Predicting Movie Success Using Machine Learning Algorithms

[1]Cary D. Butler
Jackson State University
1400 John R. Lynch St
Jackson, MS 39212
carydbutler@gmail.com

Eric Jackson
Jackson State University
1400 John R. Lynch St
Jackson, MS 39212
eric.jackson36@gmail.com

Li-jing Chang
Jackson State University
1400 John R. Lynch St
Jackson, MS 39212
li-jing.a.chang@jsums.edu

Mahzabin Akhter
Jackson State University
1400 John R. Lynch St
Jackson, MS 39212
akhterm86@gmail.com

Md Atiqur Rahman
Jackson State University
1400 John R. Lynch St
Jackson, MS 39212
M.D.Rahman@students.jsums.edu

ABSTRACT
Determining to see or not to see a movie can be a challenging decision. In the past, many people have decided to view or not to view a movie based on the reviews from critics. Other people have decided to view or not to view a movie based on what friends and others have said after they've watched it. Most people are interested in only watching a movie at the box office if it is deemed successful. This paper proposes a way to predict how successful a movie will be prior to its arrival at the box office. Instead of listening to critics and others on whether a movie will be successful, we have applied machine learning algorithms to make this decision. A total of five machine learning algorithms (K-nearest Neighbor (KNN), Gaussian Naïve Bayes (GNB), Decision Trees (DT), K-means Clustering, and Graphing Theory) were applied to a dataset comprised of movie data from 2 different sources (IMDB and YouTube). This dataset contained 500 randomly selected movies and 28 features.

Keywords
K-nearest Neighbor, Gaussian Naïve Bayes, Decision Trees, K-means Clustering, Graphing Theory, IMDB, YouTube API (Application Programming Interface), and equal-depth binning.

1. INTRODUCTION
Cinema has a profound impact on our society. Cinema is one of the most powerful media for mass communication in the world. Cinema has the capacity to influence society both locally and globally [1]. Many different kinds of movies are made every year. Some movies portray historical events, some create a culture, while some provide fantasy, and some do many more [2].

Though movies are capable of providing many different themes, what makes a movie successful? Before a movie is released to the public, is there a way to determine how successful it will be at the box office? Can we determine if we want to see a movie outside of reading a critics' review? This paper, aims to answer these questions. This paper focuses on using the IMDB 5000 Movie Dataset from Kaggle Data [3], an online source in which companies and researchers post data for statisticians, data miners, and the general public to work on to produce predictive modeling to answer various questions, which one may have on the data.

Also, movie trailer data was be collected in the form of views, likes, dislikes, and comments from YouTube, a very popular video-sharing website [4]. The movie trailer data was converted to a dataset and combined with the IMDB 5000 Movie Dataset into one data frame. Afterwards, we used the data to attempt to answer such questions as: "Can we predict if a movie will be successful, prior to it coming to the box office? What factors make a movie successful among the public?" and "What movies are similar when success rates and YouTube view counts are considered?"

To assist us in answering these questions, we applied five machine learning algorithms to the dataset. The machine learning algorithms that were applied was K-Nearest Neighbors (KNN), Gaussian Naïve Bayes (GNB), Decision Trees (DT), K-means Clustering, and Graphing Theory [5] [6].

2. Machine Learning Tasks
Three of the five machine learning tasks, KNN, GNB, and DT, are supervised learning techniques (i.e., the learning is guided by the target variable: gross sales) [7]. Under supervised learning, the values predicted through these models can be compared to the true output values to estimate the magnitude of errors.

The other two of the five machine learning tasks, K-means Clustering and Graph Theory, are unsupervised learning tasks. The procedures are called unsupervised because the predicted values will have no real output values to compare to.

To determine which supervised and unsupervised methods are best approaches to answer the questions in the introduction section, we used the procedure of 10-fold cross validation. The 10-fold cross validation process was used because compared to simple training-test data split, it will be able to reduce over-estimation of error variance through averages of the results [8][9].

[1] Cary D. Butler, PhD
Technical Director, Information Technology Laboratory
U.S. Army Engineer Research and Development Center

3. Dataset

The initial dataset used was collected from IMDB 5000 Movie dataset from Kaggle. Among these movies, we selected the ones that were released in the United States. Afterwards the original dataset contained 897 rows. Next we randomly chose 500 movies from this dataset. After collecting data from IMDB we also collected Movie Trailer Data from YouTube and combined them to predict movie gross earnings. YouTube has only been in existence since 2005, therefore, the dataset consist of movies that were released between 2010 through 2016.

