Oscar Nomination Predictions

CS 3892: Special Topics - Big Data
Tucker Kirven, Amy Pickens, Virinchi Juttukonda
May 4, 2016

MOTIVATION AND OBJECTIVE

Every year since 1929, the Academy of Motion Picture Arts and Sciences has handed out awards, Oscars, to films, actors, directors, and others involved in filmmaking. The awards are intended to recognize their achievement and excellence. Each year, the Academy releases lists of nominees for each award category, and the winners are chosen from them. Although we may not know which movie or cast/crew member will win, it is possible to predict what films and cast/crew members will be nominated.

Films are also subject to reviews and ratings from film critics. Popular review sources, including Metacritic and Rotten Tomatoes, allow users to view an aggregate of opinions from professional critics. These reviews and ratings, combined with historical Oscars and cast data from IMDb, can be used to predict which films will get nominated.

The goal of our project is to predict which movies, actors, and crew members will receive Oscar nominations. In particular, we are looking at Best Picture, Best Screenplay, Best Actor in a Leading Role, Best Actress in a Leading Role, Best Actor in a Supporting Role, and Best Actress in a Supporting Role. The project includes data aggregation, data analytics, data transformation, and machine learning and uses three independent data sources: IMDb, Rotten Tomatoes, and Open Movie Database plot summaries.

BACKGROUND AND SYSTEMS

There are plenty of sources that try to predict who will win out of a list of nominees. This is relatively easy to do since there are fewer films to analyze. However, there are not many sources that predict which films and actors will get nominated. This problem required the use of knowledge of skills and tools learned in class in combination with further outside research on machine learning techniques.

The project required the use of several big data tools. We used HDFS to store our files. We also used Spark and Python for data transformation. Scipy and Numpy were used to augment Python's processing functionality. Additionally, we were able to utilize the classifier available in Scikit-learn in our machine learning algorithms. Finally, we used PostgreSQL to retrieve movie information.

DATA SOURCES

Independent Data Sources

For this project, we used three independent data sources: IMDb, Rotten Tomatoes, and Metacritic, the latter two of which were acquired from OMDb.

IMDb (www.imdb.com)

IMDb is a database that contains information on more than 1 million films and TV series. The different categories of information we needed from this were actor name, movie title, movie year, and an actor's billing position that movie. We were able to decrease the number of titles down to approximately 20,000 films because we assumed that films that do not have ratings on either Rotten Tomatoes, Metacritic, or both will not be nominated for an Oscar. Furthermore, we also assumed that the actors most impactful to the film will be in a billing position of 10 or less.

Initially, we tried to get the data directly from IMDb. However, we chose to use the Postgres database used in class to get the information we needed since it was the most efficient way to acquire it. We wrote an SQL query that returned the title of a film, the year it was released, and the actors in the top 10 billing positions in that film for all films that were released before 2016. The result of the query was stored in a text file. Once the text file was created, we used Spark to join the IMDb data with Rotten Tomatoes. This let us filter out only the movies that had movie reviews. Next, we used Python to create the feature set of values for each movie that represented the experience of each of the actors in the top 10 billing positions (i.e. the count of movies they have been in prior to the that movie). A similar method was used for prior Oscar nominations. Ultimately, both of these data sets were fed into a machine learning algorithm that predicts whether or not a film/actor will be nominated. The algorithm uses the following classifiers: stochastic gradient descent (SGDC), Gaussian Naive Bayes (NB), and Random Forest.

Rotten Tomatoes (www.rottentomatoes.com)

For this dataset, we looked at two features, Rotten Tomatoes Critic Score and Rotten Tomatoes Consumer Score, as found on the Open Movie Database (http://omdbapi.com/). These scores are originally found on the Rotten Tomatoes; however, we were unable to get access to the Rotten Tomatoes API within the timeframe of the project, so we used the OMDb instead. We used a Python wrapper (https://pypi.python.org/pypi/omdb/0.2.0) in order to access the necessary data on the OMDB. The scores range from 0 to 100. After filtering out media types other than movies and those movies without critic scores, we had 19,085 rows of data. Each row represents a unique movie. It includes the id, title, year, critic score, consumer score, media type, runtime, metascore, released, IMDb rating, IMDb votes, box office, and country.

