**CS436 -Final Project**

**Predicting Movie Revenue with IMDB Data**

**Umut Cem Kayaalti, Egemen Inal, Selin Dinc**

**Abstract**

For our final project, our subject was to estimate the gross revenue of a movie based on prior IMDB data like budget, directors, IMDB rating etc. We used two different models to estimate the gross revenue; multi-layer perceptron neural network and the linear regression.

We expected to find neural networks would fare better than the linear regression. However, when we trained both our models, we found linear regression doing better than the multi-layer perceptron neural network.

**1. Introduction**

Our main motivation was to find a model that estimates the gross revenue based on prior data available to movie studios before a movie was released. We used IMDB as our data. We also were intrigued about which of the features were going to be the most impactful on the prediction.

The problem we wanted to solve is to determine whether to invest into a movie before going into production. To solve this problem, we have decided to implement two different models. The first model we decided to use was linear regression as regression is more suited on estimating continuous values. The second algorithm we decided to use was multi-layer perceptron neural network. Though classification algorithms work bad on estimating continues values we decided to divide our gross revenues into five different buckets that we use as classes.

Umut worked on the models, Selin worked on features and Egemen worked on the data.

**2. Technical**

**2.1 Data Manipulation**

First, we needed to clean our data from irrelevant movies. We choose to discard any data before 1970 due to two reasons. Firstly, the inconsistency of data before this time with a lot of missing values. Secondly, the inflation rates before 1970 was wildly fluctuating so, both our budget and gross revenue values would have been inconsistent.

We also decided to discard any movies that shot in different languages than English. Because the American market is highly influential in determining movies revenue.

Also, we discarded any straight to television movies since their revenue would have been lower than a similar movie with a box office release.

We changed any movies with no rating into unrated to group them. We dropped any data without a budget or gross revenue. Lastly, we dropped any other NaN values using Pandas.

**2.2 Feature Engineering**

Firstly, we removed gross revenue from our data since we were going to use it as a label instead of a feature. We separated our numeric and categorical data features. For the categorical features we used Label Encoder then One Hot Encoder to transform it into the values that we can use them on our prediction models.

Before encoding our data, we had 27 features, after encoding we ended up with 15,003 features and most of them are binary. For example, every actor will behave as a different feature and if the specific actor is in that movie it will have a 1 in that row and column for that feature.

Before using the data in the neural network model, we put the features inside a standard scaler as sum of the features had huge range compared to others. At first, we tried Normalizer as none of the numeric features were negative. However, that method was worse compared to standard scaler.

**2.3 Bucket Determination**

We determined the buckets by ordering gross revenues and splitting the database evenly to determine bucket values.

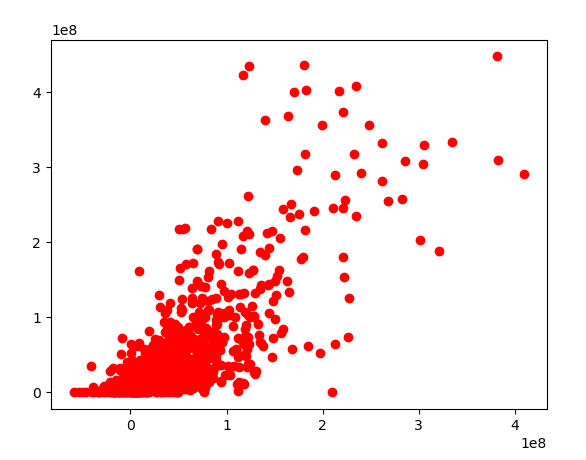
**2.4 Algorithms and Comparison**

We used ScikitLearn libraries for Python 3.7 to implement both the Linear Regression and Multi-Layer Perceptron Neural Network models. While using these models we changed training parameters like; learning rate, hidden layer size and max iterations to further optimization.

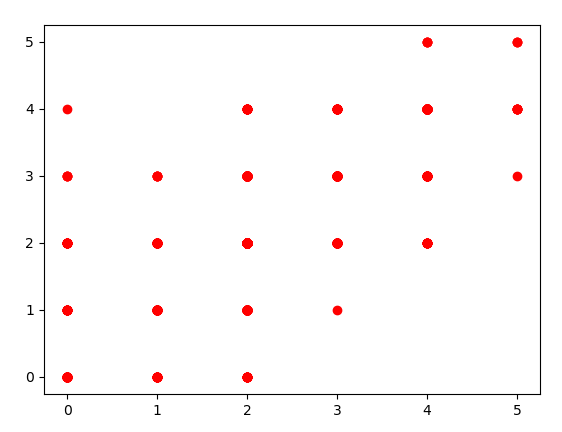
To be able to compare the outcomes of Linear Regression and Multi-Layer Perceptron Neural Network we converted the predictions of the regression model to the same buckets we used for the neural network. This helped us to see how different both of these algorithms predict different results with the same data and features.

**3. Results**

In our testing, we found the Linear Regression model had around 60% accuracy rate while the Multi-Layer Perceptron Neural Network had around 50% without categorical data and around 40% with the categorical data. This shows that our neural network receiving 15003 different features resulted in overfitting. Even with the advantage of buckets neural network performed worse.

****

**Figure 1:** Linear Regression Prediction Plot (x: predicted estimate y: real gross)

****

**Figure 2:** Neural Network Prediction Plot (x: predicted bucket y: real bucket)

**Prior Work**

* Rian van den Ander has a great project using TMDB (The Movie DB)

<https://towardsdatascience.com/what-makes-a-successful-film-predicting-a-films-revenue-and-user-rating-with-machine-learning-e2d1b42365e7>

* Matt Vitelli has a good paper regarding the challenges and different approaches.

<http://snap.stanford.edu/class/cs224w-2015/projects_2015/Predicting_Box_Office_Revenue_for_Movies.pdf>

**Conclusion**

Finally, we learned that data is the most important part of this project. We might have been able to get better results with inflation adjusted budget and gross revenues. We also might have been able to get better results by altering our genre selections.

We found the Linear Regression model had around 60% accuracy rate while the Multi-Layer Perceptron Neural Network had around 50% without categorical data and around 40% with the categorical data. This shows that our neural network receiving 15003 different features resulted in overfitting. Even with the advantage of buckets neural network performed worse.

It shows that our prior expectation about the neural network would predict better than linear regression was wrong, or data was too complicated for our neural network.

**References**

* <https://data.world/data-society/imdb-5000-movie-dataset>
* <https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html>
* <https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html>