3.1 IMDB Dataset

The IMDB 5000 Movie Dataset consists of 28 features for 5043 movies that span across 100 years in 66 countries, as well as gross earnings. There are 2399 unique director names, and thousands of actors/actresses. Below are the 28 features:

"movie_title"
"color"
"num_critic_for_reviews"
"movie_facebook_likes"
"duration"
"director_name"
"director_facebook_likes"
"actor_3_name"
"actor_3_facebook_likes"
"actor_2_name"
"actor_2_facebook_likes"
"actor_1_facebook_likes"
"gross"
"genres"
"cast_total_facebook_likes"
"facenumber in poster"
"plot_keywords"
"movie_imdb_link"
"num_user_for_reviews"
"language" "country"
"content_rating"
"budget"
"title_year"
"imdb_score"
"actor_1_name"
"num_voted_users"
"aspect_ratio"

Out of all of these features from the IMDB dataset, only "movie title", "title year", "country", and movie "gross" earnings were used.

3.2 YouTube Dataset

YouTube is one of the largest video-sharing websites and allows users to upload, view, rate, share, add to favorites, report and comment on videos. Our goal is to predict movie success (gross earnings) prior to its release based on the YouTube official movie trailer data (i.e. trailer's views, likes-dislikes, and comment counts). We utilized Python modules, and generated Python code to collect movie official trailer statistics using YouTube API. Following are the most relevant data features:

"Like count", "Dislike count", "View count", and "Comment count"

3.3 Data Preprocessing

3.3.1 Phase 1:

We extracted 500 randomly selected movies from the IMDB 5000 Movie dataset based on their gross. Afterwards we used equal-depth binning method to bin the dataset into 5 subsets using the 5 categories (df1 (Very High Success), df2 (High Success), df3 (Avg. Success), df4 (Low Success), df5 (Very Low Success)) under the feature "success rate". Next, we collected official movie trailer data of 500 movies using YouTube API. These features include trailer likes, dislikes, view counts, and comment counts. Next, we merged both IMDB and YouTube into one dataset. Finally, we removed duplicate instances of movies from the dataset. The final dataset was comprised of **491** movies.

The Heatmap (Figure 1) and the correlation matrix (Figure 2) display the correlations of 5 features, where 4 of the features (vid_view, vid_like, vid_dis_like, and vid_comment) are from the YouTube dataset and 1 feature (gross) is from the IMDB dataset. The Heatmap displays strong correlations amongst YouTube features (see Figure 1). The Heatmap also displays low, but positive correlations between YouTube features and gross (see Figure 1). For a clearer understanding of the correlations between "gross" vs. "YouTube features", we can examine the correlations matrix (see Figure 2).

The correlation between video view (vid_view) and gross is 49% (see Figure 2), whereas the video comment (vid_comment) has a correlation of 43% with gross (see Figure 2). There is a weak, but positive correlation between video likes (vid_likes) and gross of 34%. There is also a positive correlation between video dislikes (vid_dis_like) and gross of 27% (see Figure 2). In conclusion, there are no negative correlations between gross and the 4 features.

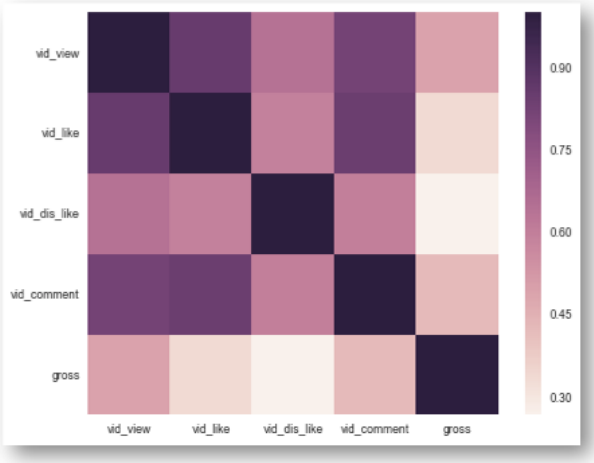

Figure 1. Heatmap (Instances -491, Features - 4)

