we have attempted using the IMDb data

to predict the ratings of movies based on particular features such as actor(s), director(s), genre(s) and plot keyword(s). The paper has been divided into the following parts. Section II talks about the dataset collection and preprocessing. Section III is about feature engineering that is required for our machine learning models. Section IV gives a brief overview about the big data related challenges faced during our work in this project. Section V talks about two different machine learning techniques used to implement our model, namely multiple linear regression and ordered logistic regression, respectively. Section VI talks about different trends observed in the dataset by applying some machine learning models. Observations and results are shown in Section VII. Finally, we conclude with Section VIII.

# Dataset Collection And Preprocessing

Movie data for approximately 1.1 million movies was obtained from OMDb (a free web service to obtain movie information) and stored in a comma-separated values (CSV) file of the order of 500 megabytes(MB). Initially, the data was fetched by making application interface (API) calls to the OMDb web API and was later stored in a relational database. However, the web server restricted us from fetching the data faster. After making a generous donation to the website owner, we now regularly receive the requested movie data in an unparsed text file every month. Since OMDb does not feature retrieval of box-office data, we personally web-scraped IMDb to fetch the same data.

To start with, the text files were parsed into a readable format. Movies with genres such as Adult, Documentary as well as television events disguised as movies corresponding to genres such as Reality TV, Talk Show, etc. were filtered out to remove noise and redundancy. Similarly, made-for-television movies were also filtered out using the Rating column. Movies with their age-related rating (Age Rating, to differentiate from IMDb rating) starting with TV, for example, TV-13, are filtered out. Despite being a huge library of movies, IMDb does not have all the required data for a large chunk of the movies. This makes it impossible to filter out television movies as well as other television services such as pay-per-views.

For the sake of our project, we considered movies based in USA only, since they have a greater proportion of the complete data. Thus, out of a total of 1103922 movies, 71565 movies were taken for our model (training and test data). 76440 actors, 20962 directors and 60376 plot keywords were considered for the training data. 15556 movies with box-office budget data, 4205 movies with box-office gross data and 4201 movies with box-office data were taken into consideration for the training and test data.

# Feature Engineering

The independent variables (or features) considered in our model are actor(s), director(s), genre(s) and plot keyword(s). The IMDb movie rating is the desired outcome or the dependent variable. Since all the independent variables here are categorical, not continuous, we needed to convert the categorical variables into a form that “made sense” to regression analysis. All the independent variables here were categorical; this is also known as an analysis of variance(ANOVA). Basically we needed to numerically represent the categories of a variable, called dummy variables in which data was coded according to a 0 and 1 scheme. As an example, we transform a variable such as Genre, which consists of categorical values such as Action, Drama, etc. into a feature matrix with scores of 1 and 0 that can be analyzed with regression.

As such, before proceeding to “apply” machine learning, we needed to create feature vectors for the independent variables: genre(s), director(s), actor(s), and plot keyword(s). A typical feature vector for actors would be a sparse matrix that would have movie IDs as rows and list of actors (from the movies in the dataset) as columns (dummy variables). If a specific actor was present in a specific movie, their intersection would have value 1; 0 otherwise.

# Big Data Related Challenges

The first challenge we faced was to successfully compress all valuable information of 1.1 million movies into a compressed format. Next, our dummy matrix was a sparse matrix, meaning it mostly had zeroes as its values. To handle memory efficiently and to take advantage of the sparse structure of these matrices, we stored them in a Compressed Row Storage (CSR) format.

Data acquisition was the bottleneck for the longest time; making individual API calls from a machine with limited processing capabilities to a server with frequent overloads. Deciding upon what machine learning algorithms to choose for analysis, taking into consideration the limited processing capabilities and the timeframe of the project.