We stored the data on the Hadoop File System. After gathering and formatting the data for the Rotten Tomatoes scores and the movies nominated for Oscars, we loaded and transformed the OMDb and Oscar data so that we could use it to create input and output matrices. The input matrix holds the critic and consumer scores. The output matrix uses the Oscar data to signify if a movie has been nominated for an Oscar. Next the data is used to train a stochastic gradient descent classifier, gaussian naive bayes classifier, and random forest classifier using Scikit-learn. Lastly, we evaluated the classifier AUCs. For a baseline, we predicted that movies with critic scores greater than or equal to 75% will be nominated for an Oscar.

OMDb Plot Summaries

Our original intention was to use reviews from critics so that we could make use of the information captured in the model, as shown in Figure 1 below, about sentiment and other potentially useful word groupings to help our prediction. This diagram depicts a set of positive words and a set of negative words plotted based on 2-dimensional representations of their word embeddings (originally 128 dimensions).

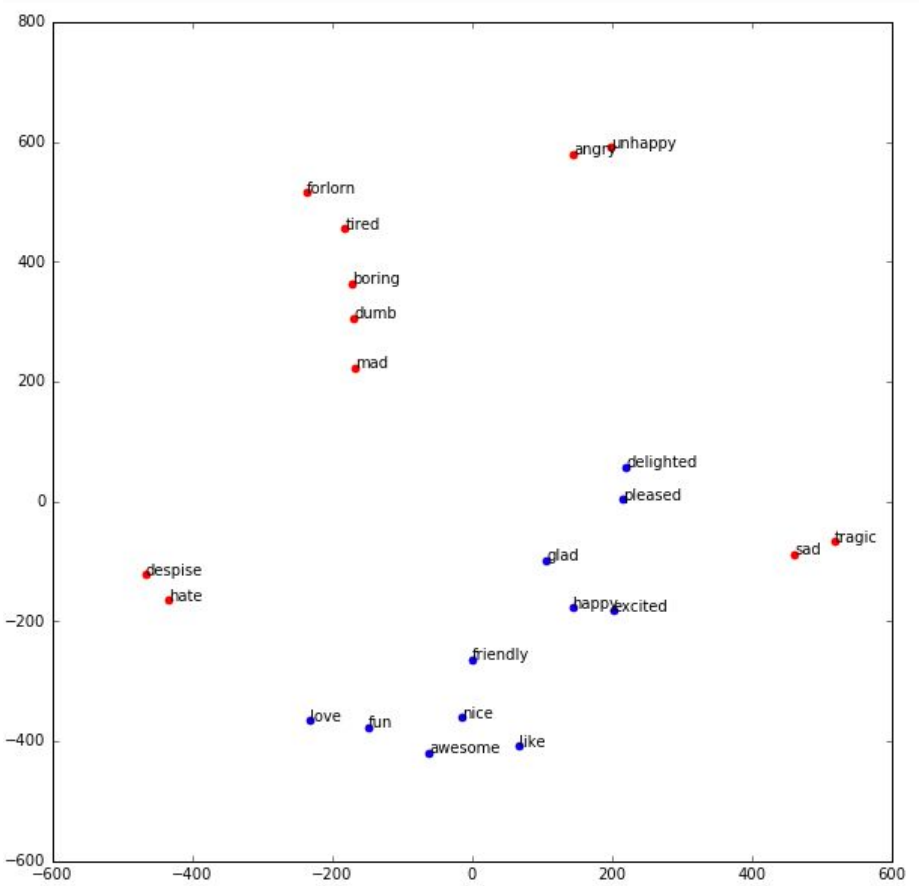

Figure 1: Negative and Positive Words t-SNE plot

The model used to create this diagram is the same as the one we used to create our plot embedding representations. In attempting to scrape for reviews, on sites like Metacritic, we learned it was difficult and near impossible to get the data in a very clean

manner. The site consolidates reviews by linking to the original sites, so each review is formatted differently. Instead of using Metacritic reviews, we decided to use the same method to see if our model could learn information about the plot in a way that would help with prediction.

We gathered our plot summaries from OMDb and created word embedding representations of the plots. We filtered out common words such as “the”, “a”, and “and,” and punctuation. Using a word vector model created from a large amount of news and subtitle data (word2vec), we got a word vector, a numeric representation of the meaning, for each word in the plot summary. Then we averaged them all together to get an embedding for the overall plot. While this method was a long shot, it would have been very interesting had it yielded good results. Other methods such as paragraph vectors were considered, however we did not feel we had a large enough dataset of plots to create a good model.