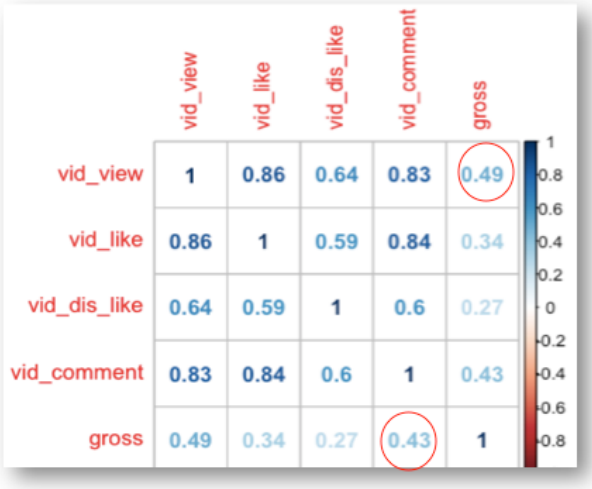

Figure 2. Correlation Matrix (Instances -491, Features - 4)

3.3.2 Phase 2:

After performing Data Distribution, Correlation and Confusion matrix using the Python libraries (seaborn and sklearn) we found the following:

- Too much variation in df2 (High Success), see Figure 3 and df4 (Low Success), see Figure 4
- df2 - poor correlation with gross (14%), this is seen by the horizontal line in Figure 3
- df4 – poor correlation with gross (16%), this is seen by the horizontal line in Figure 4

After these findings, we removed the data and now we have 293 instances left which includes df1 (Very High Success), df3 (Average Success), and df5 (Very Low Success). We added more features into the dataset in the anticipation that it might decrease our error rate we got previously.

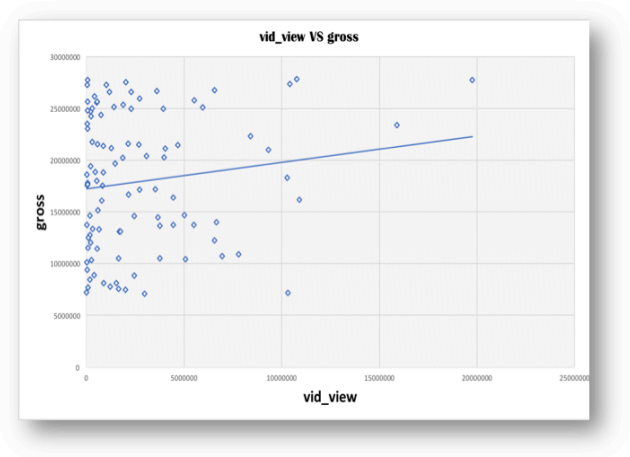

Figure 3. df2 (High Success) Distributions (Instances -491, Features - 4)

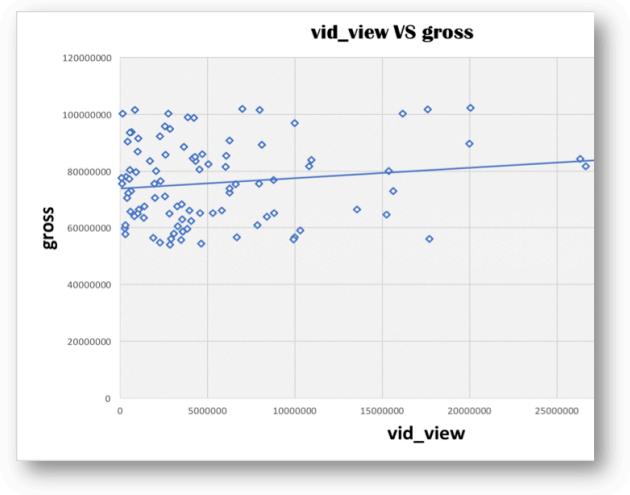

Figure 4. df4 (Low Success) Distributions (Instances -491, Features - 4)

4. Methods and Models

A total of five machine learning models were used to answer the research questions. The machine learning techniques are tools for data mining in order to learn from the data and derive the prediction models [10].

The supervised learning tasks address the following questions: "Can we predict if a movie will be successful, prior to it coming to the box office? What factors make a movie successful among the public?"