# Machine Learning Models

Our data is supervised; each movie in the data has its own rating. Thus, each movie is labeled. The obvious approach was to proceed with supervised learning. Supervised learning is further subdivided into two categories, where the dependent variable can be either continuous or categorical. Continuous supervised learning deals with regression analysis while categorical supervised learning deals with statistical classification that can be later further incorporated into a regression model. For our project, we have delved into understanding both the approaches to learn about the predictive improvements. The following are the two machine learning algorithms that were used:

1. Multiple linear regression (linear regression)
2. Ordered logistic regression (classified non-linear regression)

## Multiple Linear Regression Model

We use the following equation for the multiple linear regression model:

***y*** *=* ***β****0 +* ***β****1****x****1 +…+****β****n****x****n*  (1)

where *y* our outcome feature, ***xi*** are our independent vectors and ***βi*** our model coefficients. We estimate the coefficients by using the least-squares algorithm to formulate the line of best fit, which is essentially our regression model. For cross-validation, we used a 70-30 split for our training-testing data.

## Ordered Logistic Regression Model

Logistic regression deals with dependent variables that are categorical. It measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic function, which is the cumulative logistic distribution. The equation of a typical logistic regression is

***y\**** *=* ***x’β*** (2)

where y\* is the unobserved but exact continuous dependent value, x is the vector of independent variables and β is thevector of regression coefficients. Ordinal logistic regression (also known as ordered logit model) is a regression model for ordinal dependent variables. An ordinal variable is similar to a categorical variable, the difference lying in the fact that there is a clear ordering of the variables. Logistic regression follows the proportional odds assumption.

The basic framework of the model and the feature engineering is the same as the linear regression model: creating feature vectors for the categorical independent variables. The only change is that the IMDb ratings are treated as an ordered categorical variable instead of a continuous numeric value. The order is as follows:

0 = [0, 5), 1 = [5, 6), 2 = [6, 7), 3 = [7, 8), 4 = [8, 9), 5 = [9, 10];

For the cross-validation analysis, we used a 70-30 training-test split.