Dependent Data Source

Our dependent data source is the list of movies nominated for Oscars, including Best Picture, Best Writing/Screenplay, Best Director, Best Leading Actor/Actress, and Best Supporting Actor/Actress. We scraped the Oscar data directly from the IMDb Oscar page (www.imdb.com/awards-central/oscars) for the years 1934-2015. First, we collected the awards data in the format: year, award_title, movie_title, person, won/loss. Next, we cleaned the Oscar data to now only include the 8 categories listed above. Because award titles have changed over the years, we had to ensure that it included all variations of the awards. For example, we concluded that Best Writing and Best Screenplay are the same award. Lastly, we fed the data into Hadoop in the following format: (movie_title, year), win_code. Win_code is 0 if the movie (and any actors in the movie) was not nominated for any award. Win_code is +1 for actor/actress nomination or win and +2 for movie nomination or win. Therefore, if the win_code is greater than or equal to 1, the movie has been nominated or won an Oscar.

RESULTS

IMDb

The model was fairly accurate in predicting whether or not a movie will receive an Oscar nomination based solely on the experience of the cast (how many movies top 10 billed actors for that movie have been in prior to the current movie’s release). Stochastic Gradient Descent had the lowest accuracy, with approximately 60%. Both the Gaussian Naive Bayes and Random Forest had an accuracy of approximately 86%.

SGDClassifier [0.54790230368252901, 0.62026138295685962, 0.53150981820899434, 0.685546875, 0.60179612739234445] **0.597403301448**

GaussianNB [0.86109009210993404, 0.84226640883358561, 0.85420753694673301, 0.83430444452751196, 0.8608515251196 1716] **0.850544001507**

RandomForestClassifier [0.8724917325262036, 0.87560488014498439, 0.88467808231975043, 0.8397362813995 2172, 0.85309509569377995] **0.865121214417**

Table 1: IMDb Actor Experience Results

A similar test was run with Oscar nomination data to a similar result. This test examined how accurately our model can predict movies that will be nominated based on how many Oscar nominations the cast and crew have received in the past. Once again, stochastic gradient descent had the lowest accuracy with approximately 46%. Gaussian Naive Bayes and Random Forest had an accuracy of 86.56% and 85.97% respectively.

SGDClassifier [0.4799008954754922, 0.53442890259714737, 0.83608371642240642, 0.22564601259937395, 0.23037732636569619] **0.461287370692**

GaussianNB [0.86359381545873481, 0.89057404424187658, 0.86229093814257463, 0.86598103545374772, 0.84581155946721154] **0.865650278553**

RandomForestClassifier [0.86221263615416843, 0.89109137820386897, 0.87568595863947729, 0.83002305082969874, 0.83961664898563271] **0.859725934563**

Table 2: IMDb Prior Oscar Results

Rotten Tomatoes (OMDb)

The baseline resulted in an approximate AUC of 0.6954. This method produced 850 true positives, 6550 false positives, 151 false negatives, and 11534 true negatives. Although the number of false positives is high, the number of true negatives is even greater. One run of the machine learning algorithms with 5 folds resulted in the following AUCs:

SGDClassifier [0.5, 0.5, 0.5, 0.50098424108377093, 0.5] **0.500196848217**

GaussianNB [0.86094891866352508, 0.86314210671827474, 0.83928877522808953, 0.8117203483549903, 0.81564643252212399] **0.838149316297**

RandomForestClassifier [0.8361441341811815, 0.82832112247719103, 0.81215440973182185, 0.78309441526126622, 0.81546944137168142] **0.815036704605**

Table 3: Rotten Tomatoes Results

Increasing the number of folds does not change the AUCs. Multiple runs result in similar AUCs, with less than 1% deviation. Therefore, the gaussian naive bayes and random forest classifiers surpass the baseline, but the SGD classifier does not.

As shown in the following figure, the number of movie reviews and critic scores has grown exponentially with time. If we exclude movies from before 1990, it results in an AUC of approximately .95 which is over 10% better than the random forest classifier over all years.