The nature of the questions calls for supervised learning techniques. The questions relate to the use of input features to predict the target variable. Because the prediction errors can be calculated by comparing predicted target values with true target values, the errors will be used to compare the performance of different models.

KNN, GNB and DT models use the training data to predict a model, and the learning process is guided by the true data. Regardless of this similarity, the models also differ. The KNN uses the Euclidean distance between the input variables to predict the target variable, the GNB employs combined probabilities of the features and outcome probability, and the DT takes advantage of maximum entropy reduction as a guide to expand the decision trees [5].

For the unsupervised learning tasks, we used K-means Clustering and Graph Theory models. The unsupervised learning tasks address the following question: "How are the movies similar considering features such as success rating and YouTube view count?" The techniques are chosen based on the nature of the question. The question relates to the use of only input variables to predict the similarities of movies. For the reason that there exists no actual data to guide the model, this question is best answered by unsupervised machine learning techniques.

Both K-means Clustering and Graph Theory use input features to predict and learn similarities or group memberships among the movies. K-means Clustering runs through an algorithm by first randomly choosing K centroids (central points) to determine the first group membership and then vectors of all members in each group are averaged to obtain a new centroid [11]. The new centroids for all groups will then be used to derive new group memberships [11]. The process goes on until convergence is reached. Graph Theory, on the other hand, use the adjacency list or adjacency matrix of the nodes (i.e., movies in the present study) as a basis to aim for modularity maximization to learn and predict group membership [12]. Modularity is a measure for the strength of community structure [13][14].

5. Results KNN, DT, GNB

5.1 Results – 491 Instances and 4 Features

The initial dataset we used in the models had 491 instances and 4 independent features such as YouTube movie trailer's vid_view, vid_like, vid_dis_like, and vid_comment. The result (Figure 5) we found after running the three models (i.e. KNN, DT, & GNB) was unexpected. GNB had an average error of 73% in predictions. KNN and DT had an average error of 75% and 76% accordingly.

Figure 5. Classification Results (Instances -491, Features - 4)

5.2 Results – 293 Instances and 4 Features

One of the primary objectives of this research was to find what factors play a part in predicting movie success. Thus, the error of the prediction model can also be decreased by taking other perspectives such as introducing unexplored features that might be related to the prediction of the movie success. Therefore, we went back to the data preprocessing step (phase 2) and examined multiple approaches to improve the performance of the prediction model. First, we plotted the data distribution of each dataset (i.e. df1, df2, df3, df4, & df5). Next, the data with high variations and poor correlations with the target concept (gross) was found and removed. The data removed was df2 (High Success) and df4 (Low Success). After removal of df2 and df4, we have 293 instances remaining in the final dataset. We ran the three models (KNN, DT, & GNB) again and found that the error has decreased a bit (see Figure 6).

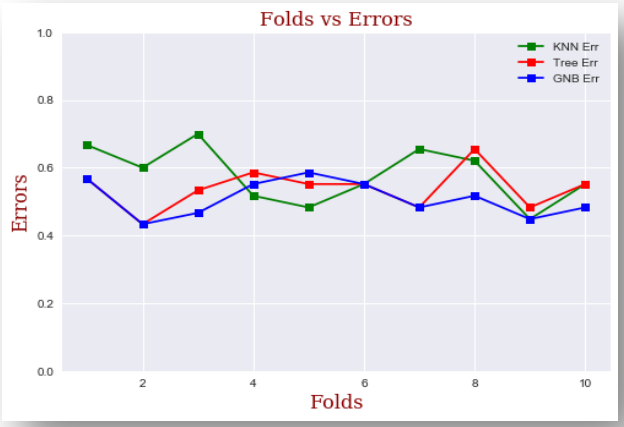

Figure 6. Classification Results (Instances -293, Features - 4)

5.3 Results - 293 Instances and 10 Features

After we removed df2 and df4, we added 6 other comparatively highly correlated features to the final dataset obtained from the IMDB dataset. The 6 added features were movie_fb_likes, actor_1_fb_likes, actor_2_fb_likes, actor_3_fb_likes, cast_total_fb_likes, and budget. In Figure 7, the experiment shows that those features not only helped to reduce the error, but also enhanced the interpretability of the prediction models. Table 1 displays the average error of the model