# Trends In The Dataset (Model *A*)

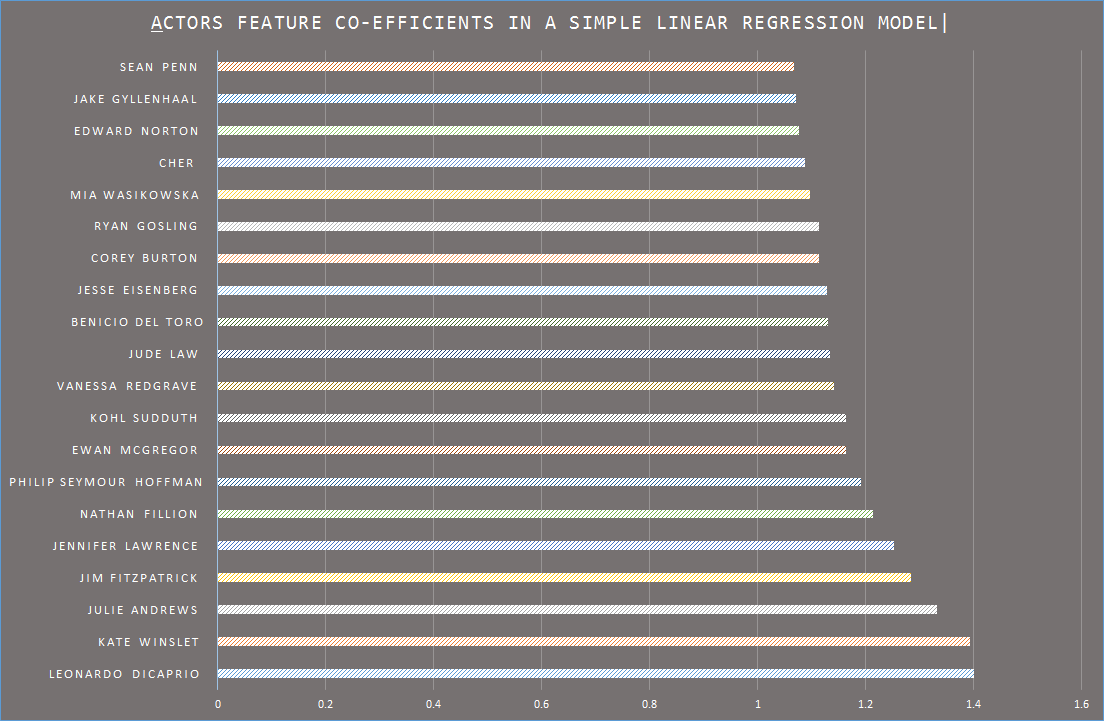
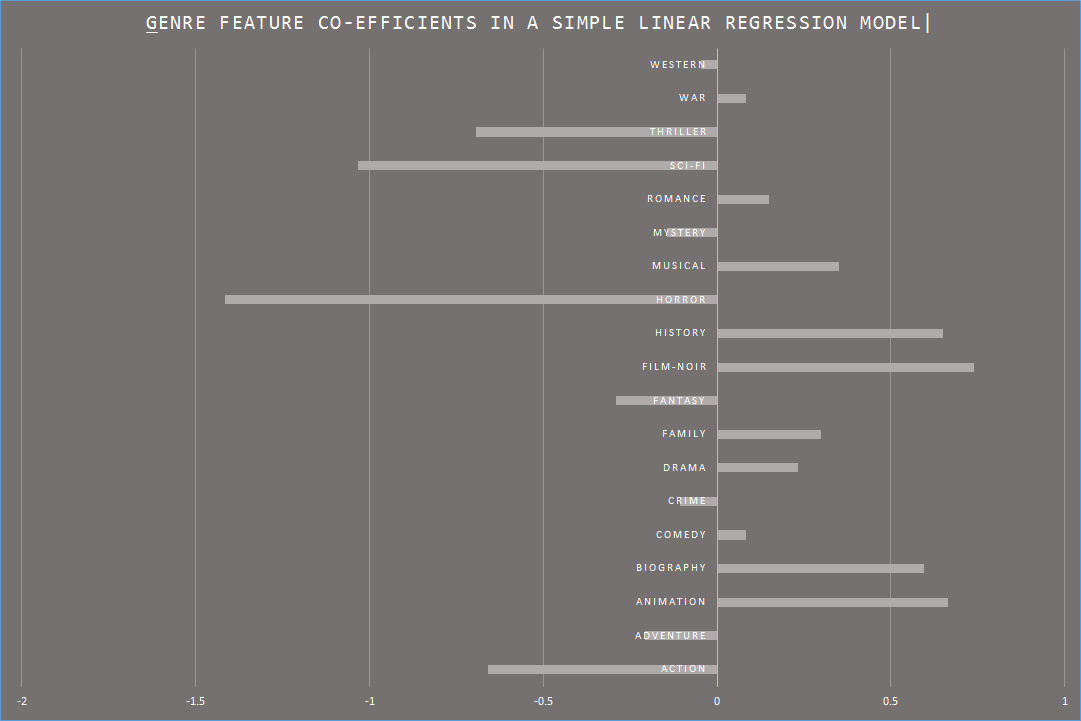
The following figures show some trends observed in the 

Fig. 1: Actors (feature) coefficients vs Actors [top 20]



