Modeling Movie Revenue Using Natural Language Processing

Sneha Kottakkudy, Lasya Pullakhandam
The University of North Carolina at Chapel Hill

1. Introduction

This paper aims to use text analysis, cross-validation, and different modeling methods to build a prediction system that can estimate anticipated revenue for a movie based on a 2 to 3 sentence overview. People have a short attention span so overviews are critical in ensuring commercial success. By determining the relationship between keywords and revenue, this model will be able to estimate the revenue range of an unreleased movie based on the overview.

1.1 Related Works

Previous papers in this field have used metrics such as genre, cast, and screens to predict revenue [1]. Many others have also utilized social media sentiment but we chose to utilize data straight from the source to estimate revenue [2].

2. Data

To get relevant data, we utilized an API provided by The Movie Database [3]. We collected data on movies from 2000 to 2019, with at least 200 movies from each year, and got 6000 observations. We retrieved each movie's id, title, overview, vote average, release date, and revenue in USD.

2.1 Cleaning Data

We removed the 'id' column after data collection. We removed all rows with missing data, which amounted to 9 samples. There were many samples with a revenue of 0. These were cable network movies or videos that people did not have to pay to watch, so we removed these 2107 observations because it is not representative of the population. We removed movies that produced over 1 billion dollars because they were rare and niche cases.

We set the types of all our columns and changed the release date to a date-time object. We also created a new column to categorize the revenue into 5 groups: under 25 million, from 25 million to 75 million, from 75 million to 175 million, from 175 million to 500 million, and from 500 million to 1 billion. Our system will predict the revenue of movies within these categories.

2.2 Data Analyzation

To better understand our data we created some visualizations by different metrics. We created a histogram to categorize the data by their vote average, though we did not find this useful for our model.

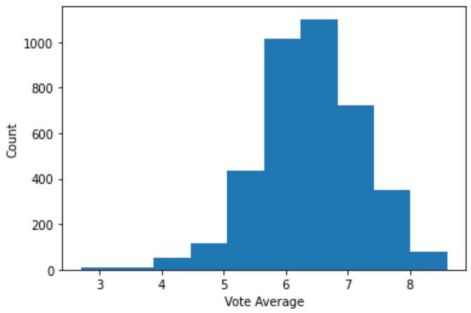

Figure 2.2.1

We also retrieved a statistical summary of our data with the mean, standard deviation, minimum, maximum, and lower and upper quartiles that were the basis for our revenue categories.

	vote_average	revenue
count	3884.000000	3.884000e+03
mean	6.421138	1.442320e+08
std	0.830765	2.196025e+08
min	2.700000	1.000000e+00
25%	5.900000	2.270706e+07
50%	6.400000	6.991741e+07
75%	7.000000	1.680251e+08
max	8.600000	2.797801e+09

Figure 2.2.2

From the movies in our sample, we also created a time plot to document the average revenue each year.

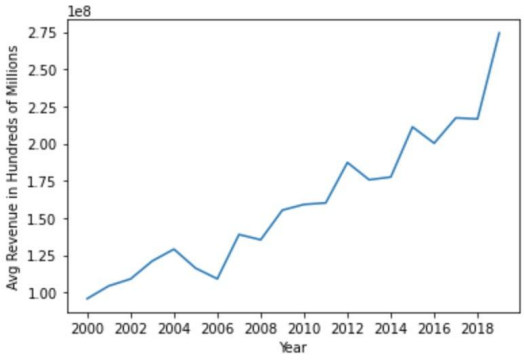

Figure 2.2.3

3. Modeling

We used the random forest to build multiple decision trees and merge them together to get a more accurate and stable prediction.

We chose random forests because it is widely used for classification problems [4][5]. We chose to do k-fold cross-validation [6] because it allowed us to resample our limited data sample for training and testing. We specifically chose 5 folds so that 20% of our data would be the test set.

3.1 First Model

This first model is our baseline model. To preprocess our data before modeling, we made all the words lowercase. We used the Bag of Words model [7] to extract features from the text and create a vocabulary of 14307 unique words occurring in all the movie overviews in the training set.

Then, we converted each text overview into token counts and made sure to keep this as a dense matrix. We made revenue categories a data frame and our bag of words matrix into another data frame. We then proceeded to do 5-fold cross-validation. From the resulting 5 splits of test and training data, we have compiled five confusion matrices.

