Predicting Movie Success Based on Pre-Released Data

Manoj Kumar
Shrenik Shah
Anna Khazan

December 29, 2015
Intro to Data Science

Business Understanding

The film industry is predicted to generate 39.1 billion dollars in box office revenue in 2015, with continued growth of up to nearly 50 billion dollars by 2020. In the past, studios have focused on relying on only a handful of very expensive movies every year to make sure they remain profitable. It is estimated that 80% of the industry's profits over the last decade was generated by just 6% of the films released; 78% of movies lost money over the same time period. These blockbuster movies emphasize the spectacular: casting as much star power as possible and pairing it with high production value. The result of this is a sky-rocketing budget. It is estimated that the average movie now costs $100.3 million after including production and marketing expenses (Movie Budgets). Film production exhibits a great deal of financial risk in an age where blockbuster movies can have budgets in excess of hundreds of millions of dollars. This risk could be reduced to an extent with the use of data science methods. Since movie scripts and projects are pitched and budgets are set and spent well before any revenue is ever made, we plan to construct a predictive model using features that are generally known before the movie release date.

The movie's success depends on the preferences of the movie-goers. If viewers do not choose to see a movie, millions of dollars can be lost, causing production companies and/or producers to go bankrupt. Therefore, the entertainment business is a serious and high-stakes process where there are high risks and rewards. In 2015, the average production costs for the top 50 grossing movies were $133 million (Box Office Mojo). For production companies, it is very important to predict what movie is going to be a success prior to making the decision of investing millions of dollars into the project. This project focuses the determinant factors of a movie’s success during the production process. The goal of our project is to better quantify the elements that are significant determinants when a making the decision to fund a production.

A movie is created in three key phases: production, distribution, and exhibition. The production phase is where the movie idea is translated into a finished product. It is during the production phase where the decision to adequately fund a motion picture is determined. In this phase several decisions can be made such as casting, genre, MPAA rating, etc. Similar to any manufacturing process, this phase is the most critical step in ensuring that the consumer’s needs, wants and/or specifications are met. With a motion picture production, the ability to design the script with the right actors, genre and adequate theatrical resources are all the elements that have been shown in prior research to ultimately determine a movie’s success.

There are many research papers that explore the production and post-production determinants of motion picture box office revenue (Deniz and Hasbrouck). This research shows that the pre-release factors that influence the box office revenue, in no particular order, are:

1. Movie genre (science fiction, drama, action-adventure, comedy, and musical)
2. MPAA rating (G, PG, PG-13, R and NC-17)
3. Release date (Christmas, Summer).
4. Star Power
5. Production Costs (Budget)

There are many other factors that contribute to a box office success after the movie is released, such as critical reviews by known critiques and word of mouth publicity and reviews of peers on social networking websites (Kennedy). However, the decision to fund a film is determined at a very early stage in the process when we don’t yet have information about critical reviews or social networking trends. Our objective is to facilitate the decision-making process of the producer or studio that must decide if a film is a viable investment, defined as a movie that will bring a satisfactory return of investment based on revenues that are generated on the opening weekend collections. Nearly 40% of the total box-office revenue for any particular film is earned in the first week of its release (Topf). By examining only the pre-release factors such as actors, directors, scriptwriters, production studio, budget, genre, planned release date and MPAA ratings that the director is targeting, we can get a better idea of what drives box office sales and a more direct picture of what are the important factors in determining if a proposal is worthy of investment.

Data Understanding

The dataset we used to understand how these factors have affected revenue in the past consisted of several thousand movies, where every instance or example in the dataset corresponded to a movie. For the target variable. we ruled out several revenue metrics, such as gross domestic revenue and gross worldwide revenue. This helped ensure that movies that were released several years ago did not appear more successful than movies that were just recently released simply because there has been more time for the earlier movies’ revenue to accrue. We went through three cycles in modeling the target variable.

Cycle 1

Target variable was modeled to be the opening weekend revenue, normalized by dividing by its standard deviation. This is a regression problem since the target variable is continuous.

Cycle 2

Our movie dataset spanned 90 years, with release dates between 1920 and 2011. When considering this as a regression problem, we planned to account for inflation by estimating what each movie's budget and opening weekend revenue would be equivalent to in 2015 (CPI Inflation Calculator). After doing this we again normalized it by dividing by its standard deviation.

Cycle 3

We then decided to look at ROI statistics for the opening weekend - opening weekend revenue divided by its budget. By dividing the total opening weekend revenue by the total budget, we found that an average movie will generate 45% of the budget in the opening weekend. We decided to use out target variable to see how each particular movie did compared to this metric for an average movie.

We chose two ROI thresholds, 0.45 and 0.6, and binarized the ROI values by giving it a label of 1 if the ROI is greater than the threshold and 0 otherwise. This is useful because it gives the producer the flexibility to set the threshold. If it is a new experimental movie he can set a low threshold and if he definitely wants the movie to be a blockbuster, he can set a really high threshold. In this way, our new model becomes independent of inflation.

Feature attributes

For each movie, our feature set consists of the following attributes.