## Fig. 2: Genre (feature) coefficients ratings probability distribution

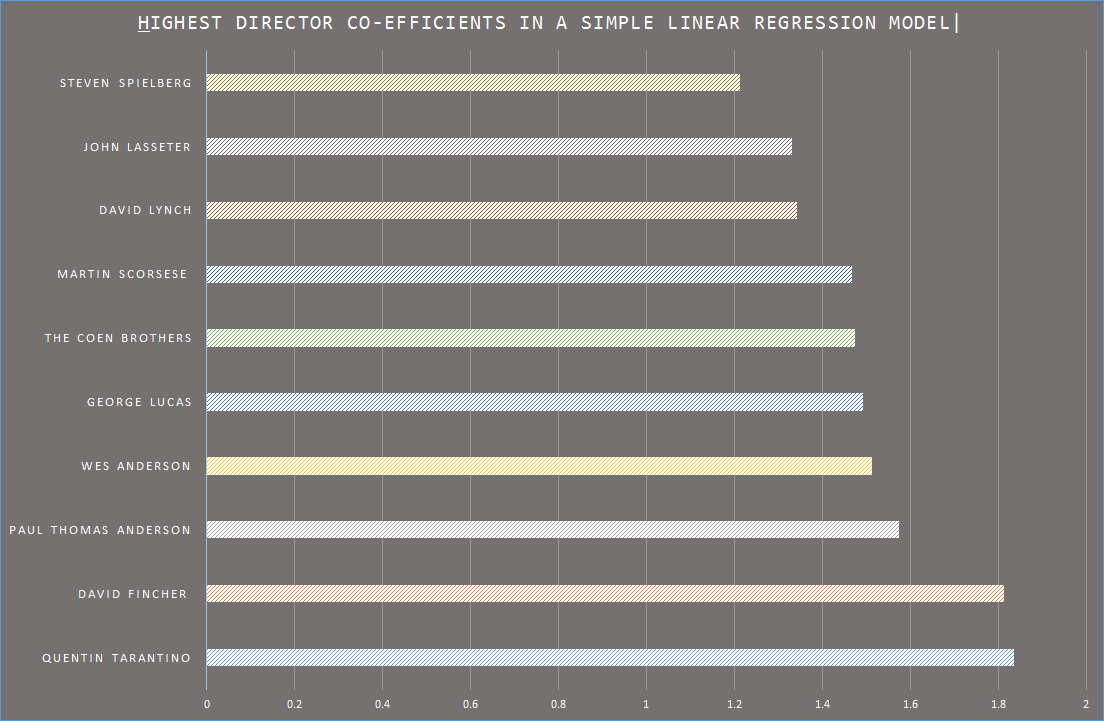


Fig. 3: Director (feature) coefficients vs Directors [top 10]

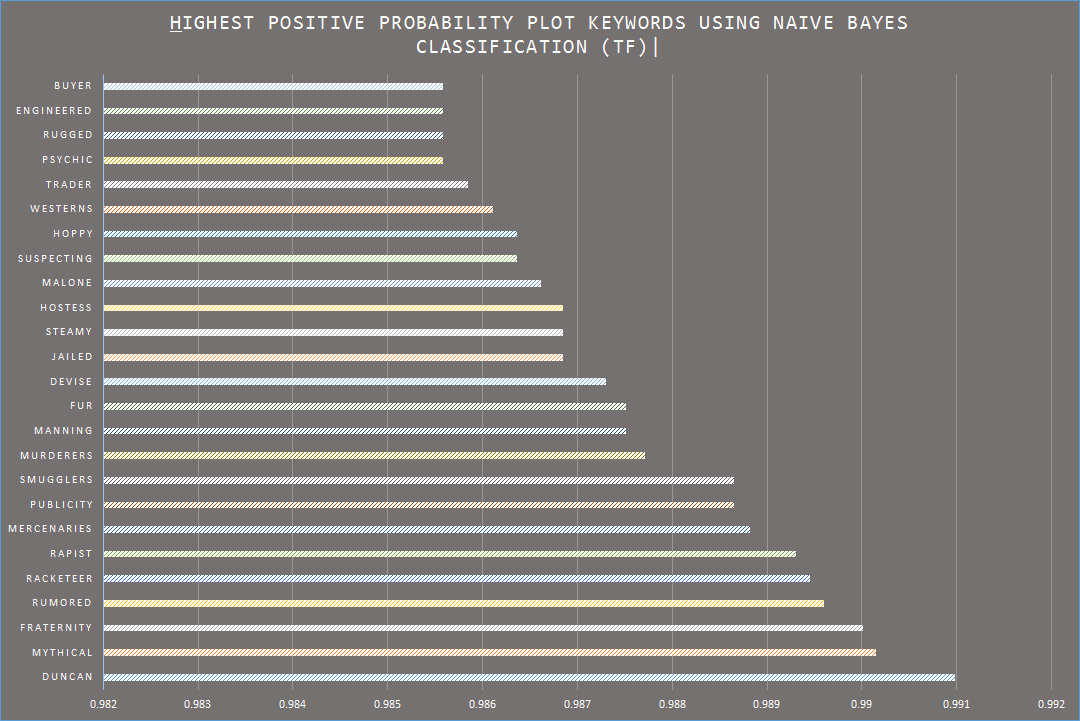


Fig. 4: Plot Keywords highest probability plot (using TF)

