Predicting Academy award winners

Ashwin asokan
Avneendra kanva

Introduction

Academy awards is an annual award ceremony hosted by Academy of Motion Picture Arts and Sciences to recognize excellence in cinematic achievements. Awards for various categories are conferred based on votes from the Academy's members. As the oldest awards ceremony in the media, the Academy Awards is responsible for bestowing the penultimate achievement in cinema, the Oscar, with which comes distinction and prestige. So the task is, given a list of nominations for a particular category, predict the winner. Are there in fact predictive features of a nominee which make him, her, or it more likely to win? Our idea to explore this question was to use movie attributes such as runtime, cast, genre, plot keywords, crew, reviews to predict the movie's academy award prospect. Particularly We wanted to investigate how well we can blend discrete features extracted from movie metadata and features from raw review text to improve the overall performance.

Problem Definition

In this project we tried to build an effective set of models to predict the results of past academy award results given the list of nominees and winner. We picked 6 estimators to evaluate for each of the top 5 categories of awards mostly based on their prominence and close relevance to the movie in question. For our prediction task in particular, input would be set of feature vectors, each one representing the nominee with in that category and output would be binary results against each nominee that indicates whether the nominee would win or not. we are trying to use the full review text of the movie to identify particular words and phrases that predict the award winning prospect.

Deliverables

1. Dataset - Well schemed data set that we collected from online resources.
2. Set of models, per category that better predicts results.
3. Set of features, per category that helps in better prediction.

Data & Features

We tried to use metadata that very well could be collected & comprehended for any movie that limited us to standard film attributes. Imdb, rottentomatoes, movie review query engine served as our main contributors for the data. First we collected an aggregated list of films and persons that were ever nominated for academy awards from its inception. We focussed on five categories in particular, Best Picture, Best Director, Best Actor, Best Actress, Best Cinematography ignoring less substantial categories like honorary awards conferred every year. For categorical metadata features like genre, we relied on mostly IMDB as source and turned them into sparse feature vectors of indicators (eg: genre=drama).

For text based review features, we relied on rotten tomatoes and movie review query engine which allowed us to scrape reviews published online from leading newspapers and independent critics. Once we had the reviews we had broken that down into bag of words with most frequent and less frequent words being dropped. We also did some amount of tuning on the text like stemming, removing stop words. We tried to move from bag of words to multiple grams like bigram, trigram but found they weren't helping much.
We have aggregated them all, the IMDb and Rottentomatoes (both audience and critic) ratings, release month, writer, director, cast, MPAA rating, year, run time, and genre feature data for the list of movies. All collected data has been discretized and binned separately. Since we are binary labeling to indicate the presence of an actor/actress in a movie, as well as labeling our known Academy Award victories for each entry we get huge sparse set of features. We decided to limit the number to top 5000 features to manage training time.

Approach

We look to compare predictive power of two kind of features, one discrete metadata about the movie and another text movie reviews and see whether we can ensemble them in sensible way to improve overall effectiveness.

We spent some time investigating the applicability of particularly two neural network models convolution and recurrent neural networks for text based movie reviews features. We heard that lately there has been huge success with neural networks and natural raw data much like our review text. But we weren't quite successful in exploiting their usefulness in our data set. One thing, they were running for hours for our data sets in our laptops to see any results. If we tried to evaluate the parameters with sample from our datasets, the results were irregular for our changes to parameters and feature representations. The performance results that we observed were not better than what we achieved with bit more of feature engineering. So we realized we have a much better chance exploring linear models within our timeframe.

We resorted to familiar linear classifiers discussed in class like gradient descent, svm, random forest, decision tree, adaboost for evaluation most part. For testing the features we started with hold out cross validation in which we trained with data before 2005 to test against data after that. Since naturally year acts as grouping factor for the awards it made more sense to have past data in training for future prediction. So most of the results reported were hold out cross validation results on our dataset.

Models

1. **Baseline:** In our hypothetical baseline model, we used random selection as the approach to select the winner. We decided that each of our model should be better than the random selection at best to be applicable for feature engineering. Our baseline approach results in 30-40% success rate since number of nominations varied across categories and over the years.

2. **Logistic Regression:** Logistic regression actually tries to learn a sigmoid curve to distinguish between positive and negative training examples. Weights for the curve are learned through maximum likelihood estimates. We did little bit of parameter tuning like l2 regularization, log loss to keep a check on overfitting.
3. **SVM:** We experimented with common kernel, radial based, polynomial and settled for linear kernel based on initial results that we observed.
4. **Decision Tree:** We considered decision tree since we learned from the data set that simple predicates like best picture director some time gets his award as well sometimes do well indicate the result. Decision trees seemed like it would easily learn simple decision rules that are easy to observe and evaluate sanity of the model for improvements.
5. **Random Forest:** The rationale for random forest is pretty much the same as decision trees where as here it will ensemble no of small decision tree classifiers which seem to work well on small subsets of data. Basically it would average the decisions overall to improve accuracy and control overfitting.
6. **AdaBoost:** On similar lines, Adaboost basically fits the overall data set on broad decision tree classifier and on subsequent iteration adjusts the weights of the missed examples to make sure they are focussed more.