Continuous attributes

a. Production budget
b. Production company weight, explained further in next section

Discrete attributes

a. Top Director

b. Top Writer 1

c. Top Writer 2

d. Top Actor 1

e. Top Actor 2

f. Top Actor 3

g. Genres [Sci-Fi, Crime, Romance, Animation, Music, Comedy, War, Horror, Film-Noir, Adventure, Thriller, Western, Mystery, Drama, Action, Documentary, History, Family, Fantasy, Sport, and Biography]

h. G, NC-17, PG, PG-13, R

i. Release Date during the summer

j. Release Date during winter holidays

Data Preparation

To start, we needed to acquire a comprehensive dataset of movies that have already been released. We identified a list of IMDb movie ID's through the MovieLens database (MovieLens), compiled by GroupLens Research. The full list included 8,927 ID's that, when appended to an IMDb (IMDb) url, linked to an IMDb webpage that listed nearly all of our movie features. We scraped the html of each IMDb webpage to find the attributes listed above. We decided to only include those records that included a budget and opening weekend revenue in USD on IMDb, had a budget of over $5,000 and recorded opening weekend revenue of over $5,000, which reduced the dataset to 2,210 records.

We then converted these features into meaningful information to our analysis. In trying to interpret the significance of a particular release date, we decided that this can be treated as a binary variable. We defined peak season as beginning on May 1 and ending on August 31, so any movie that was released between these two dates received a value of 1, while any movie that was produced during the off season received a value of 0 (Miyamoto). The holiday season is another factor that will significantly impact opening weekend revenue, so we created a separate feature to indicate this. We created another feature vector to assign a value of 1 to any movie that was released between December 21 and January 1, to indicate Christmas season and a value of 0 to all other release dates. We identified a list of 250 top directors, and assigned each movie director a binary value of 1 or 0, whether the main director is considered a 'top director' or not (The 1,000 Greatest Films (Top 250 Directors)). Using the same logic for top screenplay writers and actors, we used a list of 100 top writers and 50 top actors to determine if and how many of these writers and actors worked on any particular movie (Top 50 Popular Hollywood Actors and Actresses), (Hamell). Again, we coded this as a binary value in our input dataset. This naively assumes that every actor, every writer and every top director in this list is equally important.

Every movie webpage listed up to three genres out of 21 possible genres: Sci-Fi, Crime, Romance, Animation, Music, Comedy, War, Horror, Film-Noir, Adventure, Thriller, Western, Mystery, Drama, Action, Documentary, History, Family, Fantasy, Sport, and Biography, and one rating out of five possible ratings: G, PG, PG-13, R, and NC-17.

We normalized the three continuous variables (budget, opening weekend revenue, and average revenue per movie release for each production company) in our dataset by dividing each value by the feature's standard deviation. When considering production companies and the impact they will have on a particular movie's revenue, we used a data source that listed the gross box office revenue, along with the number of movies each production company has released. To convert this data to a more usable format, for each company we divided the gross revenue by the total movie count to get the average revenue per movie. We then considered the top 425 production companies that released more than 4 movies in total and gave each a weight equal to the percentage of the total average revenue per movie across all companies that it contributes (Movie Production Companies).

Our final dataset had an average budget of $34,319,764 with a standard deviation of $36, 369,884, and an average opening weekend revenue of $12,120,026 with a standard deviation of $16,833,403, not accounting for inflation. The most common movie rating was R, with 848 movie samples, followed by PG-13, with 669 movie samples. The most common movie genre was Drama, with 1,071 movie samples. Comedy and Action were also highly prevalent in the dataset, with 868 and 536 records, respectively. The least common movie genres in the dataset with Documentaries, Westerns, and Film-Noir.

Modeling

In this section we describe the machine learning algorithms used, the intuition behind why each of the algorithm used and the parameters used to tune each model. From the various parameters of each model, the parameter that gave the best mean AUC-ROC score across 3 splits was chosen.

Note that if overfitting is specified in any of the model descriptions, it is nullified to an extent by choosing the best model obtained from cross-validation.

Cycle 1

Regression models with the target variable not taking into account inflation.

<u>**Linear Models**</u>

Depending on the data, it is likely that a simple linear model might be a good fit. However, with linear models there is a strong bias that the features are linearly correlated with the target variable.

1. **ElasticNet**

The ElasticNet model combines the zeroing effect of unimportant features using the l1_ratio parameter and the balance between the convexity of the problem and the zeroing effect by the alpha parameter. In this way even if the coefficients are not directly interpretable, we can easily figure out what the least important features because they will have a value of 0.0.

Tuning the max_iter parameter enables us to get a balance between complexity and generalization of the model. The lower the value of max_iter, the earlier a model will stop training; if the value of max_iter is higher, the model trains until convergence (overfits), thus affecting the ability of the model to generalize to new data.