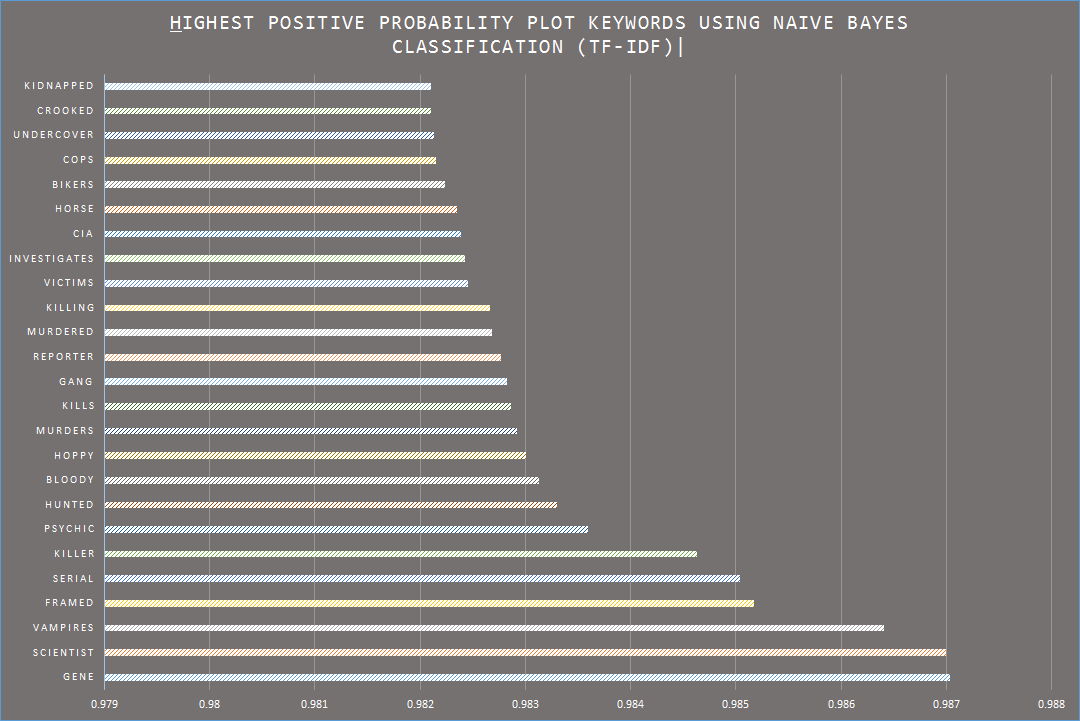


Fig. 5: Plot Keywords highest probability plot (using TF-IDF)

## 

data that give us some insight into the dataset. Figure 1 is a bar graph shows the trend of the highest feature coefficients in a simple linear regression model for the actors in the dataset. Thus, only the individual actors were used as a single predictor for analysis. Leonardo DiCaprio tops this list with a coefficient weight of 1.401. Most of the names in this

list are associated with famous actors, but there are exceptions such as Jim Fitzpatrick, Kohl Sudduth, etc. The reason for this is the noise present in the data as mentioned before: not being able to filter out TV movies adequately. Since the performance of television stars do not generally reflect on similarly on the bigger screen, it is essential to filter out this data.

Similarly, Figure 2 shows the feature coefficients for various genres. Unlike actors, common genres such as Action, Sci-Fi, Thriller are associated with good and bad movies. So naturally, movies with unique genres such as Film-Noir are bound to have higher coefficient value. This trend also signifies that Horror movies are the most commonly made movies with bad ratings.