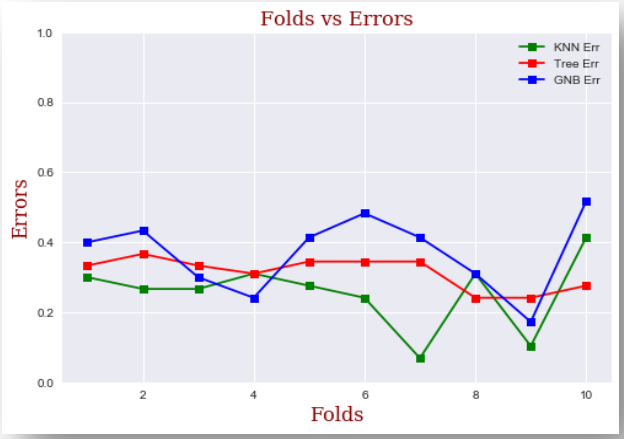

Figure 7. Classification Results (Instances -293, Features - 10)

Models	Avg. Error
KNN	27%
Decision Trees	30%
Gaussian Naïve Bayes	38%

TABLE 1: Classification Results

Clearly, it is observed that KNN has outperformed DT and GNB. The error is still high, but it has reduced quite a bit. Next, we ran feature importance using the Decision Tree classifier in Python, and sorted the output (see Table 2). In Table 2 the importance of the features are ranked from highest to lowest. For example budget is the most important feature, with an importance value of 0.253529, followed by movie_facebook_likes, with an importance value of 0.211586, etc. From Table 2, the least important feature is actor_1_facebook_likes, with an importance value of 0.052694.

Features	Importance
budget	0.253529
movie_facebook_likes	0.211586
vid_view	0.080102
vid_comment	0.076362
vid_dis_like	0.074187
vid_like	0.073310
cast_total_facebook_likes	0.064534

actor_2_facebook_likes	0.058441
actor_3_facebook_likes	0.055255
actor_1_facebook_likes	0.052694

TABLE 2: Feature Importance in Sorted Order

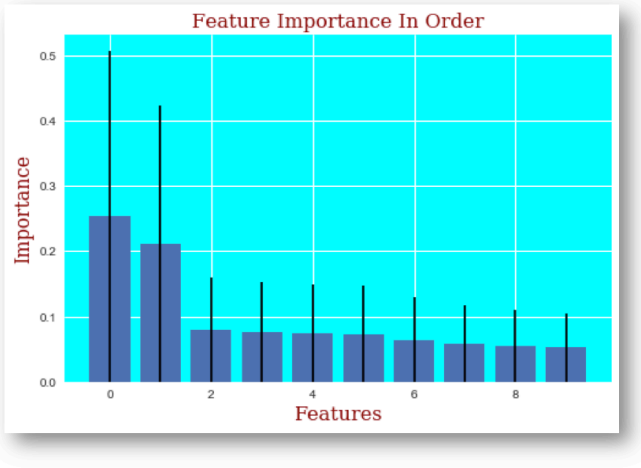

Figure 8. Feature Importance in Sorted Order

Figure 8 represents the graphical importance of the features based on Table 2.

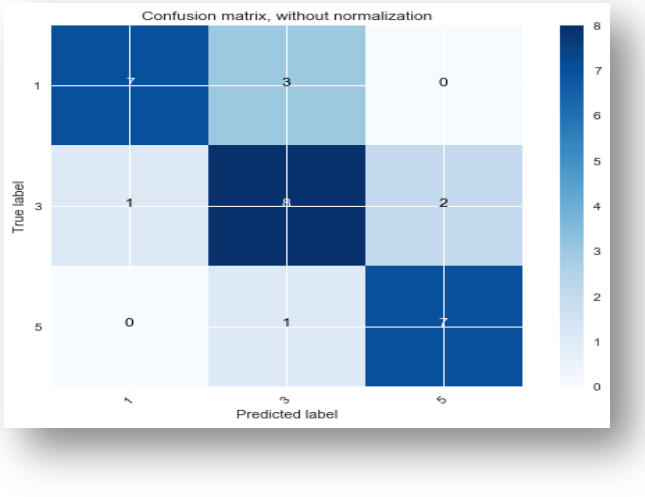

Figure 9. Confusion Matrix of Predicating Movie Success

In the confusion matrix (see Figure 9), we found that the models are doing well in the area where the true (target) label is 3 (Average Success). True (target) labels 1 (Very High Success) and 5 (Very Low Success) are causing errors in the predicted labels.

