OCCIDENTAL COLLEGE
COMPUTER SCIENCE DEPARTMENT

Computer Science 349: Machine Learning
Project No. 3: Predicting Hollywood Success

Dr. Kathryn Leonard

Shasta Clokey

Los, Angeles, CA, November 13, 2017

Description of Data and Research Question

The goal of this analytic venture is to identify features in movies which manifest a high ranking, create a successfully rated movie, and enable a movie to attain a substantial revenue. To do this, we will examine a dataset containing both numerical and textual data entries for 1000 unique movies spanning from 2002 until 2016. The data categories are as follows :C1 = Title, C2 = Genre, C3 = Description, C4 = Director, C5 = Actors, C6 =Year, C7 = Runtime (min), C8 = Metascore, C9 = Votes, C10 = Rating, C11 = Revenue (M), and C12 = Rank.

Our set of response variables consisted of 1000 observations of movie rank, rating, and revenue. In order to create a set of classifications for the rank of a movie, we transformed the set of movie ranks into a ten part classification set, 1 denoting 1-100 … 10 denoting 901-1000. In order to classify whether a movie had a successful rating, we transformed the set of movie ratings into a binary classification set, true denoting above 7.5 and false denoting below. Because we planned to use a regression to predict the revenue, we left the set of revenue in terms of millions of dollars .

Our initial set of quantitative predictors included year, runtime, metascore, and votes. However, we determined that the most efficient way to utilize the movie description and title categories was to replace their entries with the corresponding length of each piece of data adding two categories to our set of quantitative data. Our goal with the quantitative dataset was to determine whether the length of a title/description, the year, the runtime of a movie, the metascore, or the number of positive votes cast by critics have a significant effect on the ranking of a movie, on the success rate of a movie's rating, or the revenue a movie produces.

Using Logistic Regression to Predict Rank

To predict the range in which a particular movie would rank from 1 to 1000, we used a logistic regression classification scheme. This method allowed us to use our qualitative and quantitative datasets to determine whether a movie would be ranked in the top 100, 101-200, etc. The logistic regression algorithm hinges on the equation for a logistic probability curve(1). By manipulating (1), we can formulate the equation for the log odds, or logit, which is a less computationally expensive way to measure the probability(2). By choosing the betas associated with each predictor in a way that maximizes the likelihood that the predicted output is equal to the true output, we can create a model to be used for classification of future data(3).

logistic regression → probability P(y=1 | x)

$$P(x) = \frac{e^{\beta_0+\beta_1 x_1+\ldots\beta_p x_p}}{1+e^{\beta_0+\beta_1 x_1+\ldots\beta_p x_p}} \qquad (1).$$

$$\log\left(\frac{P(x)}{1-P(x)}\right) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p \qquad (2)$$

$$\text{argmax}\ \Pi_{i=1}^{n} p(X_i)^{c_i}(1-p(X_i))^{(1-c_i)} \qquad (3)$$

prob that X gets label 1, raised to (1- the true label)

prob that X gets label 0, raised to the true label (0 or 1)

Using Support Vector Machines to Predict Rating

Because we needed to classify whether a given movie's rating was above or below 7.5, we used the support-vector-machine binary decision making model, which utilizes a functional decision boundary and a finite subset of the observations as "support vectors", to classify new data. The goal of the algorithm is to separate the data points so that all points of one class label are on one side of the decision boundary, all points of the other class label are on the opposite side, and between the two groups of data there is a buffer zone called the margin which is bounded by data points which are given the name "support vectors". The most general form of the algorithm works by specifying a function to use for the decision boundary, deciding how much error you are willing to tolerate within your training data to account for outliers, using these two characteristics to iteratively find the maximal margin between the two groups of data, and finally using this decision boundary to predict results based on future observations(4). There are two ways to handle datasets with complex decision boundaries using support vector machines. The first is to map every data point to a higher dimension where a simpler decision boundary can be leveraged, then map it all back to the initial dimension to classify the data. This method was found to be incredibly computationally expensive, so the math gods shined down and introduced a powerful technique from the study of integral transforms called the kernel(5). Using these kernels, which are fancy inner products, we can specify more interesting decision boundaries like nth degree polynomials or infinitely-dimensional radial decision boundaries. For our model we examined linear, polynomial, and radial decision boundaries, and found radial to produce the lowest error rate(6).

$$\text{maximize} \quad \frac{|b_1 - b_2|}{||\vec{\beta}||}$$
$$\text{subject to} \quad y_i(\vec{\beta} \cdot \mathbf{x}_i) \geq 1$$

(4)

$$\vec{\beta} = \sum_{I=1}^{n} \alpha_i \mathbf{x}_i \quad \rightarrow f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i K(\mathbf{x}, \mathbf{x}_i)$$

