Milestone 5: Final Report[1]

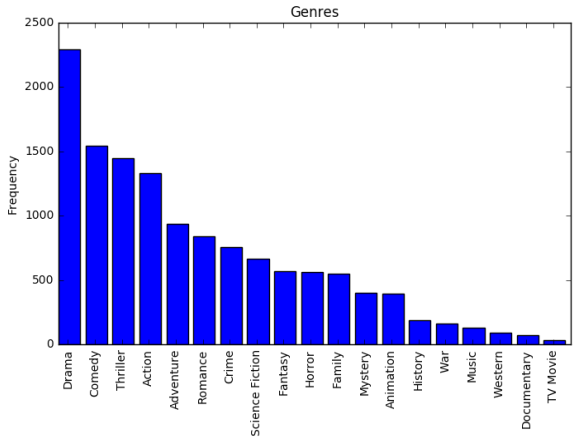
Kyle Honegger, Mark Hill, Joseph Reilly

Introduction

Our task for this project was to predict the genre(s) of a large number of movies utilizing two different approaches: traditional machine learning and deep learning. As movies can fit into more than one genre quite frequently (action-horror, romantic-comedy, etc.) we decided to structure this project as a multilabel classification task. We acquired quantitative and textual descriptors, as well as movie poster images, for a large number of movies and attempted to predict multiple genres for each movie using traditional machine learning (ML) algorithms and a deep learning approach. Our dataset included more than 150,000 observations and images, and while there certainly may be benefits to leveraging a dataset of this scope, we found that working with this amount of data presents a number of challenges, which we highlight in our conclusions. Throughout this report we compare the results of our multiple models and their advantages/disadvantages, and discuss potential future directions.

Data and Pre-Processing

Our data source was The Movie Database (TMDb) which contains information on roughly 300,000 movies. Since many older movies appear to be missing poster data, we downloaded information on all TMDb movies from 1940 through today, and considered only movies for which there was a poster image and at least one genre tag. We recorded certain features for each, retaining a core set of features representing what we expected to be the minimum number necessary to compare performance of our classical and deep learning models. To accelerate data gathering, we parallelized acquisition by distributing data download among 20 AWS EC2 instances and merging these batches together to form our training set.



Prior to processing our features, we had to address the large class imbalance between genres that is visualized to the left. Left unaddressed, our model would likely over-assign these more common classes while being highly inaccurate on the less represented classes. We collapsed our genres into seven broader categories:

[1] We would like to thank our TFs Rashmi, Jerry, Kela, Joseph, and Zona for their help and grading throughout this final project.

If genre label contains:

action, adventure, fantasy, sci-fi, thriller, western --> *action (Genre 1)*
crime, drama, mystery, music, history, war --> *drama (Genre 2)*
family, animation --> *family (Genre 3)*
comedy --> *comedy (Genre 4)*
romance --> *romance (Genre 5)*
documentary --> *documentary (Genre 6)*
horror --> *horror (Genre 7)*

These seven labels were based on Ivasic-Kos et al.'s work on movie classification from the machine learning literature (2014). This grouping results in at least one genre tag for every movie while eliminating niche genres that would be extremely difficult to predict at scale, such as "TV Movies". While still being a multi-label classification task, this re-categorizing of genres should improve predictive power of our model. "Documentary" remains vastly under-represented as a class but was retained because it often appears alone in movie descriptions.

Exploration of Quantitative and Textual Features

Features included in our dataset are: IMDB ID, genres, title, release date, plot overview, tagline, budget, revenue, popularity on TMDb, vote count, vote average, original language, production company, spoken language, run time, number of actors, number of crew, and whether the film is adult in nature. Categorical variables, which constitute the majority of our features, were encoded in our dataset as one-hot binary variables, and continuous quantitative predictors were encoded as floating-point numbers. When producing the one-hot encoding for production companies, we kept only the 100 most common companies because there were over 46,000 unique companies, which would have significantly increased the dimensionality of the dataset, resulting in increased time to fit each model.