5.4 Results - K-means Clustering & Graph Theory

The unsupervised learning tasks of the study explores whether K-means Clustering or Graph Theory is the better model for group similarity considering the "success rating" and "YouTube view count" features. The performance measure used is the modularity score, which is a measurement of quality of communities inside a network of nodes, or in this case, movies. After a better model is determined, that model was used to calculate the modularity score of the full sample.

First, we used the equal-depth binning method to transform the YouTube view_count feature to a categorical variable of five categories: very high success, high success, average success, low success, and very low success. The binning was done because categorical variables are needed for the Graph Theory model.

Afterwards, the data was randomized and went through the 10-fold cross-validation process. During this process, each model was tested through 10 different samples selected from the entire dataset. The goal of the validation process is designed to see which model has the better modularity score. A Z-score is calculated to see if the mean difference in the modularity scores between the two models is statistically significant.

5.4.1 Results - Cross-Validation between K-means Clustering and Graph Theory

The result of the 10-fold cross validation with the full sample (n = 491) yielded a Z-score of 77.05, higher than 2.33 at 98 percent confidence interval. This showed that the two models are statistically different in their modularity scores. On average, the modularity score of the Graph Theory model is 0.965 higher than the K-means Clustering. Therefore, the Graph Theory model performs better than the K-means Clustering model (see Figure 10). The procedures were performed with two features, (i.e., success_rating, YouTube view_count category).

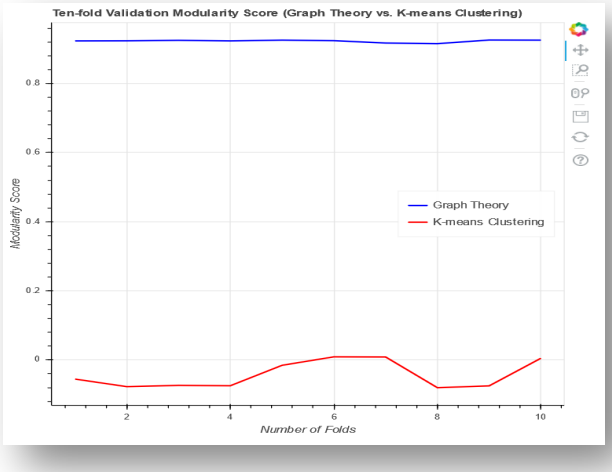

Figure 10.Moduarlity Score 10-Fold Comparison between K-Means Clustering and Graph Theory (Instances - 491, Features - 2)

A similar pattern was observed when the 10-fold cross-validation process was ran on the reduced sample (n = 293). This process resulted in a Z-score of 32.828, also higher than 2.33 at 98 percent confidence interval. The modularity score of the Graph Theory model is on average 0.913 higher than the K-means Clustering model. Therefore, the Graph Theory model performs better than the K-means Clustering model.

5.4.2 Results - Use of Graph Theory to Detect Movie Communities

Since the 10-fold cross-validation process showed that Graph Theory outperforms K-means Clustering in both the full sample (n = 491) and the reduced sample (n = 293), Graph Theory is used to detect movie communities. The analysis was performed mainly using the Python NetworkX library. The result showed that for the full sample, there are a total of 25 communities of movies with a modularity score of 0.923, while the reduced sample have 15 communities with a modularity score of 0.872 (see Figures 11 and 12).

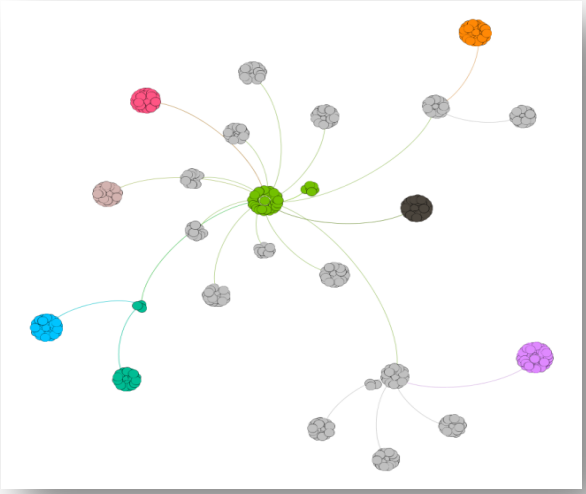

Figure 11. Layout of 25 Communities of Movies for the Full Sample (n = 491)