Figure 3 is shows the trend of the highest feature

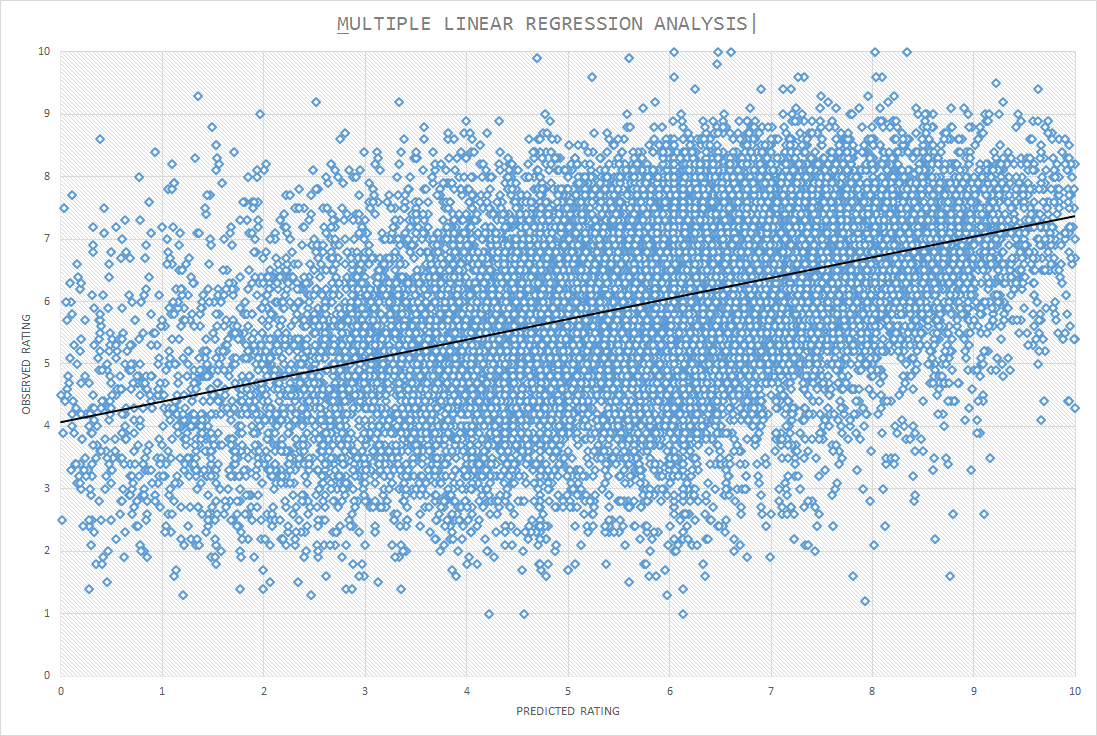


Fig. 6 Predicted Rating vs Observed Rating

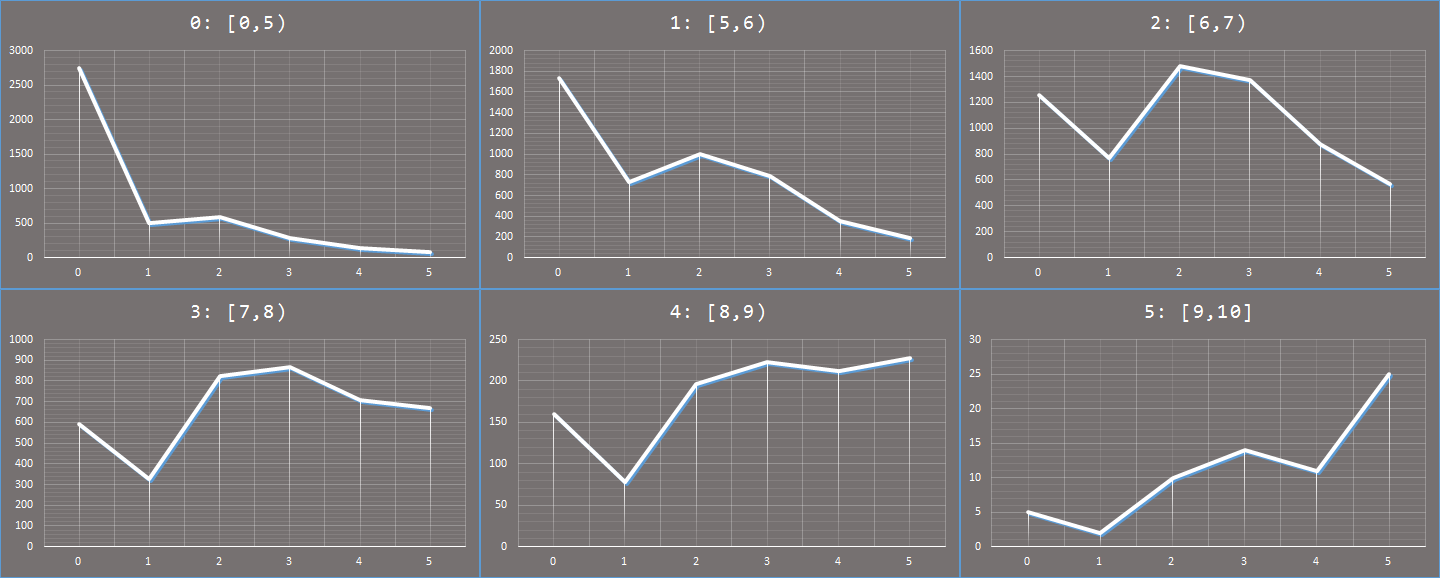


Fig. 7: Predicted Ordered Rating vs No. Of Predictions

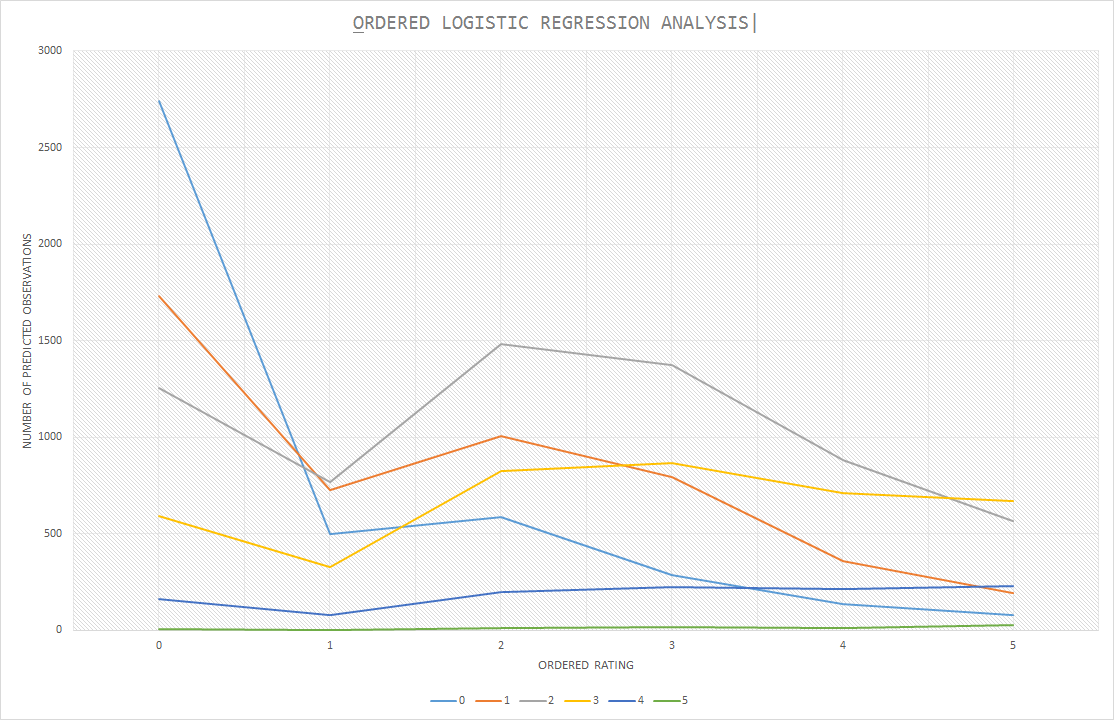


Fig. 8: Predicted Ordered Ratings vs Ordered Rating

coefficients in a simple linear regression model for the directors in the dataset. Figure 4 is a graph of Naïve Bayes using just term frequency. A problem with this approach is that proper nouns and sparsely distributed words get a higher probability.