Feature Engineering & Error Analysis

- **Baseline:** To start with we picked up all possible metadata about the movie and about the person for respective category for consideration. Predominant attributes we used are MPAA ratings, IMDB user ratings, film length, release month, cast, technical crew, genre and plot keywords. For example,

  [‘The Martian’: {‘genre=Sci-Fi’: 1, ‘genre=Drama’: 1, ‘producer=Twentieth Century Fox Film Corporation’: 1, ‘length’:144, 'mpaa':’pg13’,‘actor=matt damon’: 1,'actress=jessica chastain':1,'director=ridley scott':1 ....}]
  With this feature set we got our baseline results around 0.5 accuracy and f1 score more or less across the models.
- **First Pass:** One problem we noticed in our features is that they were highly sparse in the sense that huge number of them that were seen in training set weren’t seen in test. Rationale behind such an observation is that most often academy awards tends to be a one time thing for an individual lifetime that he gets one or two. There were very few elite individuals who seem to have repeatedly getting nominated over time. For instance actress Meryl Streep got nominated 15 times and won three times so far. Otherwise most part of the distribution is more often skewed that it becomes apparent to overcome this with better representation.

We improved our binning with better size like rather than having discrete values we used buckets instead. For example Replaced {'running time':144} => {'running time>120': 1} The weight vectors were used to decide the importance of the feature to the result. Some common features like language, critic ratings seem to have been same across the nomination set that they seem to have hurt the accuracy overall. We tried to get rid of such correlated features. That seemed to have helped.

- **Second Pass:** For the text reviews we tried to get in sentiment of the review estimated from other open source model as separate feature. But as expected that didn't seem to help much and we found positive sentiment for nominated movies. We realized that movies that win awards generally win at multiple venues like other award shows that happen before academy award show. We didn't get to test that hypothesis due to time constraint. But we suspect as per our references it would have helped immensely.

Results

Award Category	Gradient Descent	Random Forest	Decision Tree	AdaBoost
Best Picture	(0.5,0.5)	(0.64,0.63)	(0.60,0.61)	**(0.64,0.64)**
Best Director	(0.56,0.56)	(0.55,0.55)	**(0.65,0.65)**	(0.63,0.64)
Best Actor	(0.52,0.52)	(0.47,0.47)	**(0.53,0.53)**	(0.49,0.49)
Best Cinematography	(0.5,0.5)	**(0.6,0.6)**	(0.54,0.54)	(0.51,0.51)
Best Actress	(0.49,0.49)	**(0.63,0.63)**	(0.61,0.61)	(0.61,0.61)

Figure. 1 (Precision, Recall)

This one shows the precision and recall values for 5 different award categories under different estimators we tried. We are giving more importance to relevance metrics than accuracy since we have a skewed training set to work with.

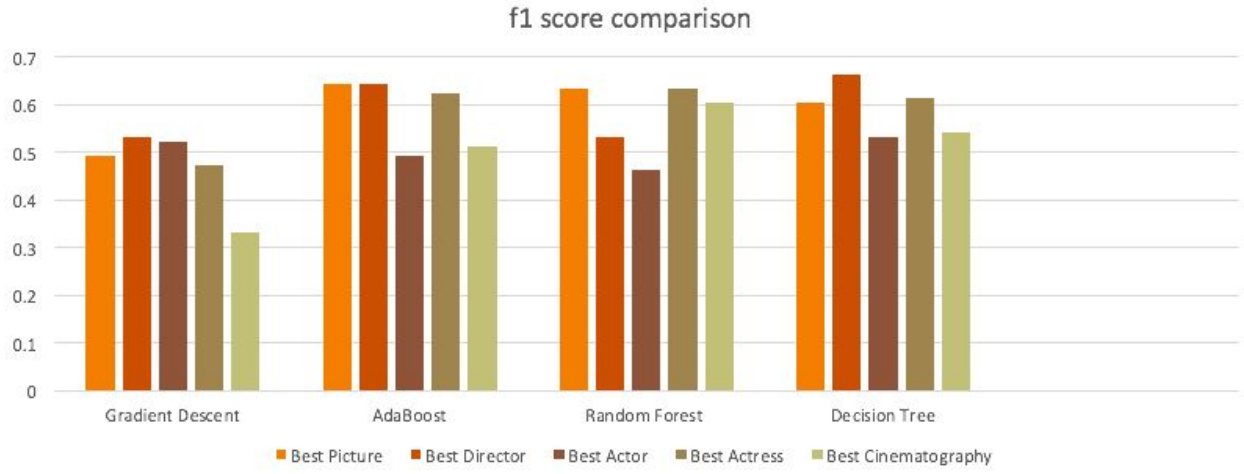
Award Category	Gradient Descent	Random Forest	Decision Tree	AdaBoost
Best Picture	(0.5,1.57)	(0.63,0.275)	(0.60,0.231)	**(0.64,0.28)**
Best Director	(0.56,0.14)	(0.55,0.11)	**(0.65,0.31)**	(0.63,0.279)
Best Actor	(0.52,0.045)	(0.47,-0.05)	**(0.52,0.056)**	(0.48,-0.02)
Best Cinematography	(0.5,0)	**(0.6,0.202)**	(0.54,0.08)	(0.51,0.02)
Best Actress	(0.49,-0.012)	**(0.63,0.27)**	(0.61,0.23)	(0.61,0.23)

Figure. 2 (Accuracy, Matthew's Coefficient)

This one shows the accuracy and matthew's correlation coefficient for same award categories.

Third one below shows the histogram plot of f1 score comparison of classifiers that we used.

Figure. 3 (f1 score)



Challenges

Neural network models training wasn't friendly - Both the neural network models that we tried were more of like a black box which doesn't seem to reveal indicators that we can use to improve them. We had no clue whether changes that we were making are helping us clearly. The models were training for a very long hours without sufficient progress indication. We settled for familiar linear models and decision trees as good opportunities to work with after a rough run.

Unbalanced training set - Our data set by nature was skewed with more false examples than true ones as the award process selects 6-10 nominations and the award goes to single winner. In our data model we picked the nomination results as input and tried to predict the winner. This left us with skewed distribution of binary results to work with. We investigated couple of approaches to negate that,

1. Undersampling the negative examples.
2. Oversampling the positive side by generating similar dummy examples.

We found undersampling the majority by replacing cluster of samples by the cluster centroid of a K Means algorithm work well. So the tactics is to use K-Means to fit the majority data, the number of clusters equal to no of minority samples we have and compute the cluster centroids. The majority samples are then completely replaced by the set cluster centroids.

Feature engineering - We used the raw classifier result as baseline and did a bit of feature improvements to guide the training. We dropped some common features which were not distinguishing the results in appreciable way. We have tried to document our process in more detail above in separate section above.

Conclusion

We list below best models against each category of awards based on our observation. We can generalize that decision tree based classifiers were able to fit our data set better than linear models. This again fits well with the general hypothesis that there isn't a strong linear relationship between movie attributes and the winning prospect. Bunch of academy award members come together to vote on best craft seems to lean towards bunch of non linear factors to decide whom to vote for. There are definitely some hidden external factors such as social relevance of the picture, maturity of the audience that we haven't been able to account into calculation.

Category	Best Estimator	Key Features
Best Picture	AdaBoost	Include[critics ratings, plot keywords]
Best Director	Decision Tree	Exclude[plot keywords, movie release date]
Best Actor	Decision Tree	Include [user ratings] Exclude [plot keywords]
Best Cinematography	Random Forest	Exclude[user ratings, plot keywords]
Best Actress	Random Forest	Exclude [cast members in the movie being nominated, plot keywords]

We conclude that our results strongly convey that best estimator for such a chaotic voting process can't be a linear one but definitely something that can capture non linear ensemble of small decisive factors.

Future Directions

1. Modeling the scenario as regression problem with weights to each nomination - Ideally the academy award process involves 6-10 entries as input and one entry emerges as winner at the end. We have turned the situation into classic binary classification problem which leaves us with skewed distribution and no guarantees on results. Instead we would like to model the classification not as simple binary problem but as a selection from a nomination set where the model need to assign weight to each nomination like in a regression problem. The weights across the nomination set should sum up to one. We haven't quite figured out how to construct such a model effectively though.

2. Exploring layered features - Our initial thought which we mentioned in the project proposal was to explore the layered features more to see how well they help in the

classification process. For example like using features like particular actor acting in a movie made by director having no of wins in his own category.
Most of the features we used were one dimensional and related to the variable directly with respect to the movie, we didn't explore more than one level of relation. We hear that exploiting long range dependencies between variables is helping immensely with results nowadays.

3. Ensemble the metadata based classifier and text classifier - Another initial thought we had in the proposal is to merge the features from discrete metadata about the movies with the raw text based reviews for the respective movie. We weren't able to merge them in meaningful way to improve the results. We kept the classifiers separate by the end. But it would be interesting to see how we can ensemble both of them for better results.

Who did what?

There was academy award winners data available in imdb. We got the nominations and winners list from there. All scripts written for the efforts are in python. Task list below.

1. Initial exploration of CNN model applicability - [Avneendra]
2. Initial exploration of RNN model applicability - [Ashwin]
3. Dataset extraction scripts: Text reviews - [Ashwin]
4. Dataset extraction scripts: Movie Metadata - [Avneendra]
5. Models script implementation - [Avneendra]
6. Unbalanced data set normalization exploration - [Ashwin]
7. First pass feature modeling and analysis - [Ashwin]
8. Second pass feature improvements - [Avneendra & Ashwin]
9. Error analysis & conclusive remarks - [Ashwin]
10. Plots & reports - [Avneendra & Ashwin]

Repository - https://github.com/Ashwinasokan/PredictAwards

References

1. Predicting the 85th Academy Awards - Stephen Barber, Kasey Le, Sean O'Donnell [link]
2. Predicting box-office success of motion pictures with neural networks [link]
3. Movie Reviews and Revenues: An Experiment in Text Regression [link]
4. Unbalanced Dataset handling tactics - Quora Discussions [link]
5. Pre-production forecasting of movie revenues with a artificial neural networks [link]