Parameters searched

a] alpha = [0.0001, 0.001, 0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]

b] l1_ratio = [0.0001, 0.001, 0.01, 0.1, 0.2, 0.3, 0.4, 0.6, 0.7, 0.8, 0.9, 1.0]

c] max_iter = [1, 5, 10, 100, 500, 1000]

2. **LinearSVR**

The benefits of a LinearSVR model is that classification decisions are less affected by outliers. Tuning the C parameter enables us to achieve a balance between the complexity and generalization of the model.

a] C = [1, 5, 10, 100, 500, 1000, 5000, 10000]

<u>**Tree-Based Models**</u>

1. **DecisionTreeRegressor**

The pros of using decision trees are that both categorical and continuous features can be handled without doing any sort of preprocessing on the data. Also the feature_importances_ attribute can be directly interpreted to show the importances of the different features unlike linear models.

An important disadvantage of trees is that they have a tendency to capture random noise and tend to overfit. The complexity of each tree can be controlled by the max_depth, min_samples_split and the min_samples_leaf parameters.

a] max_depth = [1, 5,7,9,11,13,10,15,20,30,40,50]

b] min_samples_split = [1, 50,60,70,80,90,100,110,120,130,140, 150, 160, 1489],

c] min_samples_leaf': [1, 10,20,30,40,50,60,70,80,90,100,110,120,125,130,140, 745]

2. **RandomForestRegressor**

RandomForestRegressor fit "n_estimators" number of decision trees on the model and averages the predictions across all decision trees. For each decision tree, the data is bootstrapped which prevents overfitting.

With n_estimators there is a trade-off between the time the model takes to train and the generalization performance. The importance of each feature can be directly obtained by the features_importance_ attribute as well.

a] max_depth = [1, 5,7,9,11,13,10,15,20,30,40,50]

b] min_samples_split = [1, 50,60,70,80,90,100,110,120,130,140, 150, 160, 1489],

c] min_samples_leaf = [1, 10,20,30,40,50,60,70,80,90,100,110,120,125,130,140, 745]

d] n_estimators = np.arange(1, 20)

3. **GradientBoostingRegressor**

Gradient Boosting as compared to RandomForests is a sequential algorithm where intuitively in each stage (where the number of stages are controlled by the n_estimators parameter), more importance while training a decision tree is given to the mistakes made by the previous fit decision tree. This importance is tuned by the learning_rate parameter. Similar to RandomForests and DecisionTreeClassifier the importance of each feature can be got from the feature_importances_ attribute.

a] max_depth = [1, 5,7,9,11,13,10,15,20,30,40,50]

b] min_samples_split = [1, 50,60,70,80,90,100,110,120,130,140, 150, 160, 1489],

c] min_samples_leaf': [1, 10,20,30,40,50,60,70,80,90,100,110,120,125,130,140, 745]

d] n_estimators = np.arange(1, 20)

e] learning_rate': [0.1, 0.3, 0.5, 0.7, 0.9]

<u>Ensemble models</u>

The BaggingRegressor uses the same intuition (i.e fit each estimator on a bootstrapped version of the data) as the RandomForests but the base estimator here can be used any model.

Here we use the LinearSVR and the ElasticNet models as the base estimators.

1. **BaggingRegressor with ElasticNet**

a] l1_ratios = [0.1, 0.2, 0.3, 0.4, 0.5, 0.7, 0.9]

b] alphas = [0.0001, 0.001, 0.1, 1., 10.]

c] n_estimators = np.arange(1, 6)

d] max_samples = 893, 100, 10, 200, 220

2. **BaggingRegressor with LinearSVR**

a] C= [1, 10, 50, 100, 1000, 10000]

b] n_estimators = np.arange(1, 6)

c] max_samples = [893, 100, 10, 200, 220]

KNeighborsRegressor

An advantage of using KNeighborsRegressor is that we can use the kneighbors_graph method to get the movies that are very similar to each other with minimal extra effort.

The major disadvantage of KNeighborsRegressor is that each time we would like to predict the revenue of a new movie it would take a lot of time. Also since the features are a mixture of continuous and categorical the distance or similarity metric to be used is not clearly defined.

For the similarity measures we chose to tune across the euclidean and manhattan distance metric To obtain the probabilistic estimates we tuned across the two variants (the "uniform" version which gives neighbor the same weight and the "distance" version which weights each neighbor by the inverse of its distance)

a] n_neighbors = [5, 10, 11, 20, 100, 200, 500] b] weights ['uniform', 'distance']

c] metric':['euclidean', 'manhattan']

SVD with BaggingRegressor (ElasticNet)

We used a pipeline with the output of the SVD supplied as input to the second model (which in this case is BaggingRegressor with the ElasticNet model as the base estimator).

The number of components that SVD reduces the input feature size to is tuned by n_components.

Though it is unlikely to give a higher score on the test data due to the loss of importance by reducing the features, it will reduce the train time in most cases since the dimensionality of the input to the second model is much lesser.

a] n_components = [2, 3, 5]

b] n_estimators = np.arange(1, 6)

c] max_samples = [893, 100, 10, 200, 2200]

d] base_estimator = ElasticNet

l1_ratios = [0.1, 0.2, 0.3, 0.4, 0.5, 0.7, 0.9]

alphas = [0.0001, 0.001, 0.1, 1., 10.]