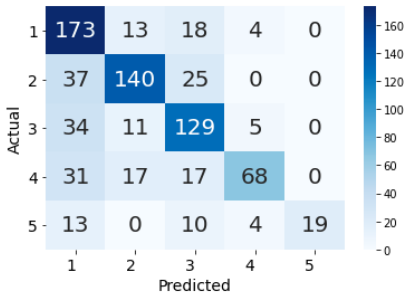

Figure 3.1.1

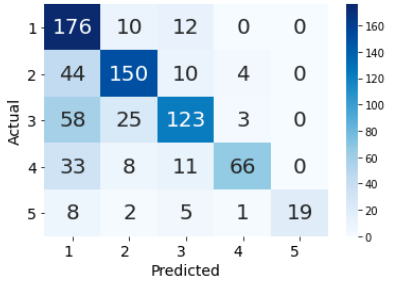

Figure 3.1.2

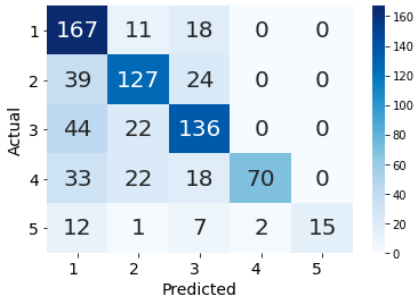

Figure 3.1.3

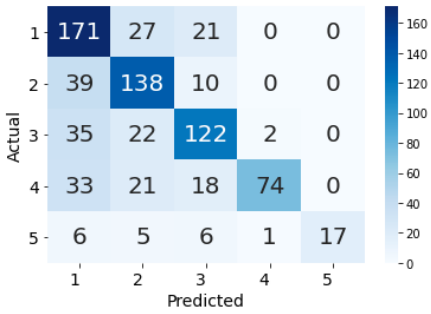

Figure 3.1.4

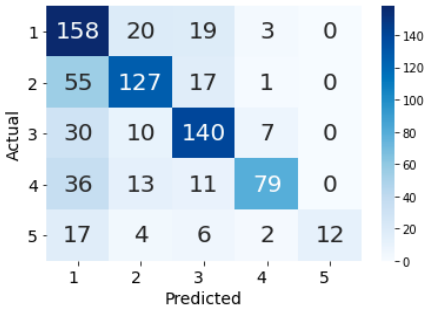

Figure 3.1.5

By averaging the accuracy score of these models created from the 5-fold cross-validation, we determined the average accuracy score to be 68.14%.

3.2 Second Model

In our second model, we wanted to try further preprocessing the data, changing the weight of certain words, and altering the random forest classification.

Before modeling, we removed stop words, removed punctuation, and made all the words lowercase to preprocess our data. Stop words are commonly used words that we want our model to ignore because they are not useful in natural language processing and take up unnecessary space and time. Our resulting vocabulary list after this preprocessing was 15454 words.

This time, we used a weighted case of our Bag of Words representation with TF-IDF [8]. Through this method, if a word occurs frequently, its impact is scaled because it is empirically less informative than words or features that occur in a small fraction of the data.

For the random forest, kept the class weight balanced to adjust weights proportional to class frequencies in the training data. We have compiled five confusion matrices from the resulting 5 splits of test and training data of this new model.