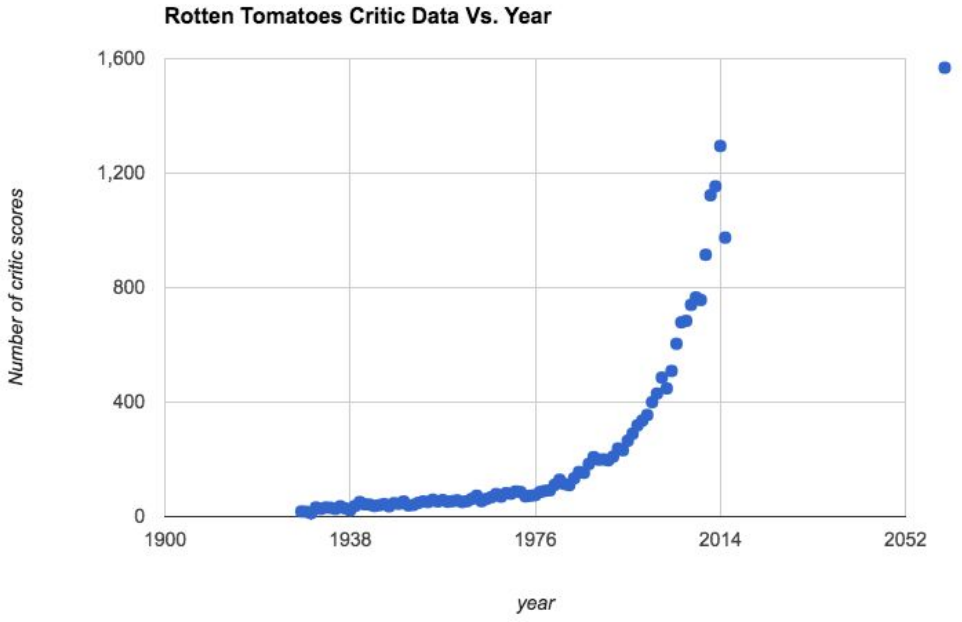

Figure 2: Exponential Increase of Rotten Tomatoes Critic Scores

Plot Summaries (OMDb)

Using embedded plot summaries did not produce accurate predictions. It provided only around 55% accuracy which is not significantly better than randomly guessing (50%), as shown in following table.

SGDClassifier [0.49763313609467458, 0.52455621301775146, 0.46301775147928992, 0.65784023668639047, 0.49940828402366866,...] **0.509521989415**
GaussianNB [0.71301775147928992, 0.66782544378698216, 0.61220414201183426, 0.48905325443786984, 0.63113905325443787,..] **0.547437379313**
RandomForestClassifier [0.77278106508875744, 0.49874260355029582, 0.52677514792899405, 0.67714497041420119, 0.60968934911242612,...] **0.538091940699**

Table 4: Plot Summaries Result

In order to improve this accuracy, we attempted to create a model that could identify trends in plots and how they affect Oscar nominations with respect to time. We simply added the year of the movie's release to the feature set. This addition increased our accuracy to 71%. This result is interesting; however, we believe including this feature is unfair due to the nature of our data. It is sparse for earlier years and very dense for more recent years. For example in the early 1930's through late 1940's, we have data for about 40 movies and roughly 10 oscar nominations per year. In contrast, for 2014, we have data for 920 movies and only 9 oscar nominated movies. Therefore, the classifier simply learned that movies with lower dates were 'more likely' to be nominated. To verify our reasoning, we used only the year as a feature. This test resulted in 78% accuracy. From these results, we decided to keep the year of the movie out of the feature set.

Combined Independent Data Sources

To further test our predictions, we aggregated our independent data sources. For our combined prediction, we used the following features:

- OMDb: critic score (from Rotten Tomatoes), consumer score (from Rotten Tomatoes), film duration, metascore (from Metacritic), imdb-rating, imdb-votes, year
- IMDb: actor experience and prior Oscar wins by cast
- Plot Summaries

We created a file that can test any combination of these features by using booleans to include or not include a specific feature. Our aggregated program can also be run over any subset of years. We found that including the plot summary data does not improve

our prediction accuracy; we receive better or same results without this feature because it mostly adds noise. We also found that running the program over all years resulted in similar accuracies as running it only between 1990 and 2015. The following are the results of one run over all features except plot from 1990-2015.

SGDClassifier [0.5, 0.5, 0.5, 0.5, 0.5] **0.5**

GaussianNB [0.9380531263091747, 0.95058150956740406, 0.94284441999659474, 0.91795017877490082, 0.93620093513025027] **0.937126033956**

RandomForestClassifier [0.94844470046082952, 0.97781030214922793, 0.93321807918483879, 0.9322063311199299, 0.92510444907207312] **0.943356772397**

Table 5: Aggregated, 1990-2015

The next box shows the results for years 1934-2015 using all the features except plot summaries.