Since we had some text data, we ran a Naïve Bayes classification model on the movie plots to get some insights about the data. We associated movies with a rating of 7.5 and higher as having positive plots and the rest as negative. We also took both, text frequency and the inverse document frequency approaches to reflect the importance of a keyword to the plot in Figure 5. Words such as gene, scientist, vampires, serial, killer, etc. tops the list of words with highest positive probability.

# Observations And Results

## Linear Regression Model

The independent variables would be the specified actor(s), director(s), genre(s), and plot keyword(s) with their feature values (*xn*) equal to 1. For cross-validation analysis, we used a 70-30 training-test split. Figure 6 is a scatterplot that represents the distribution of movie ratings – the x-coordinate represents the predicted rating, and the y-coordinate represent the observed rating.

Scatter points lying around the **line of best-fit** (black line), indicated a correlation between the observed and the predicted ratings. Given below are the error values:

‣ R-squared Error: 0.2121

‣ Tolerance: 79%

‣ Standard Error: 0.0045

‣ Root Mean Squared Error (RMSE): 1.764

‣ Mean Absolute Error (MAE): 1.370

## Ordered Logistic Regression Model

Figure 7 has six graphs - each of these individual graphs represent the predicted ordered rating on x-axis and the corresponding number of predictions on the y-axis for movies belonging to a certain ordered rating. For example, on the first graph, for movies with an observed ordered rating of 0, the x-axis represents the predicted ordered rating for those movies and the y-axis represents the corresponding number of predictions. For movies with observed rating of 0, 2, 3, and 5, the highest count of predicted ratings belonged to the same. Except for ordered ratings of 1 and 4, but they were still hovering around that range. Figure 8 is an accumulated graph of the six graphs in Figure 7.

Given below are the error values:

‣ Root Mean Squared Error (RMSE): 1.606

‣ Mean Absolute Error (MAE): 1.206

The main issue here is that Scikit-learn or any other prominent machine learning libraries such as Statsmodel does not support a standalone module for ordered logistic regression yet, thereby giving rise to the problem of standardization of analysis results. We used a user-written ordered logistic module (made by Fabian Pedregosa, obtained from his GitHub repository).

Table 1 is a side by side comparison of the two models’ performances.

|  |  |  |
| --- | --- | --- |
| Errors | Model *A* | Model *B* |
| RMSE | 1.764 | 1.606 |
| MAE | 1.370 | 1.206 |

*Table 1: Comparison of the errors of the two models*

# Conclusion

## Ordered logistic regression performed better than multiple linear regression, based upon RMSE and MAE estimators. An R-squared value of 0.21 for linear regression indicates a very low correlation; however, it can be possibly argued that for real-world data it is relatively high as the model is able to explain 20% of the variation. Further research plans could be to look into some more machine learning models, such as Support Vector Machines (SVMs), to get the estimation of the best model for our predictor system. Addition of more relevant features like time of release, budget, as well as analyzing user movie reviews, social network data, Google trends, etc. could possibly help the model predict more accurately. Also, we could look into incorporating movies beyond the ones based in USA, provided complete information is present in the dataset. Developing a web front-end for this model could be highly useful to end-users.

Acknowledgment

This research was a part of the course project for Big Data and Data Science (CSE 4990/6990) at Mississippi State University.

References

1. Nithin V R, Pranav M, Sarath Babu PB, Lijiya A, “*Predicting Movie Success Based on IMDb Data*”, International Journal of Data Mining and Applications, Vol. 3, June 2014, Pages 365-368
2. IMDb.com, OMDb.com

1. [↑](#footnote-ref-1)