Cycle 2

Regression models with the target taking into account inflation.

We trained 3 models in this case, as we noticed the r2_score obtained from these models was much lower than the corresponding r2_scores obtained without taking into account inflation.

1.DecisionTreeRegressor

a] max_depth: [1, 5,7,9,11,13,10,15,20,30,40,50]

b] min_samples_split: [1, 50,60,70,80,90,100,110,120,130,140, 150, 160] c]min_samples_leaf: [1, 10,20,30,40,50,60,70,80,90,100,110,120,125,130,140]

2. Linear regression

3. ElasticNet with alpha=0.1 and l1_ratio=0.0001

Cycle 3

Converting the opening weekend target variables to binary labels depending upon the ratio with the budget.

The intuition behind using each model is the same as explained in training the regression models, except that the problem is now classification.

<u>**Tree based models**</u>

1.**DecisionTreeClassifier**.

max_depth = [1, 5,6,7,9,11,13,10,15,20,25,30,40,50]

min_samples_split = [2,3,4,5,8,10,12,14,16,18, 20,30, 1489]

min_samples_leaf = [1, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 20,30,40,50,60,66,67,68,70,71,72,73,75,80,90,100,110,120,125,130,140, 745]

2. RandomForestClassifier

n_estimators = [1, 10,12,13,15,20,25,30]

max_depth = [1,5,10,15,18, 20,22,30,35,38,40,42,50,80]

min_samples_split = [1, 20,30,35,38,40,42,45,50,60,70, 100, 1489]

min_samples_leaf = [2,4,5,10,15,17,20,30, 50, 100, 745]

3. GradientBoostingClassifier

learning_rate = [0.1, 0.3, 0.5, 0.7, 0.9] max_depth = [1, 5, 10, 20, 30]

n_estimators = [5, 20, 30, 40, 50],

min_samples_leaf = [745, 500, 200, 100, 50, 10, 5],

min_samples_split': [1489, 1000, 500, 200, 100, 50, 10, 5]}

Neighbors-based models

n_neighbors = [2, 5, 10, 20, 30, 60, 70, 80, 90, 100, 110, 200, 500],

weight = ['uniform', 'distance']

metric':['euclidean', 'manhattan']

Linear Models:

LogisticRegression

A major disadvantage of using LogisticRegression for our data is that it might not be possible to tell which features are important by looking at the coefficients directly.

Tuning the C parameter helps us achieve a balance between generalization and complexity.

C = [10**-5, 10**-4, 10**-3, 10**-2, 10**-1, 1, 10, 100, 1000, 10000]

SVC

C = [0.01,0.1,1.0,10,100]

kernel = ['linear', 'poly', 'rbf', 'sigmoid']

Ensemble Models

BaggingClassifier with LogisticRegression

BaggingClassifier with LinearSVC

BaggingClassifier with SVC

The same parameters for max_samples and n_estimators are used across all bagging models.

max_samples = [893, 100, 10, 200, 220]

n_estimators = np.arange(1, 15)

The parameters used to tune the base estimators, LogisticRegression, LinearSVC and SVC are the same as used above.

Pipeline with SVD and another model.

The parameters used to tune the SVD are n_components = [2, 3, 4, 5]

The parameters used to tune the ensemble are the same as above.

SVD with BaggingClassifier(LogisticRegression)

SVD with GradientBoostingClassifier

VotingClassifier

Ensemble with the best combination of 3 models obtained from above. This is been described in detail in the evaluation section.

Evaluation

We used 3 fold cross validation with 33% of the data as holdout data for testing and evaluating our models. Since our dataset has an uneven distribution of classes, using accuracy as a measure would not make sense for comparing models, thus we are using AUC to compare models. Also, since our business problem does not comprise of ranking the target instances, we need not use lift curves and cumulative response curves to select the top x%. The best models were obtained by doing a parameter search on the specified models to get the best mean cross-validation score after doing a 3-fold split on our training data.

On our first iteration of the modeling process, we ran several regression models on the original dataset that did not account for inflation. Below are the models we used to predict opening weekend revenue, with corresponding R-squared values:

Gradient Boosting Regressor - 0.57

Bagging Regressor with Elastic Net - 0.58

Bagging Regressor with LinearSVR - 0.54

K-Neighbors Regressor - 0.58

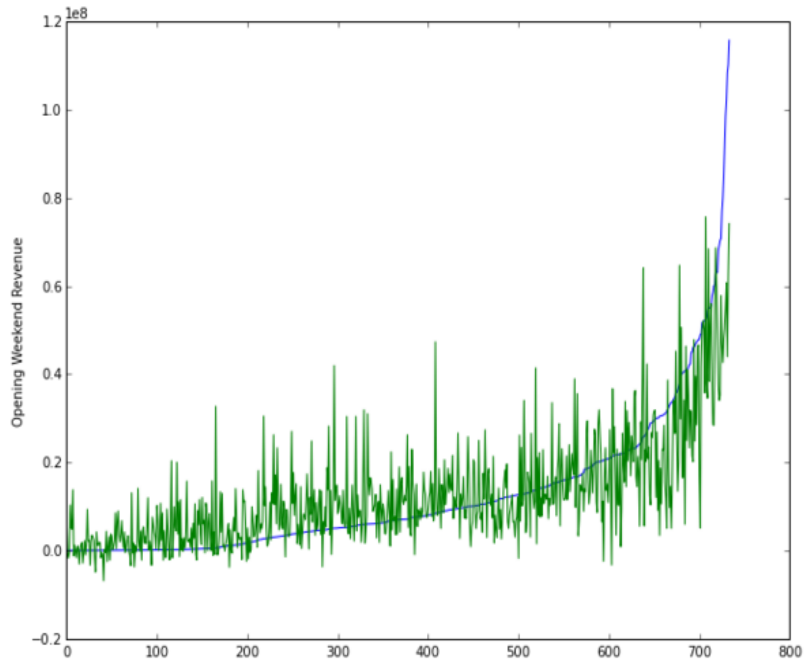
Decision Tree Regressor - 0.46

LinearSVR - 0.59

Elastic Net - 0.59

Random Forest Regressor - 0.56

Results of the model with the highest R-squared value, LinearSVR, were as follows:



The blue line shows the actual opening weekend revenue normalized by the standard deviation while the green line shows the predicted value obtained by the best model that shows significant variation from the actual values.

In training our regression models, considering the effect of inflation using the models we get very poor R2_values, around 0.118 - 0.2, for all the models mentioned in the modeling section.

We found the following attributes were most dominant information gains, once we decided to treat this as a classification problem:

Feature Importance (.45 cutoff)

Decision Tree Feature	Importance
budget	0.3913295
Drama	0.3100431
production_weights	0.1544753
Horror	0.08942511

Random Forest Feature	Importance
budget	0.1882432
Drama	0.165421
production_weights	0.1566651
Horror	0.09216485

Gradient Boosting Features	Importance
budget	0.3913726
production_weights	0.214107
Drama	0.2070846
R	0.06222008

The above tables show the most influential features in some of our classification models. For each model, budget was the most important feature to consider. The standard revenue that a production company tends to accrue for prior movie releases was also an important indicator, as were two particular movie genres: Drama and Horror.

Feature Importance (.60 cutoff)

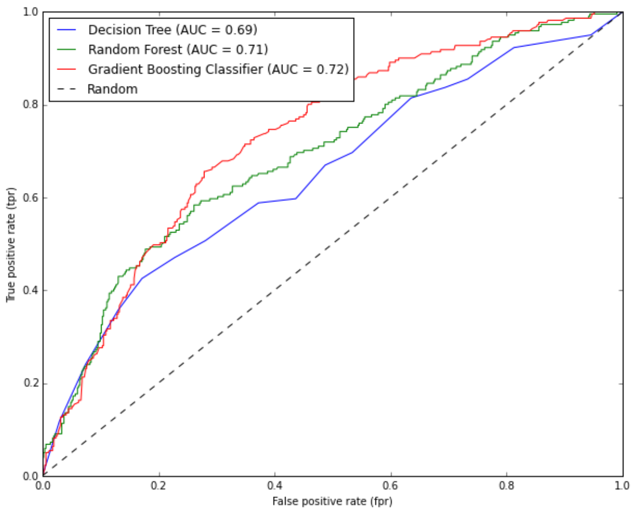
Decision Tree Feature	Importance
Drama	0.3838285
budget	0.3433555
production_weights	0.1470199
Horror	0.1136333

Random Forest Feature	Importance
budget	0.3117492
Horror	0.1838174
Drama	0.1168157
production_weights	0.1093749

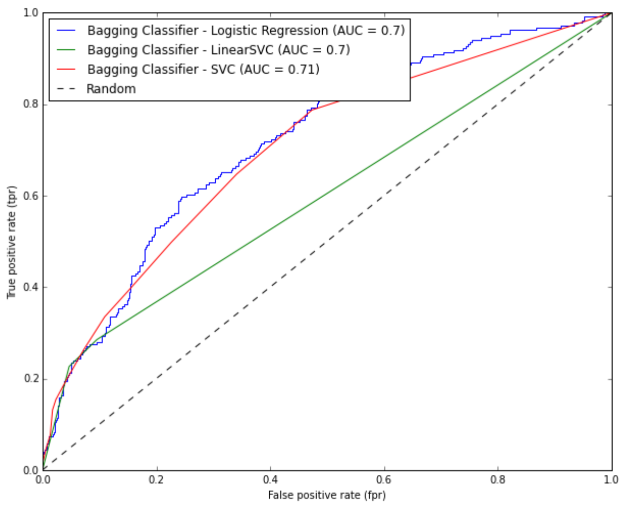
Gradient Boosting Features	Importance
budget	0.2632788
top writer 1	0.2523882
top director	0.1773303
top actor 1	0.1142691

The above tables show the most influential features at a higher threshold for opening weekend success. With this higher standard the movie genre plays a more important role in determining the opening weekend revenue, and with Gradient Boosting, more specific features like having a famous actor, director, or writer involved with the film, become even more influential.

