Estimating Overall Movie Gross using IMDB Data

Akishev Arsen, Kabir Md, Lin Kevin

December 14, 2016

Abstract

The focus of the project is to determine what kind of algorithm is better suited to predict a non binary value such as the grossing of a movie. Both supervised and unsupervised learning algorithms were implemented and their average error rate was utilized to evaluate their success rate. In this paper, cross validation of predictive algorithms were also analyzed by comparing the best model discovered with validation against the simple model.

1 Introduction

With the recent growth of machine learning there are multiple angles from which the machine learning algorithms can be applied and implemented. Specifically industries such as cinema with thousands of data points are the greatest benefactors of this new technological development. The traditional methods of understanding the potential of a movie involves understanding the director, genre and actors that are involved with the movie. There were also traditional requirements of having actual people make these choices with the given data. Of course these solutions are slow and inefficient, thus are in dire need of replacement. In this paper we focus on predicting the potential movie grossing based on factors collected from an IMDB movie data-set. The data consists of both classical data points and also new social data points.

2 Related Work

A recent publication has evaluated the data gathered from IMDB and has come to a conclusion that movie categories such as action and adventure lead to higher grossing movies compared to its competing categories of biography and documentary[WZ09]. While there are other forms of criteria that allow studios to judge the overall success of a movie the main factor that is important to consider is the difference in budget and final grossing. The success of a movie cannot be set on the goal of hitting a high enough grossing, this is a biased representation of success. Also the success of a movie cannot always be based of news media mentions that it has had. While there are more accurate models build with the combination of news and IMDB data[WZ09] it cannot be the case for all movies since some were not produced around the time of mass media growth. In the data-set provided by the IMDB scrapping process there is a distinction in the data that is available to process. The data is split into classical factors such as MPAA rating, budget, actors and actresses, it is also split into social factors such as Facebook likes for the actor and likes for the movie. As mentioned in [Kos15] there is a likely chance that considering the data-sets from both social and classical perspectives will provide much more accurate results in predicting the success of a movie.

3 Data-split, Algorithms, & Cross Validation

We ran three different algorithms: two of which are supervised and one unsupervised learning algorithms. We will be displaying the results for the aforementioned algorithms below. The original data was modified in order to remove certain movies that did not have the necessary data such as original budget or final grossing. We split the entire data-set into two portions: 75 % of the entire data-set was used as the training data-set, with the remaining 25 % being used for testing. All of the data was normalized before testing. To convert attributes that cannot be simply converted to a number such as a word, we used the number of occurrences of the word and

replaced it with that number. For success rate of the algorithms we determined our average error rate for each algorithm to be within 30 %. As concluded in [SY16] the results of using k-fold cross validation over hold-out validation on colossal data sets is worth the extra 0.1 - 3 % accuracy even if there is a time trade-off. For the Perceptron algorithm we utilized 4-fold cross validation and for K-nearest neighbors algorithm we utilized 2-fold cross validation. For both of those algorithms we also evaluated the result without validation to analyze if cross validation benefited each models. All the tests were done on a Macbook Pro 8GB RAM 2.7 Ghz Intel Core i5. Prior to running all of the algorithms we normalized all of the data using the formula: $normalized = \frac{dataset[attr]-MIN[attr]}{MAX[attr]-MIN[attr]}$.

3.1 K-Means

- We implemented K-Means as the only unsupervised learning algorithm. We designed a simple K-Means implementation with only three clusters.
- K-Means had an average error rate of **65%** and **68%** and its approximate run-time to build and predict movie gross was between **2000 milliseconds** and **3000 milliseconds**. In respect to both run-time and error rate K-Means was the worst performing algorithm and did not fall within our set 30% average error rate.
- We wanted to determine the effectiveness of clustering to determine a non-binary value. We predicted a movie gross based on the average gross of the cluster it was closest to. Below Table 1 Shows the Silhouette Coefficient values for a sample clustering run.
- Table 2 shows the average error percentage and Approximate Run-time for a test run.