During our exploratory data analysis (EDA) phase, we made several interesting observations about how these variables relate to movie genres. Certain budget and revenue sizes are more typical of certain genres, e.g. Hollywood summer blockbuster versus artistic ballet documentary. Different genres are typically rated higher or lower, with drama typically on the high end and action on the lower side of the spectrum. Certain production countries and companies specialize in specific types of movie so this could potentially be a strong predictor of certain genres. Later, we present results on the effect of removing some of these predictors and discuss the relative importance of their inclusion in our models. We also discuss the use of principal components analysis for dimensionality reduction to speed the model fitting process and improve performance.

To incorporate potentially valuable textual data into our feature set, we used Latent Dirichlet Allocation (Blei, Ng, and Jordan, 2003) to embed the text from plot overviews in a 20-dimensional space, where each dimension represents a latent "topic" variable. This technique works by modeling each observation as a mixture of unobserved "topics," and assigns a mixture proportion for each of these inferred topics to every movie. We used these proportions directly as additional quantitative predictors for each movie.

Image Data

The poster images retrieved from TMDb had irregular sizes and formats, so all images were resized to a standard size of 300 x 185 pixels before storage. Prior to use, the images were standardized to have a sample-wise mean of 0 and standard deviation of 1. We excluded grayscale images, and any other image formats that could not be converted to RGB. In addition to sample-wise standardization, we also tried a more “natural” form of normalization by converting images to HSV and normalizing the “Value” channel before converting back to RGB.

After removing films with missing features and unsuitable poster images, our final dataset contained 533 features across 133,851 movies. An archived version of this dataset (final_project_data.zip) is available for download here: http://bit.ly/2pxFPxr. We also created a second version of the dataset with erroneous values (e.g. a budget of $0.00) replaced by the mean budget across movies, which we then compared to the non-altered dataset.

Traditional ML Methods

We reserved 33% of our data for testing and used the remainder as a training set. We selected three different models to fit our data that were well-suited to multi-label classification tasks: Naive Bayes (NB), logistic regression (LR), and Random Forest classifier (RFC). NB and LR used a one-vs.-rest classification approach, essentially modeling each genre as the output of a single classifier, since it requires only fitting one classifier per genre. While not the most powerful approach, it was more efficient than other classification schemes and we expected it to give decent initial results. To handle the remaining class imbalance, we adjusted the weights to be inversely proportional to the class frequencies for the LR and RFC, and set the priors in NB to the observed frequencies. We chose Hamming loss as our performance metric because it gives the label-by-label accuracy for our multi-label classification, and not an "all or nothing" accuracy indicating whether we got the exact combination of genres right for a particular movie. It is easily interpretable and appropriate for all three models.

Naive Bayes

The Naive Bayes model achieves a Hamming loss of 0.31 prior to optimization. This means that roughly 69% of our genre labels are predicted correctly for the test set. Since each genre assignment is a 50/50 choice, this would seem to be a large improvement over a random classifier. We calibrated the class probabilities to explore how varying the class threshold would affect our loss on the test set. Upon doing this, the Hamming loss improved to 0.213, but only because the model was assigning 0 to all genre labels. This is clearly not the desired effect to answer our research question. We could try to impose a constraint on the model that at least one genre is always chosen, e.g. applying a softmax transformation, but this assumes labels are mutually exclusive and is thus incompatible with multi-label scenarios like this one. To circumvent these issues, we turned to logistic regression.

Logistic Regression

We utilized stochastic gradient descent to efficiently fit a logistic regression model to the large training dataset. We again used a one-vs.-rest scheme and the Hamming loss performance metric. The model was tuned via cross-validation and C=100 was identified as the optimal regularization parameter value. While the Hamming loss of 0.350 with this classifier is worse than the 0.215 achieved with NB, it does at least assign movies at least one genre. Presumably, performance would improve with some finer tuning of the regularization strength.

We implemented PCA and kept the minimum number of PCs required to explain 90% of the total variance ($p = 327$) to see if we could improve performance by reducing the dimensionality of the feature space. We hoped to make the model fitting faster and improve the quality of our predictions. The resulting Hamming loss of 0.357 is slightly worse than that achieved using the full data set, indicating that the features aren't very compressible, i.e. information is distributed widely across features. Accuracy by genre is fairly similar across action, drama, comedy, and horror, while family and documentary genres are predicted with slightly higher fidelity.

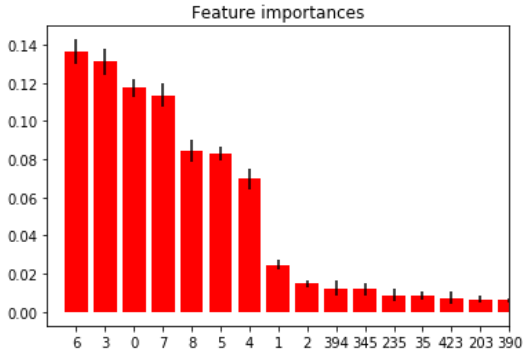
Random Forest

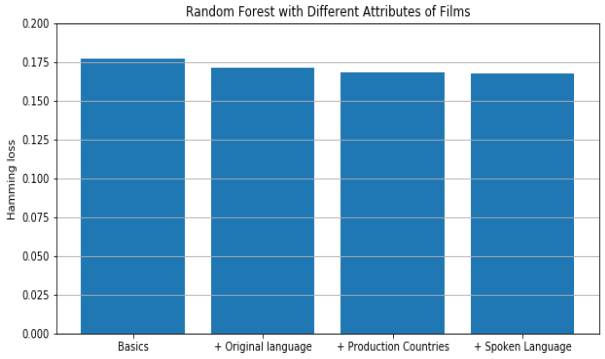
After tuning the number of estimators and max depth via cross-validation, we achieved a Hamming loss of 0.168 via the RFC. This means that roughly 83% of our genre labels are predicted correctly for the test set. We also compared the Hamming score metric with a more traditional overall accuracy metric. While interesting to compare to the Hamming loss, this gives us a much coarser picture of how our model is performing at assigning genres. 29% of our films' genres are predicted exactly as we assigned them. Since a random classifier would need to assign seven binary genres correctly to get one movie correct on this harsh metric, each movie would have a 0.5^7 or roughly 0.8% chance of being assigned correctly. We fit this model on our data that had been reduced by PCA and saw only a 1% decrease in performance. Depending on the goals and needs of a project, this slight loss in accuracy may be worth it for a greatly increased computation speed. Our attempts to impute erroneous feature values led to a 2% decrease in accuracy. Future work could explore other methods of handling this issue.

This model correctly identified 4 of our 7 combined genres almost 90% of the time, while action and comedy are accurately labeled roughly 75% of the time. Drama is the only genre with a lower than 70% accuracy rating, but it appears with a wide variety of other film types, thus making its discrimination more difficult. We also explored feature importance to our model as well as the effect of adding such a large number of dummy variables as features.

As we can see from the below visualization, seven features proved the most important in determining the movie genres. The top nine contributors came from our "basic" core of quantitative data scraped from TMDb and are, in order: running time, popularity, release year, # of actors, # of crew, vote average, and vote count. Budget and revenue come last among these variables and suffer from having a large amount of missing data compared to our other features. Adding all our

language and country predictors as indicator variables leads to a 1% improvement in classifier performance. Whether this small gain is worth the computational cost is debatable.

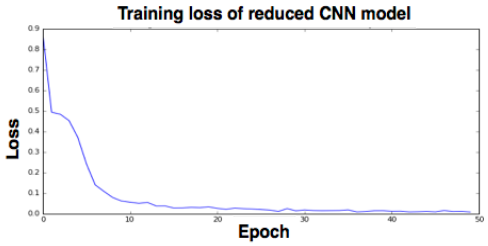


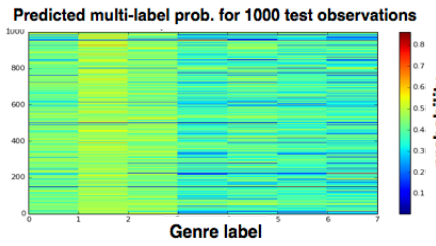


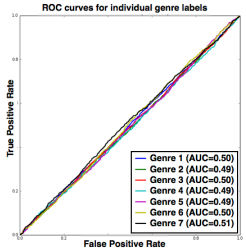
Deep Learning

Original Model

We implemented a convolutional neural network adapted from the Keras "VGG-like convnet" tutorial. This is essentially a much simplified version of the pre-trained VGG-16 model that we fine-tuned. The network consists of two 32-unit convolution layers with ReLu activation, followed by a max pooling layer to reduce layer size by one half, a repeat of this motif with a 64-unit convolution layer, and then two fully connected layers - the first with 256 units and ReLU activation and the 7-unit output layer with sigmoidal activation. We used dropout to regularize the model and trained using a binary cross-entropy loss function and the RMSprop optimizer. We tried a number of other optimizers, including SGD w/momentum, Adam, and Nadam - none of which showed much difference in performance and were all slightly slower to learn than RMSprop. We tested several learning rates for the optimizers, but they performed similarly across a range of values. We first trained this model on a small set of 512 images to make sure it worked well, then moved on to training this model on sets of 10,000 images. We tuned the learning rate for a set of 10,000 training images first using 10 epochs for each different learning rate for the sake of time. Once a suitable learning rate was found, we then trained the model with 50 epochs (below) to see how the cross-entropy loss decreased as the number of epochs increased (left). We found that this model predicted multiple genres across and within test

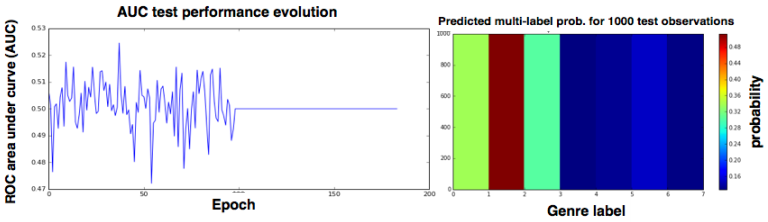






observations (center). However, an examination of the ROC curve and AUC values for the trained model revealed that model did not predict better than chance (AUC=0.5, right), indicating

the model had significantly overfit the training data. These results were disappointing, so we tried altering a number of parameters, including network size (up to 64-unit convolution layer), method of image standardization (HSV normalization), length of training (20k steps), and test set size (full 60k image training set). None of these improved performance (right). Surprisingly, we observed that the augmented model initially began to predict with AUC above 0.5 on average, but after about 100 epochs (10K update steps) it converged to 0.5 and stayed there (left). When we examine the genre predictions made by this model (right) we can see that it learned to predict the same genres for every movie, with probability proportional to the class probability of the test set. This behavior was robust to a number of other parameters, e.g. optimizer, learning rate, and data sampling. Ideally, we could have attempted different network architectures, regularization methods, and training schemes, but we ultimately ran out of time because of the long training time required on a dataset this size. These results were disappointing, but they highlight the difficulties in training a large deep network and illustrate the importance of cross-validating a model.



Pre-Trained Model

Our pre-trained network is a variation on the VGG16, a 16-layer network used to good effect in the ILSVRC-2014 competition (Simonyan & Zisserman, 2014). Weights were pre-trained on ImageNet, the input was specified to the same 300 x 185 x 3 size we used in our original model, and the same binary cross-entropy loss function was used to train and evaluate performance. Fully connected layers were added at the end to specify our 7 desired output labels. Maximum binary accuracy approached 80%, but this was because the model had learned to return only zeros for its predictions, much as our Naive Bayes model did. This model did not perform as well as we had expected, given its extensive preexisting natural image exposure, but additional modifications may make it possible to improve performance.

Conclusion

Of the traditional machine learning models we examined, tree-based classifiers appear to perform best. Our Random Forest model avoided over-fitting our training data and had the highest predictive accuracy according to our performance metric. Our deep learning model, however, suffered from overfitting despite dropout regularization. While our original deep learning model did not fall into the trap of predicting 0 for all labels, it nonetheless predicted the same probability for all genres for all movies. Future work could include exploring alternative methods for regularizing neural networks, comparing bagging algorithms like RFC to boosting algorithms like AdaBoost, and attempting to refine other pre-trained neural nets for use in our classification task.

Appendix

All notebooks can be found at https://github.com/CS109b/movie-genres

Screencast can be viewed at https://youtu.be/T4w-bONg7JE

References

Blei, D.M., Ng, A.Y., and Jordan, M.I. (2003) "Latent dirichlet allocation." *Journal of Machine Learning Research* (3): 993-1022.

Ivasic-Kos, M., Pobar, M., & Mikec, L. (2014). Movie posters classification into genres based on low-level features. In *Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2014 37th International Convention on (pp. 1198-1203). IEEE.

Simonyan, K. & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556