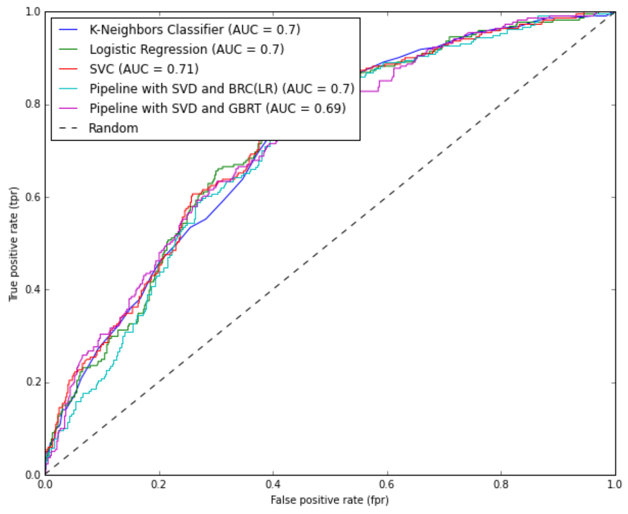
ROC Curves (.45 cutoff)



Considering only tree-based models, Gradient Boosting provides better results than a Decision Tree or a Random Forest model. The difference becomes especially pronounced as the fpr goes above 0.2.

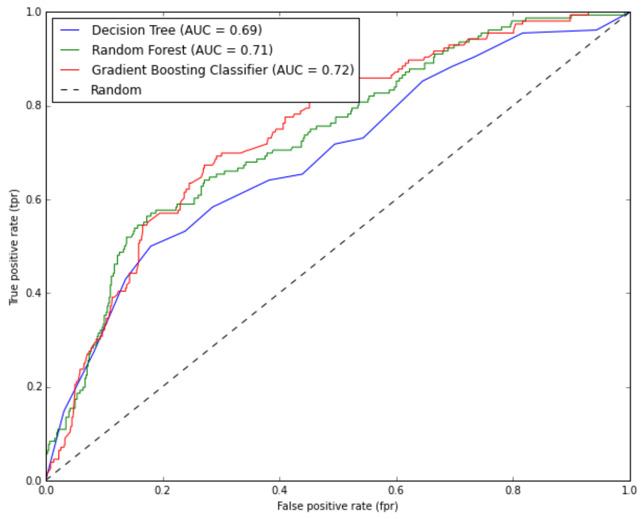


Out of the bagging classifiers, those incorporating Logistic Regression and SVC do better than those trained using LinearSVC.

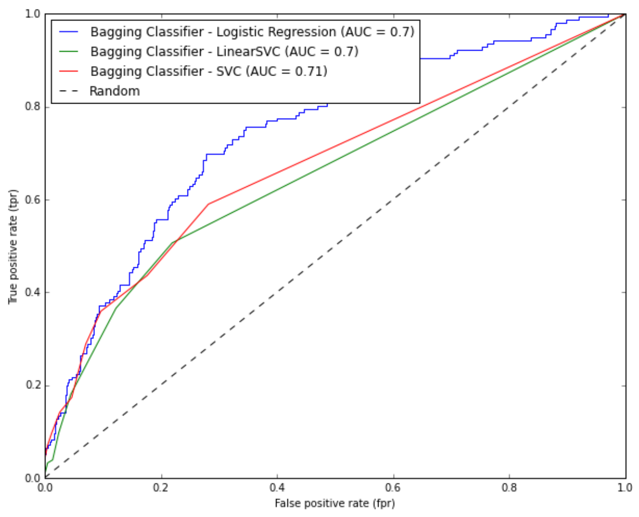


We also fit our training data to a K-Neighbors Classifier, Logistic Regression, SVC models, and two pipelines combining SVD and either BRC (LR) or GBRT. The model returning the most correct results per total training samples was SVC.

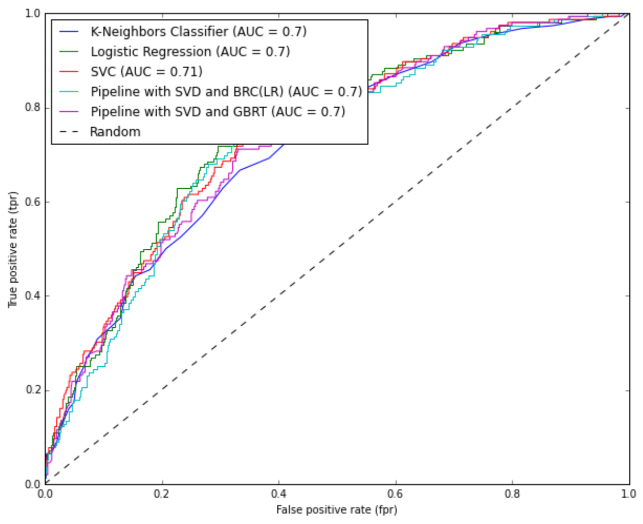
ROC Curves (.60 cutoff)



Increasing our threshold for a successful movie did not impact the tree-based models significantly, so the Gradient Boosting Classifier was still the best choice among these three options.



Increasing the threshold did change the bagging classifier models slightly. While the AUC remained essentially the same, the bagging classifier using SVC nearly matched the bagging classifier using Logistic Regression, though the latter was still the best choice among these options.



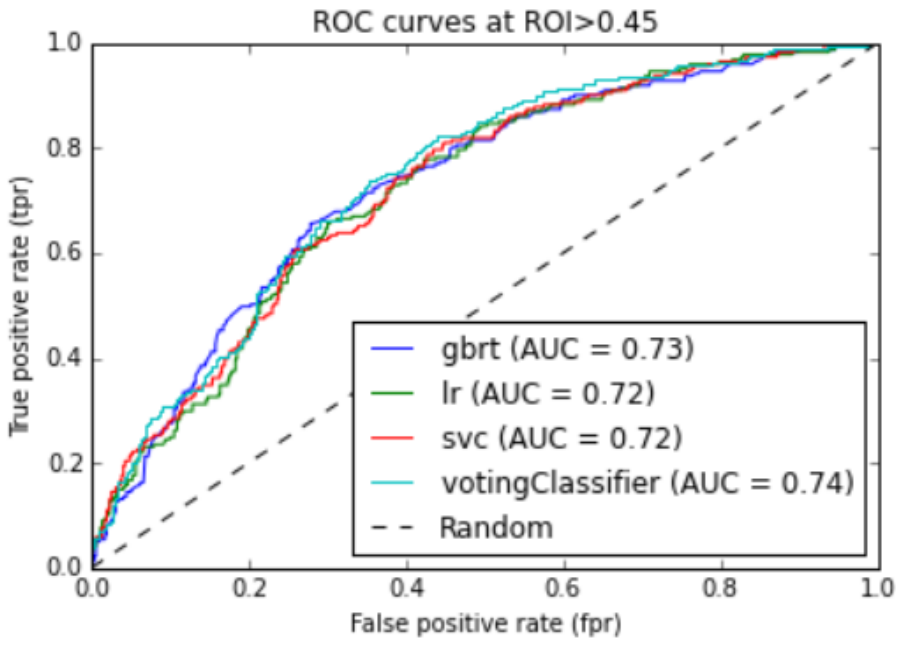
As is the case above, the threshold difference did not impact these models significantly. SVC was still the best option when true and false positive rates, with an AUC of 0.71.

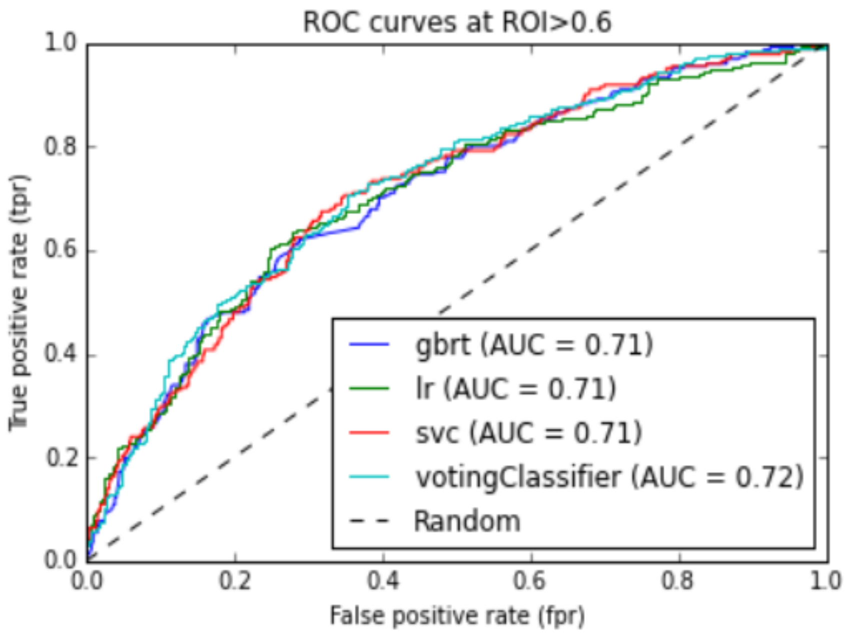
All the curves drawn above are for the models that give the best area under the curve.

Voting Classifier

We used the voting classifier in scikit-learn to obtain the final probabilistic estimates for a new movie to predict if it is a hit or not. We did a grid-search over all possible combinations of three models to get the best AUC-ROC score. The voting classifier works by averaging the probabilistic estimates of all the three models (by giving equal weights to all three models) to obtain new probabilistic estimates which we use to plot the new area under the curve. Intuitively this is useful because if one model gets a probabilistic estimate completely wrong, the other models compensate for it.

In practice the number of models used and the weights given to each model in the VotingClassifier can also be tuned, however for this project we settle for the best 3 giving equal weight to all.





The above figures show the results of the VotingClassifier, broken down into its component models, for the two potential thresholds one can set.

Setting a threshold of 0.5

Confusion matrix of our model at ROI>0.45

	p	n
Y	52	34
N	169	479

Confusion matrix of our model at ROI>0.6

	p	n
Y	59	35
N	162	478

This shows that we have a very low false positive rate even a threshold of 0.5, which means our model is able to predict most of the duds correctly and thus able to minimize the risk for producers of investing in wrong projects.

Deployment

If a production company can afford to produce only one movie with a fixed budget and a set of x different movies that it has been approached with, they can use this model to choose which one is most likely to provide a return on investment. Sometimes even after the movie is ready for production it might be possible to change some aspects, such as certain elements of the script of the MPAA rating, to give the maximum possible boost in the probability of being a hit.

Since our best model is a voting classifier ensemble model, it might not be straightforward to understand how it works, as it could potentially have too many technicalities to be easily grasped by our producer stakeholders. However we can use the feature importances from the tree based models and the zeroed out coefficients from the linear models to give an idea of which factors are most likely and unlikely to boost the chances of it being a hit.

In addition, our model is based on a limited number of movies from the IMDB dataset. It is possible that this is not a good representative of all the movies produced so far and might be biased towards movies that have a minimum amount of popularity. This can be mitigated to an extent by mining more data. Beyond additional movie samples, there are many other features that we would ideally include in future iterations. Features such as the music director, production sound team, visual and special effects, costume designers, animation teams and make up artists can also have a huge impact on the success of a movie.

With the recent boom in popularity in online services such as Netflix and Hulu, it is possible that the movie industry may be shift away from the traditional metrics of success that have guided production decisions in the past. There may well be a movie genre that does exceptionally well online for an extended period of time, even if it was not an instant hit in theaters, or bypassed this step in the release process altogether. If this were true, we would need to alter this model to find patterns in such movies and determine what other metrics, besides opening weekend box office revenue, may indicate an more extended and steady return on investment in the long run.

Finally, as mentioned before, features such as famous actors, writers and directors are based on a list derived from online blogs and other resources that are totally subjective. It also assumes that every famous actor, writer or director listed in these websites contributes equally to the success of a movie, which is not true. A way to solve this would be to rank all given movies in terms of opening weekend ROI and give a weight based on the number of times a certain actor, director, or writer has appeared in these top-grossing movies.

There will also be outliers in such industries, and domain knowledge will be critical in understanding the outcomes of our model. For example, the recent success of the latest Star Wars movie release has been an unprecedented success in the box officer, likely exceeded any model predictions of initial revenue coming in from this particular source. Understanding the types of genres, series, or even actors that have such strong support is critical in developing a comprehensive outlook for any particular movie.

Bibliography

"Box Office Mojo." *Box Office Mojo*. N.p., n.d. Web. 29 Dec. 2015.

"CPI Inflation Calculator." *U.S. Bureau of Labor Statistics*. United States Department of Labor, n.d. Web. 28 Dec. 2015.

Deniz, Borga, and Robert Hasbrouck. "WHEN TO GREENLIGHT: Examining the Pre-release Factors That Determine Future Box Office Success of a Movie in the United States." *International Journal of Economics and Management Sciences* 2.3 (2012): 35-42. 2012. Web. 28 Dec. 2015.

Hamell, Paul. "Greatest Screenwriters." *IMDb*. IMDb.com, n.d. Web. 29 Dec. 2015.

Im, Darin, and Minh Thao Nguyen. *Predicting Box-Office Success of Movies in the U.S. Markets*. *Www.standford.edu*. N.p., Fall 2011. Web. 28 Dec. 2015.

IMDb. IMDb.com, n.d. Web. 29 Dec. 2015.

Kennedy, Alec. *Predicting Box Office Success: Do Critical Reviews Really Matter? Www.stat.berkeley.edu*. N.p., n.d. Web. 28 Dec. 2015.

Miyamoto, Ken. "What Are the Best Months to Release a Movie for Theatrical Viewing?" N.p., n.d. Web. 29 Dec. 2015.

"Movie Budgets." *The Numbers*. Nash Information Services, LLC, n.d. Web. 28 Dec. 2015.

"Movie Production Companies." *The Numbers*. Nash Information Services, LLC, n.d. Web. 28 Dec. 2015.

"MovieLens." *GroupLens*. N.p., 06 Sept. 2013. Web. 29 Dec. 2015.

"The 1,000 Greatest Films (Top 250 Directors)." *The 1,000 Greatest Films*. TSPDT, n.d. Web. 29 Dec. 2015.

"Top 50 Popular Hollywood Actors and Actresses." *IMDb*. IMDb.com, n.d. Web. 29 Dec. 2015.

Topf, Patrick. "Examining Success at the Domestic Box-Office in the Motion Picture Industry." *Www.iwu.edu*. N.p., 2010. Web. 28 Dec. 2015.

Member Contributions

Business Understanding Lead: Shrenik

Data Understanding Lead: Anna

Data Preparation Lead: Anna

Modeling Lead: Manoj

Evaluation Lead: Manoj

Deployment Lead: Shrenik