Table 1: Silhouette Coefficient for K-Means Cluster

K-Means Cluster	**Silhouette Coefficient**
cluster 1	0.998730
cluster 2	0.998561
cluster 3	0.998794

Table 2: Error and Run time for K-Means

Category	**Value**
K-Means Average Error	67.8037%
Approximate Run-time	2792 ms

3.2 K-nearest neighbor

- The first supervised learning algorithm that we implemented was K-nearest neighbor or K-NN algorithm. We used simple euclidean formula to find the closest neighbor.
- For this algorithm we tested two different models of K-NN. First model was the simple K-NN design that utilized all of the training set and the second Model was the 2-fold cross validated model.
- Overall regardless of numbers of neighbors used cross validated model of K-NN generated better results than the simple model. For various parameters used on average the cross validated model's average error was approximately 20% lower than simple model's average error rate.
- Following Table 3 shows the analysis of the different models and their error rate using **43 neighbors**, which gave the best set of results. The run-time was evaluated by taking overall run-time and dividing it by 2 since the two models were predicting simultaneously. The best average error rate generated by K-NN was approximately **65%**, which is somewhat lower than K-Means but it still does not meet our set standard error of 30%.

Table 3: Error and Run time for KNN

Category	**Value**
KNN Two Fold Validated Model Error	65.308%
KNN Simple Model Error	84.2063%
Approximate Run-time	1347 ms

3.3 Perceptron

- Perceptron was the second supervised learning algorithm that we implemented for our dataset. For this algorithm we used the standard design for a Perceptron algorithm.
- For this algorithm we used two models again one that is the simple model and the other is 4 fold cross validated model.
- Just like running K-NN algorithm, we also ran the models of the Perceptron together and retrieved each perspective error rate and approximate run-time.
- Table 4 below shows the data acquired from Perceptron for a test run. Overall this algorithm was the best with respect to both average error rate and run time. The cross validated model met our standard requirement as its error rate was approximately **17%** and the the simple model was also very close to our requirement as it was approximately **35%**. For both models the optimal learning rate was **.5** and the weights were initialized to **1**.

Table 4: Error and Run time for Perceptron

Category	**Value**
Perceptron Four Fold Validated Model Error	17.0455%
Perceptron Simple Model Error	34.8965%
Approximate Run time	13 ms

4 Closing thoughts on our algorithms and Results

Based on the analysis of the error rate of each algorithm we determined that machine learning algorithms that utilizes distance between data points are inefficient for predicting non-binary values such as movie gross. We believe that the reason for that is that when the data points are very scattered apart, using distance formulas such as euclidean formula becomes ineffective. Given that we only used a small sample set, we believe that we had many scattered points which led to the inefficiency in K-NN and K-Means algorithms. Perceptron performed the best and met our standards because it relies on the weights of the attributes rather than the distance between points. As mentioned in [Gei04], in the case that plot data is scattered, it may be better to use a Perceptron. Finally, to improve our KNN accuracy, we can implement KNN Structured Prediction as explained in [PD11]. KNN Structured Prediction is able to handle varied datasets more efficiently then the regular KNN. We also showed through our data analysis that cross validation significantly improves the performance of the models, as it lowered K-NN error by 20% and halved the error of the Perceptron implementation.

References

[Gei04] F. Geibel, P. & Wysotzki. Applied Intelligence. 21(1):45–56, 2004.

[Kos15] A. Bhave H. Kulkarni V. Biramane P. Kosamkar. Role of Different Factors in Predicting Movie Success. *ICPC*, (3):1–4, 2015.

[PD11] Mitja Pugelj and Sašo Džeroski. *Predicting Structured Outputs k-Nearest Neighbours Method*, pages 262–276. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[SY16] Sanyam Shukla Sanjay Yadav. Analysis of k-fold Cross-Validation over Hold-out Validation on Colossal Datasets for Quality Classification. *IACC*, 6:78–83, 2016.

[WZ09] Steven Skiena Wenbin Zhang. Improving Movie Gross Prediction Through News Analysis. *WI-IAT*, 3, 2009.