SGDClassifier [0.5, 0.60792817494817097, 0.5, 0.5, 0.5] **0.52158563499**

GaussianNB [0.90519802679816919, 0.91031231191065332, 0.92290761051294057, 0.87941550190597206, 0.91503042867651974] **0.906572775961**

RandomForestClassifier [0.93713037796425036, 0.94259011569584705, 0.94614207851267307, 0.936488497291513 4, 0.95181736106466919] **0.942833686106**

Table 6: Aggregated, 1934-2015

The random forest classifier beats the gaussian naive bayes classifier by 4% and the SGD classifier by 42%. The ranking of accuracy of these classifiers is approximately the same for the individual independent data sources and the aggregated data.

DISCUSSION

The accuracy of the predictions by the aggregated program is greater than that of each individual independent source. The following table shows a summary of average classifier scores:

Data Source/Classifier	SGD	Gaussian NB	Random Forest
IMDb: Actor Experience	0.5974	0.8505	0.8651
IMDb: Prior Oscars	0.4612	0.8656	0.8597
Rotten Tomatoes Scores	0.5001	0.8381	0.8150
Plot Summaries	0.5095	0.5474	0.5380
Aggregated	0.5215	0.9065	0.9428

Table 7: Summary of Results

It is clear from this table that all the features combined perform better than they do individually and that gaussian naive bayes and random forest classifiers are more accurate than stochastic gradient descent. As expected, plot summaries are the least accurate predictor of Oscar nominations.

CONCLUSION

Based on the results, we can conclude that our model can accurately predict which movies will be nominated for Best Picture, Best Screenplay, Best Actor in a Leading Role, Best Actress in a Leading Role, Best Actor in a Supporting Role, and Best Actress in a Supporting Role.

For each classifier, stochastic gradient descent was the least accurate method. Gaussian Naive Bayes and Random Forest were within 5 percentage points of each other in all cases. Of the classifiers used, the plot summaries provided the least accurate result. This was to be expected since they had limited assessment of the film. The aggregated classifier had the best results, with both the Gaussian Naive Bayes and Random Forest showing an accuracy of greater than 90%. This was also expected since combining the data allowed for a holistic view on the film instead of relying on one aspect. The remaining three classifiers were similar in their Gaussian Naive Bayes and Random Forest results. Of those three, the “Actor Experience” classifier had the best stochastic gradient descent result while “Prior Oscars” had the worst.

LIMITATIONS AND FUTURE WORK

The main limitation we encountered was the availability of necessary data. First, we were not able to access the Rotten Tomatoes API directly, so we had to look into using another source, Open Movie Database. Fortunately, OMDb gives the Rotten Tomatoes

scores, Metacritic score, and brief plot summaries. Second, we were not able to access all of the information we needed from IMDb. We had to scrap the website for awards information. Third, we were unable to get usable data directly from the Metacritic website. We had hoped to use sentiment analysis on these reviews; however, because clean data was difficult to get, we decided to use plot summaries instead.

The second limitation was the consistency of the data over time. Although the Academy Awards have taken place since 1929, the data we are using to predict the nomination of these awards was not prevalent until the world wide web became commonplace. Many of the older movies do not have critic reviews or scores.

There are several ways that the project could potentially be extended. One future extension is the addition of individual movie reviews. Reviews would provide a better assessment of the film than the plot summaries currently do. They would also provide better sentiment analysis. The model could also analyze popular opinion about films by analyzing tweets or Facebook mentions. Although popular opinion may not be a better predicting tool than critic reviews, it can still be used to provide further sentiment analysis.

DIVISION OF WORK

Amy Pickens

- Created and formatted Rotten Tomatoes data
- Used Rotten Tomatoes scores for prediction
- Worked on final report

Tucker Kirven

- Created and formatted the dependent data source
- Created and formatted the plot summary data
- Used plot summary data for prediction
- Aggregated independent data sources and used them together for prediction

Virinchi Juttukonda

- Wrote PostgreSQL queries to retrieve IMDb data
- Helped write the Spark program to clean/filter IMDb data
- Worked on final report

GITHUB REPOSITORY

https://github.com/kirvenjt/oscar_prediction