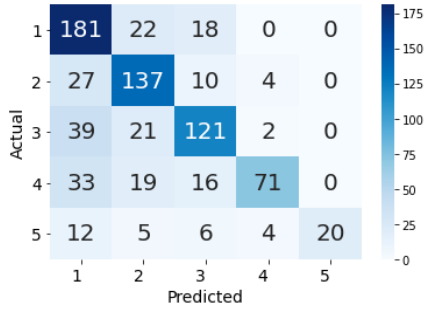

Figure 3.2.1

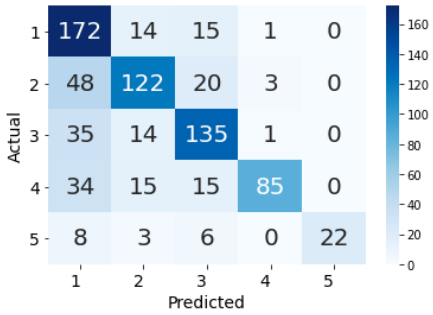

Figure 3.2.2

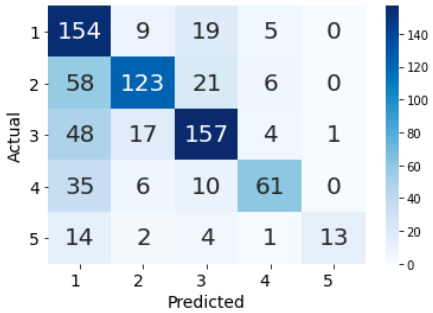

Figure 3.2.3

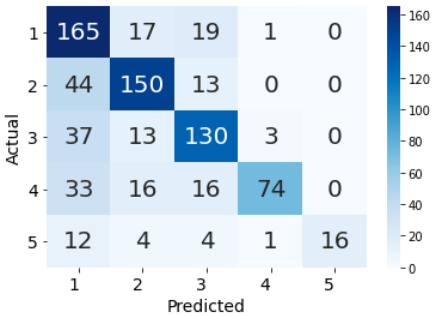

Figure 3.2.4

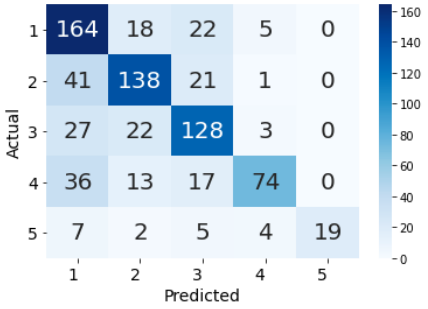

Figure 3.2.5

By averaging the accuracy score of these models created from the 5-fold cross-validation, we determined the average accuracy score to be 68.56%.

4. Conclusion and Further Analysis

In conclusion, our second model did not do much better than the baseline model. Another drawback was the vocabulary list in the second model became larger because removing punctuation counted hyphenated words as new words (e.g. full-time to fulltime). We also would want to create a model that would have worked even with the movies that produced over 1 billion dollars in revenue.

For further analysis, we could create a model that subsets by genre and examines what keywords for each genre generate the same revenue. Another aspect of the model could be to analyze which words contributed the most to the revenue prediction so it would be helpful for studios to improve their overview of a movie. We could also look into the same problem, but account for the movie budget and predict net profit instead of revenue.

5. References

1. Lee, K., Park, J., Kim, I. *et al.* Predicting movie success with machine learning techniques: ways to improve accuracy. *Inf Syst Front* 20, 577–588 (2018). https://doi.org/10.1007/s10796-016-9689-z

2. Ibrahim Said Ahmad, Azuraliza Abu Bakar, Mohd Ridzwan Yaakub, Movie Revenue Prediction Based on Purchase Intention Mining Using YouTube Trailer Reviews, Information Processing & Management, Volume 57, Issue 5, 2020, 102278, ISSN 0306-4573, https://doi.org/10.1016/j.ipm.2020.102278.

3. *The Movie Database API*. (n.d.). The Movie DB. https://developers.themoviedb.org/3/discover/movie-discover

4. Palczewska A., Palczewski J., Marchese Robinson R., Neagu D. (2014) Interpreting Random Forest Classification Models Using a Feature Contribution Method. In: Bouabana-Tebibel T., Rubin S. (eds) Integration of Reusable Systems. Advances in Intelligent Systems and Computing, vol 263. Springer, Cham. https://doi.org/10.1007/978-3-319-04717-1_9

5. K., V. (2020). A Random Forest-based Classification Method for Prediction of Car Price. *International Journal of Psychosocial Rehabilitation*, *24*(3), 2639–2648. https://doi.org/10.37200/ijpr/v24i3/pr2020298

6. Z. Nematzadeh, R. Ibrahim and A. Selamat, "Comparative studies on breast cancer classifications with k-fold cross validations using machine learning techniques," 2015 10th Asian Control Conference (ASCC), 2015, pp. 1-6, doi: 10.1109/ASCC.2015.7244654.

7. Zhang, Yin & Jin, Rong & Zhou, Zhi-Hua. (2010). Understanding bag-of-words model: A statistical framework. International Journal of Machine Learning and Cybernetics. 1. 43-52. 10.1007/s13042-010-0001-0.

8. Qaiser, Shahzad & Ali, Ramsha. (2018). Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents. International Journal of Computer Applications. 181. 10.5120/ijca2018917395.

9. Géron, A. (2017). *Hands-On Machine Learning with Scikit-Learn and TensorFlow*. O'Reilly Media.

10. *scikit-learn: machine learning in Python — scikit-learn 0.24.2 documentation*. (n.d.). Scikit Learn. https://scikit-learn.org/stable/

11. lasya1125/COMP562FinalProject. (2021, May 9). GitHub. https://github.com/lasya1125/COMP562FinalProject