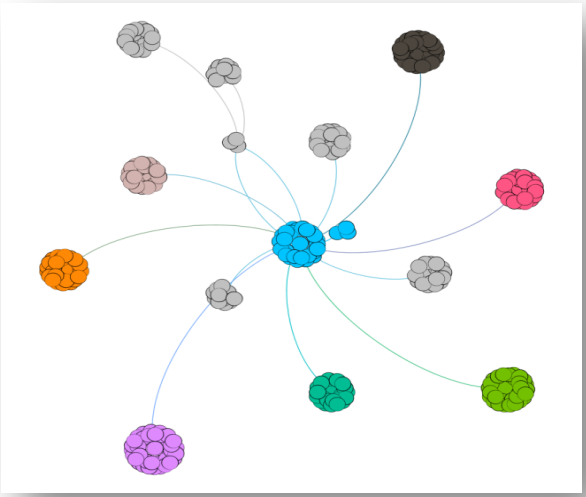

Figure 12. Layout of 15 Communities of Movies for the Reduced Sample (n = 293)

6. Conclusions

Ultimately, this paper proves that amongst the supervised learning models, KNN outperformed the DT and GNB learning models when the number of instances was decreased from 491 to 293. On the other hand, GNB performed better than the KNN and DT learning models when the number of instances was high (491 as compared to 293). Also, this research shows that these supervised learning models predict better when there are high correlations between the independent features and the target concept. For example, features such as "budget", "vid_view_counts", and "movie facebook likes" were not only highly correlated with "gross", but also reduced the prediction error of all the machine learning models examined in this research. However, the results from the unsupervised learning models conclude that, Graph Theory obtained a higher modularity score than the K-means Clustering model. This suggests that Graph Theory performed better than K-means Clustering in grouping movies with similar success rates.

Based off of the results presented in this paper, we have found a way to determine whether a movie will be successful prior to it arriving at the box office. The factors that play key roles in predicting movie success are budget, Facebook, and YouTube data. In conclusion, high movie budget, high Facebook likes, and high YouTube view counts equates to high movie success.

7. Future Scope

In the future, we would like to increase both the number of movies and features in the dataset. Such features may include movie genre, lead actor's name, and the country where the movie originated. We would also like to include other social media sources of movie data collection such as Twitter. Other learning models that we want to apply to the movie data are the following supervised learning models: Random Forest and Neural Networks. We are interested in comparing results from these models with those expressed herein. Finally, we want to apply two other unsupervised learning models called Partitioning Medoids (PAM) Clustering and CLARA-Clustering to the movie data.

8. REFERENCES

[1] K. Thompson, "Nation, national identity and the international cinema," *Film History,* vol. 8, p. 259, 1996.

[2] H. Zhou, T. Hermans, A. V. Karandikar and J. M. Rehg, "Movie genre classification via scene categorization," in *Proceedings of the 18th ACM international conference on Multimedia*, 2010.

[3] Kaggle, "Kaggle," Kaggle, 2017. [Online]. Available: https://www.kaggle.com/datasets. [Accessed 15 February 2017].

[4] YouTube, "YouTube," YouTube, 2017. [Online]. Available: YouTube.com. [Accessed 15 February 2017].

[5] T. M. Michell, Machine Learning, Redmond, WA: McGraw-Hill Science/Engineering/Math, 1997.

[6] M. Tsvetovat and A. Kouznetsov, Social Network Analysis for Startups: Finding connections on the social web, " OŔeilly Media, Inc.", 2011.

[7] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in *Proceedings of the 23rd international conference on Machine learning*, 2006.

[8] A. H. Fielding and J. F. Bell, "A review of methods for the assessment of prediction errors in conservation presence/absence models," *Environmental conservation,* vol. 24, pp. 38-49, 1997.

[9] R. Kohavi and others, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Ijcai*, 1995.

[10] I. H. Witten, E. Frank, M. A. Hall and C. J. Pal, Data Mining: Practical machine learning tools and techniques, Morgan Kaufmann, 2016.

[11] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics),* vol. 28, pp. 100-108, 1979.

[12] M. Chen, K. