Predicting Movie Genres

Paul Harris, Luke Morgan-Scott, Mitchell Foster

CS109b | Spring 2017

Sections:

1) Project Motivation and Goals
2) Data and EDA
3) Genre Configuration
4) Traditional Statistical and ML Methods
    -Features
    -TF/IDF bag of words
    -Random Forest Classifier
    -Logistic Regression
    - Discussion
5) Deep Learning
    -Poster Characteristics
    -CNN From Scratch
    -CNN Pre-Trained
6) Conclusion

Project Motivation and Goals

Project Motivation:

This project was motivated by the desire to assign a particular genre to any given movie without having to watch it but instead by using only the available metadata, such as a movie's plot keywords, a list of actors/director, budget, movie poster, etc. The amount of movies grows larger each year; For the sake of efficiency it is important to continuously seek to refine and examine the ways in which we classify and categorize this information. Several business models (Netflix, Hulu, etc.) rely on 'in-house' built classification techniques for their collection of movies, yet their algorithms are frequently updated and expanded to represent their collections in even greater detail. This is to show that there is no single best approach to algorithmically matching a movie to any specific genre or set of genres. Rather it all depends on how you wish to approach the task and how specific you choose to be.

The genre of a movie is also a major component/factor on which movie viewers base their movie selections. Movie genres encompass so much of what the movie has to offer, from the plot to the cinematography. While a viewer could focus on a main character (i.e their favorite actor/actress), that narrow view would leave out many other possible factors that could help or hurt the selection of a specific movie. Genre basically combines all of these into a few select words and inevitably allows the viewer to make a simpler selection.

Goals:

For this project we had several sequential goals in mind.

1) To seriously consider what a genre or set of genres sought to convey about a movie and how reported genres could be used in a simpler way
2) To use a movie's metadata to classify genre
3) To build a model which was actually 'useful' (Outperforms base case, minimizes false positive rate)
4) To examine whether deep learning with movie poster information would prove useful for genre classification

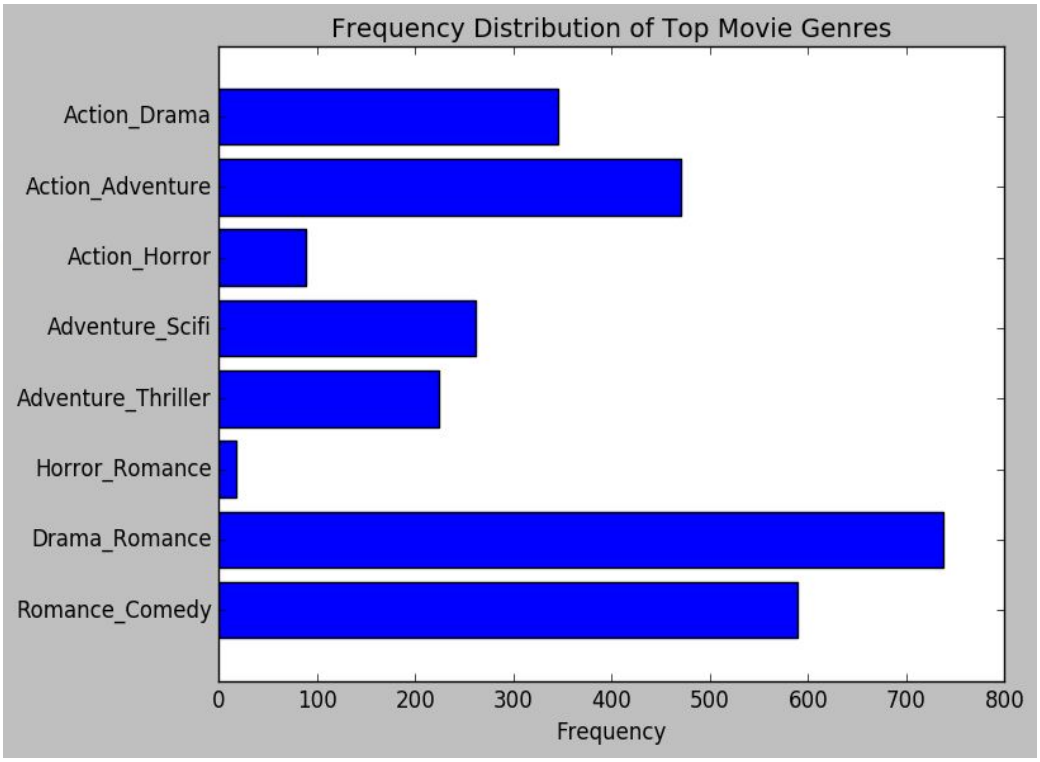
---

Data and EDA

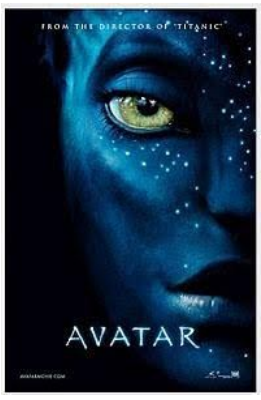
We relied on The Movie Database (TMDb) and Internet Movie Database (IMDb) to provide the majority of the data for this classification project. Combined these databases contain well over 500,000 movies from which to sample. For our traditional statistical methods, we decided to use a stratified sample of about 1500 movies with budget, director, leading actor, and plot keywords as metadata, motivated by what we believed to be likely indicators of a movie's genre. For deep learning we decided to use 5000+ observations along with their associated poster pictures. Although we attempted to control for any genre being under/overrepresented in our dataset

during the initial data pull, things quickly became more complicated: 1) Some movies have a greater amount of genres applied to them than others, 2) Of these movies, certain genre pairs seemed to occur more frequently than others, 3) This caused persistent imbalances in the reported genres in our data. We decided to check the overall frequency distribution of genres in our dataset as well as the frequency of common genre pairs to further examine how genres are related to each other.

```
Counter({'Action': 1153,
         'Adventure': 923,
         'Animation': 242,
         'Biography': 293,
         'Comedy': 1872,
         'Crime': 889,
         'Documentary': 121,
         'Drama': 2594,
         'Family': 546,
         'Fantasy': 610,
         'Film-Noir': 6,
         'Game-Show': 1,
         'History': 207,
         'Horror': 565,
         'Music': 214,
         'Musical': 132,
         'Mystery': 500,
         'News': 3,
         'Reality-TV': 2,
         'Romance': 1107,
         'Sci-Fi': 616,
         'Short': 5,
         'Sport': 182,
         'Thriller': 1411,
         'War': 213,
         'Western': 97})
```



First off, we see that our worst offenders (overrepresented) happen to be Action, Adventure, Comedy, Drama, Romance, and Thriller genres. We also noticed that certain genres were more likely to be seen together than others (Action_Adventure, Romance_Comedy, Drama_Romance). This information was critical in helping us then decide how we wanted to think about our unique simplification for a movie genres.



Genre Configuration

Our discovery about the differing frequency of certain genres helped us to deal with the case of a movie being labelled with multiple genres. Often in both TMDb and IMDb, as many as six genres would be reported to a movie (especially popular ones, such as Avatar). Despite this overflow of information, we were led to believe that some genre types provided more useful information for distinguishing a movie than others did. Among

some of the genres reported for Avatar were Action, Adventure, Fantasy, and Sci-Fi. Because most movies (evidenced by genre frequency counts above) fell into the category of Action/Adventure, we decided to prioritize the presence of more niche (less reported) genres when presented with this information. Furthermore, we tended to give priority to genres such as Animation, Horror, and Sci-Fi rather than genres such as Adventure, Fantasy, and Thriller, which are seemingly more vague. This is an application of domain knowledge and it speaks to our desire to simplify such loosely categorized information into more specific, definitive categories. Additionally, we accepted the fact that some genre pairs were inseparable and to prioritize one genre over would destroy too much valuable information. These scenarios fully warranted the creation of a new genre category altogether. Such was the case for movies which were reported as Romance and Comedies, or Action and Drama, etc. Imposing this hierarchy of genres achieved the goal of intelligently simplifying the multi-label problem as well as the distribution imbalances due to over/under representation by prioritizing more niche genres and shedding off the more generic, over-utilized genre categories.

Traditional Statistical and ML Methods

For this project we decided to implement a Logistic Regression and a Random Forest. Both have pros and cons for this type of data. Logistic Regression is a simple yet effective traditional classification algorithm and Random Forests are non-linear, robust, and flexible. Certain drawbacks of Logistic Regression include the assumption of a linear decision boundary for our features. The drawbacks of Random Forests Classifiers are that they tend to not perform as well for high-dimensional and sparse data. This latter point was specifically a concern because we chose to implement a bag of words for our features, which are both quite high-dimensional as well as sparse.

Features: For both classification algorithms, we narrowed down our feature list (plot keywords, leading actor, director, budget, log_budget, content rating) to allow us to keep only the most useful of our initial features (plot keywords, leading actor, and director). The plot keywords were transformed into a bag of words, and instead of using actors and directors as categorical variables they were merged into the bag of words, as this way were were able to keep the dimensions of feature within a reasonable range. Overall model performance was initially gauged using a sample of around 1500 observations, and then later when details were finalized we decided to use the entire 4500+ observations.

TF-IDF bag of words: When crafting the bag of words, we decided to opt for the TF-IDF approach (term-frequency / inverse document frequency). This approach was ideal for our project because it automatically emphasizes words that occur frequently and de-emphasizes those that do not. We removed stop words because they were not useful and allowed the bag of words to account for level-2 n-grams (bigrams) so that word pairs would be considered. Consideration was given to stemming the bag of words, which would mean that words would be broken down to their stems, but it turned out that improvements were marginal at best when implemented and sometimes the strategy destroyed too much information. We thus decided to keep the full keywords in our bag of words.

Random Forest: Parameters for the number of estimators for this model were tuned using Grid Search. We then ran 10-Fold Cross Validation on the best parameters to calculate the model's prediction accuracy on the test set. Before cross validation and parameter tuning, the model's classification score was around 36%. After tuning, the model's classification score was around 39%.

Logistic Regression: 10-Fold Cross Validation was performed on the logistic model as it was for the Random Forest. The Logistic Regression model did not perform as well as the Random Forest, as after cross validation, the prediction accuracy on the test was consistently around 30%.

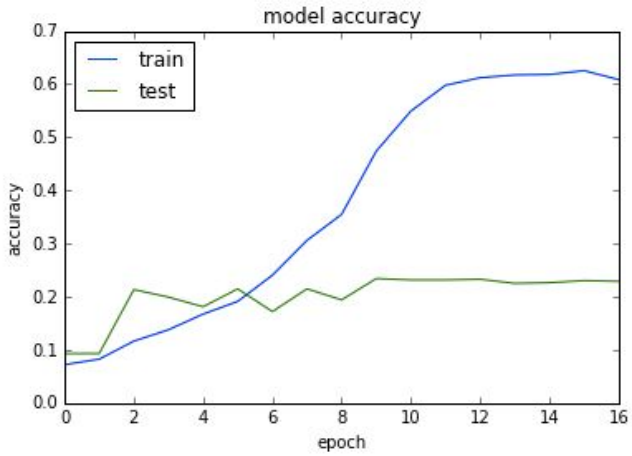
Discussion (Comparisons to Baseline): Overall, our traditional models both outperformed the baseline of predicting the most frequent genre for every movie (13% accuracy). While accuracy on the test set could be improved, our predictors and models are consistent and serve as a good start for further developing genre classification through supervised learning.

Deep Learning

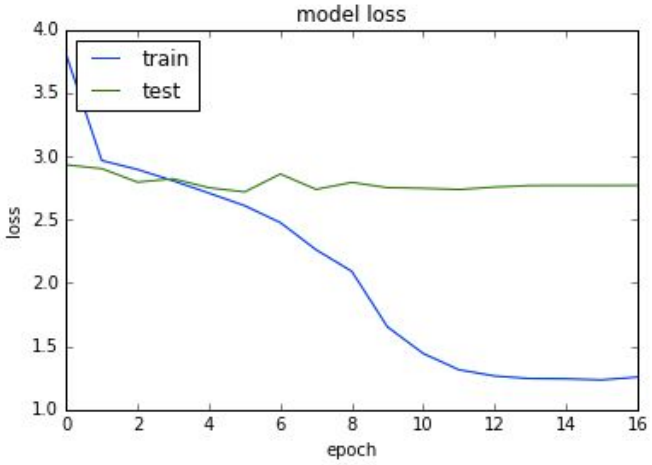
We tackled the challenge of identifying genre with movie posters by using two different Convolutional Neural Networks (CNN), one trained from scratch and one pre-trained. Deep learning, a method that excels with large amounts of data, was an appropriate match for this classification problem. On top of this, CNN provided us a great way to account for the nature of the image data. Images are made up of many structures that build upon themselves. Each data point, a pixel or other similar small structure, is thus inherently related to its neighbors. CNN takes this location relationship into account, which further helps to improve predictive accuracy.

Poster Characteristics: Most movies had a single movie poster associated with themselves in the TMDb database. A tiny minority had no movie poster associated with their title whatsoever; these movies were discarded from the dataset. Post-cleaning of the data, there were 4831 RGB images with dimensions transformed to be 185x185 pixels.

CNN From Scratch: The CNN model created from scratch used Keras with a TensorFlow backend and a GPU provided by Amazon Web Services (AWS). We used a ReLu activation function, max pooling, dropout regularization, and an Adadelta optimizer. The learning rate started at 0.1, but a callback function was used to reduce the rate by a factor of 0.2 whenever the validation loss plateaued. We also applied an early stop callback function to stop the fit whenever the validation loss did not decrease for more than

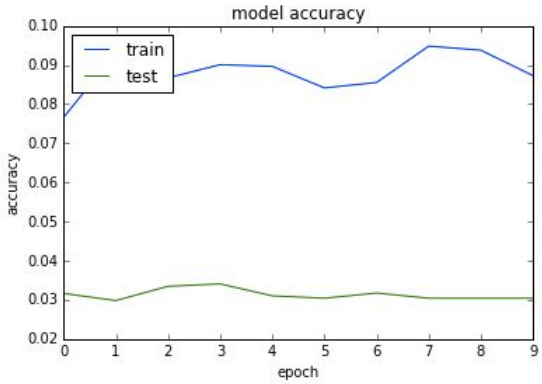


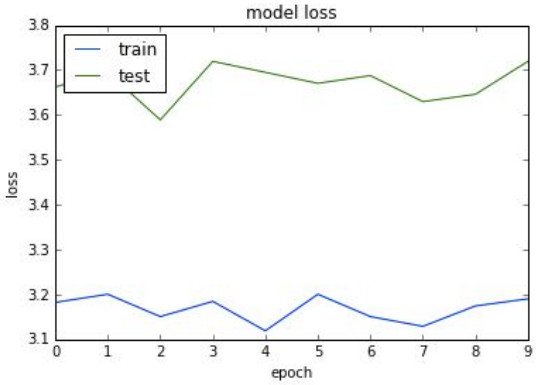
6 epochs. The resulting model was run on a training set of 3091, a validation set of 773, and a test set of 967. The resulting accuracy on the training set peaked at 60% accuracy on the training and 21% accuracy rate on test. Despite adding several layers and switching around different tuning methods, the accuracy peaked at this value. The answer for this seemed to be found in the loss values for both train and validation. In turns out that we simply didn't use enough data. When looking at the loss value for train, it is significantly smaller than the loss value to which the validation set converged(by about 4x when allowed to continue without early stopping). In this circumstance, our model will almost certainly improve with more data.



CNN Pre-Trained: We fine tuned the VGG16 CNN model that was pre-trained on data from ImageNet, an image database. ImageNet contains thousands of images relating to multiple sources and ideas. This pretrained data was testing our deep learning function's ability to still predict movie posters correctly when initial data inputted into the model did not pertain to movies at all. We added two models to the top of the network and froze the rest of the bottom layers.

The results of the pre-trained model can be seen below through the loss and accuracy graphs.





If you compare these results to the model we made from scratch, the results are significantly different. While the accuracy is low, the model loss appears to stay in the same position throughout all epochs as well. We can not attribute this error to data size as we did with the scratch model. One possible solution could be the addition of more layers into the model. We could also create variation by adjusting the amount of layers we freeze. This model needs more

adjustment compared to the one we made from scratch as the pre-trained data adds another element to which our deep learning must account.

Conclusions

Our goal was to determine whether it is possible to accurately and consistently infer a movie's genre using only the available metadata. Overall, we see that a movie's metadata does contain information that might possibly lead to more accurate genre classification in the future. Both Logistic Regression and Random Forests show that given this multi-class prediction problem, plot keywords, director, and leading actor information do contain valuable classifying information. This corresponds well with our domain knowledge that often certain directors direct certain genres and certain actors typically act in certain genres. It is not surprising that plot keywords contributed a significant portion of the model's predictive ability, as this is generally the purpose of a keyword. Surprisingly, movie poster images also contain a degree of predictive ability for certain genres. The deep learning implementation with the movie posters outperforms the baseline of 13%, though the traditional statistical methods outperform the deep learning approach. Extensions for the future would include combining the information from both sources and evaluating whether that would increase the overall classification accuracy. Future improvements could be found by pulling more observations for the deep learning approach as well as finding additional sources of metadata to scrape that is not contained in the TMDb or IMDb APIs. Considering the nature and sparsity of the metadata, classification accuracies of around 38% are informative as they highlight the ability to make meaningful connections from such data, and it would not be reasonable, given more comprehensive computing resources, that we may be able to pull enough data and metadata to provide even more consistent and reliable genre classification methods.