(5)

nontrivial example kernels:

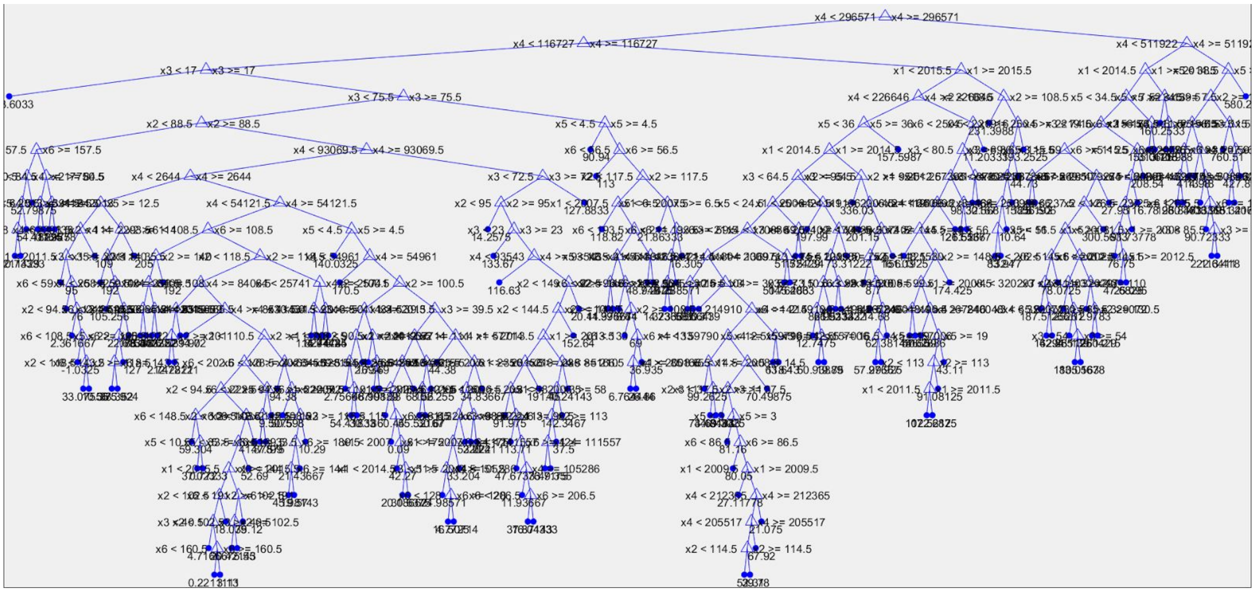
1. polynomial decision boundary: $K(\mathbf{x}, \mathbf{x}_i) = 1 + (\mathbf{x} \cdot \mathbf{x}_i)^d$
2. radial basis decision boundary: $K(\mathbf{x}, \mathbf{x}_i) = e^{-\gamma \sum_j (x_j - x_{ij})}$

(6)

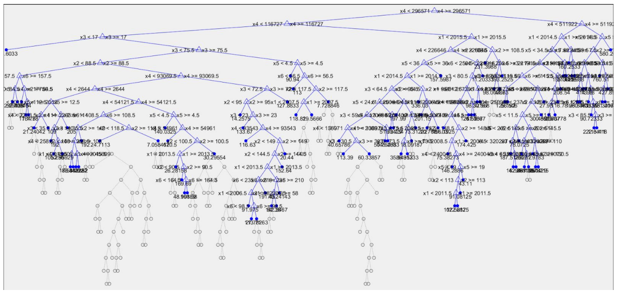
Using Regression Trees to Predict Revenue

In order to predict the amount of money a given movie would make given a set of predictors, we used a pruned regression tree to filter through our predictors and produce reasonable estimated response values. The regression tree algorithm functions by making boolean decisions based on the predictors, forming a descending tree structure starting at the principal node(root) and ending at the terminal nodes(leaves). The value assigned to each terminal node in a regression tree is the average value of all points in the training set which fall into that category in the model. To make predictions on new data using a decision tree model, the new observations are fed into the tree structure and filtered through the branches until they land in one of the tree's terminal node where they are assigned the value of that leaf. Pruning the tree involves cutting out decision layers until the regression error rate is minimized. Pruning can help ensure that the training data is not overfit by the model. Below are displays of the tree model we used to predict revenue with no pruning(7), with moderate pruning(8), and with the model heavily pruned(9).

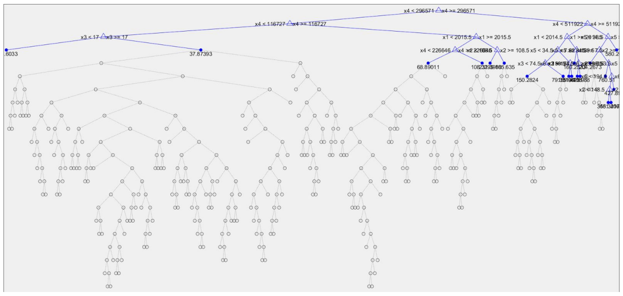
(7)



(8)



(9)



Analyzing Error of Classification/Regression Models

In this section, I will analyze the errors associated with using multi logistic regression, radial support vector machines, and regression trees to predict the rank of movies(category from 1 to 10), their rating(above or below 7.5), and the amount of money they will generate (represented in millions of dollars). The predictor-set for the model whose error I will be examining includes year, runtime, metascore, votes ,title length, and description length. I will discuss why I excluded the textual predictors(genre, actor, and director) in the following section.

We calculated the boolean error of our two classification techniques by adding up the number of times we got the wrong result and dividing by the total number of observations. We calculated the total classification error for our logistic regression model by adding up the absolute value of the difference of the predicted class labels compared to the true labels of the original dataset. To calculate the total error in our regression tree model, we used the same process as the total error for classification, except used the predicted revenue and the true revenue for the two inputs to our calculation.

After running cross-validation on on our logistic regression model, we were able to calculate both the boolean error probability of our rank predictions and also how much we were off by on average for each of our k folds during validation. Our average boolean error for logistic regression was roughly 87%, meaning that our predictions for ranking would be right 13% of the time. This probability appears to be small, but given that there are ten possible categories, the chance of being right by random chance is only 10% meaning that our model gives those who use it a 3% advantage over their competitors. Additionally, examining how far off each of our predictions are on average can tell us if we are predicting classes close to the true rankings. Our average total error for our cross validated logistic regression model was roughly 330, and given that each of our 10 folds for validation were 100 elements in size, the average error for each prediction was about 3.3. While our algorithm can only predict the right ranking category(1-100 etc) 13% of the time, the results from our model are only off by on average three categories. When trying to determine how successful a movie will be, this predicted range could be exceedingly valuable.

After computing the error for linear, polynomial, and radial support vector machine classification, we found that radial classification produced the lowest boolean error probability at 24%. This is a sign that the decision boundary for our training data was too complicated for a polynomial to fit. This result is substantial in that we can predict whether a movie will be rated above or below 7.5 with only a 24% chance of error. This will allow movie producers to determine with substantial confidence whether a movie will be successfully rated or not.

After constructing and subsequently pruning my regression tree, I found that predicting the revenue using this model produced an average error of roughly 25 which corresponds to twenty five million dollars in movie revenue. Twenty five million dollars seems like a large error, but when movies are generating up to 300 million dollars + in revenue, being able to predict the revenue within 25 million dollars is a substantial source of foresight for the producer of a film.

Analyzing Textual Data Categories/Suggestion for Future Work

After conducting an initial counting analysis on our qualitative dataset, we found that there were 21 unique genres, 644 unique directors, and 1986 unique actors in our qualitative, textual dataset. In an attempt to reduce the size of our set of unique actors, we first extracted only the first actor listed in each entry and determined by inspection and societal context that this identified the list of leading actors for our set of 1000 movies. This reduction brought our set of actors down to 526 unique names. We decided that we needed to further reduce the number of actors and directors in our qualitative predictor sets, so we counted the frequency of appearance for each of the 526 leading actors and 644 directors, sorted the two lists so the actors/directors with the most appearances in movies were shown first, and finally extracted the 25 most popular leading actors as well as the 25 most popular directors. We chose to reduce our predictor set in this way so we could test if having a popular actor or director will make the movie rank higher, have a higher rating, or make more money. This analysis could help producers and showrunners make decisions of whether to hire big-name actors and directors, or to cast for roles and find new talent. Using genres as one of our predictors would allow us to determine which kinds of movies are the most successful. Unfortunately, the top 25 actors and directors only appeared in roughly a tenth of the total movies in our dataset. When computing the necessary transformations on this sparse data, the resulting matrix was always non invertible. This caused an issue with our classification and regression algorithms. Because the producer client our team was working with wanted our analysis right away, we made the determination to exclude the three textual predictors(genre/actor/director) and produce a usable analysis with the remaining quantitative data. Given more time, a novel solution to this non-invertibility problem would be to map the actor and director predictors to predictors stating whether or not the observation contains a top 25 actor/director. This way, instead of dealing with a non-invertible bag of words matrix, we would have a singular column of zeros and ones. This would produce the same information of whether or not having a top 25 actor/director increases the revenue, raises the rank, or betters the rating.