

Group 8 DSCI445 Paper DRAFT

Group 8

Kelsey Britton · James Chinnery · Robin Thrush · Kaitlynn Walston

Introduction

Our group chose to work with movie data and our goal is to build a predictive model that predicts the gross revenue that a movie makes. The dataset was pulled from IMDB and is filled with movies made in 1986 to 2016. Some variables included in this dataset are budget, star actor, director, rating, and genre. To make the data and interpretation more digestible, each of us first worked with three predictor variables and saw how they each affected the gross of the movie. Then, we came together and discussed the results and what changes we had to make to the data to build a model. We combined the ways we changed the data to have a final cleaned dataframe. We then each took on a different method that would inform us of which predictors are the most influential to predicting a movie's gross revenue using that final dataframe and discussed the results we got.

Methods

Kelsey:

The variables that I cleaned were release date and movie title. The original release date column had multiple inconsistent format types, so I extracted just the month as a numerical value. I also used this value to assign each entry a release season (March, April, May as Spring, June, July, August as Fall, etc.). For the movie's title I used sentiment analysis to assign each entry a sentiment score. This was done by combining AFINN word sentiment with some additional scoring adjustments to account for punctuation to allow titles with "?" and "!" to reflect a stronger emotional tone in either direction.

James:

Budget was the main column I worked on. Since over a third of the data was missing I chose to flag it and fill it with the median value for that year. In theory this should allow the models to still use the data from those entries and simply trust the budget less. For the runtime of the movies it was numeric and well behaved so it was left alone. The writers were initially lumped into 5 categories. This was somewhat arbitrary though and a better approach was to lump into two categories since so many writers only have one movie then this threshold could be tuned though I this was changed for some of the methods as they were better equipped to handle target encoded data

Robin:

The variables I worked with were director, company, and country. They are all characters, and unfortunately too unique to rely on the standard dummy encoding. To navigate this, I decided to

assign countries to regions, and allow for regions to be dummy encoded because there were way less unique values. When doing this the function automatically made “Africa” the baseline region which caused the other regions to have an unusual standard for the regions and their influence on gross. Africa only had 10 movies associated with it, and all the movies had relatively high gross, and so to fix this I changed the baseline region to be North America which had 5040 movies associated with it, and a much wider range for gross. For the director and company I decided to use frequency encoding which allowed for the data to be digestible for a linear regression model. The caveat with using frequency encoding is that if there are directors that have the same frequency, they will be treated identically, despite being associated with different movie revenues. I transformed the data using log1p() which was supposed to help with dealing with skew, especially with the variables that had an extremely large range (gross, budget, etc). I fit a full model with the director, company, and region. When log transforming data, you interpret the results as how the percent change in the variable relates to the percent change in gross. So for example a 1% increase in director frequency is associated with a 0.807% increase in gross with all other things held constant. The full model with these three variables revealed that director seemed to have more influence than the company or region. I also fit a residual vs fitted plot and QQ plot which revealed that non-constant variance and that the model seemed to be more consistent in predicting high-gross movies than mid to low gross movies. QQ plot mostly normal, left tail was slightly curved, so some non-normality was going on. Overall, okay model with just the three variables, but likely could be improved with the inclusion of other predictors.

Kaitlynn:

The variables I worked with were genre, rating, and stars. Since they were all categorical, I started out by creating frequency tables to visually see how much each category occurs. After that I did a chi-sq test and contingency function to compare the frequencies and see if there was any high correlation. Then, I created a LASSO model to try to predict the gross revenue and also shrink the coefficients that are not as impactful. Using the LASSO will help us identify which of the variables have the strongest influence on gross revenue while reducing the noise from the other, less impactful predictors.

Together:

All of these cleaning methods were combined into a single csv file for use in the modeling.

Results:

Kelsey: Pure Linear Regression

For the pure linear regression model, all variables were standardized and log-transformed for comparability. The model explained 59% of the variation in log-transformed gross revenue ($R^2 = 0.594$) and the RMSE was 1.56. Among the variables, the vote count on IMDb, the frequency at which the production company popularity (denoted by the frequency they appeared in the data set), star popularity, and runtime were all highly significant positive predictors. The genre being Drama, the genre being Crime, and the IMDb score had the most significant negative effect on gross income. In total, there were 23 significant predictors within the model, encompassing notoriety, runtime, IMDb score, region, genre, region, budget, release season and rating. Overall, the linear regression model identifies strong, interpretable relationships between some movie attributes and their gross income.

Kaitlynn: LASSO

Robin: Elastic Net

After some group data cleaning, we were able to have 1 comprehensible dataframe with all the predictors we would need to run shrinkage and best predictor models. At the end of the cleaning we were still left with 4 columns that were character values: genre, rating, region, and season. To handle this I relied on dummy encoding, hoping the model would select the best reference category. The rest being numerical data, I log transformed ($\log(\text{variable} + 1)$ so zero was not an issue). This was hoping that skew would be more controlled. The process to build the net model was using a recipe, letting R choose the possibilities for the lambda value, and constraining alpha to between 0 and 1. Then using cross validation with the end goal to find the best lambda and alpha that resulted in the lowest RMSE. Fitting the final model, the RMSE was 1.56 and R^2 was 0.59. To see how far off predictions were I divided 1.56 by the mean of the original gross that was log transformed. This told us that the predictions from this elastic net were about 10% off. (Need to add the “best coefficients/predictors” that elastic net found, residuals vs fitted, and QQplot)

James: Tree Boosted/ Bagged Random

The nominal variables were split into two classes and each was handled differently. For those with low cardinality, namely the release month, genre, rating all had < 15 unique columns. These columns were traditionally one hot encoded. For the rest I decided to forgo the binning initially used as to allow the trees more granularity on where to make the splits and use target encoding. This was because many of the categories like writer and director had cardinality in excess of 2500 and one hot encoding would make the feature space rather spares. Instead I used target encoding with a smoothing parameter to reduce overfitting concerns. This kept the data dense and hopefully imparted some meaning into the value used to represent it. It should help differentiate between people/companies who truly made high earning movies.

For the models themselves the random forest was tuned on the number of predictors per tree and the number of points required for a split. Tuning was done with bayesian optimization over 20 iterations. The boosted tree was similar except it was tuned on learning rate max depth and the number of points required to split it was trained using the same method. Overall the approach with these trees was to preserve as much data as possible and let them do their magic. Though our dataset isn’t massive it is large enough that CV should keep us from overfitting too badly. The importance of log transformation was also clear with the rmse converted back to real dollars was nearly twenty million lower.

The tree based methods achieved an rsq of .508 and a rmse of roughly 40 million in real dollars. Its important to not place too much importance on the rmse here though. By nature the biggest movies will dwarf all the others in terms of revenue which manifests as highly left skewed which has a large impact on the mean. Despite this rsme was used as the metric for two reasons. It is a standard for most regression problems due to its smooth characteristics and in an environment like movies theres far more theoretical upside to predicting the next big hit as opposed the next middle of the pack title. The direct result of that is an rmse that is highly influenced by the massive outliers at the top even after the log transformation which makes the number itself somewhat unreliable. Instead one can focus on the rsq which says that 50% variance is explained by our model. Its not amazing but for financial data rather respectable for an ensemble model. The reason I mention that ensemble models as a class in general is because the struggle with monotonic relationships seen in financial data. Its a reasonable expectation that spending more on a movie should increase revenue. This

isn't always the case but the general idea it better captured by linear modeling approaches. This is seen in the superior performance of the elastic net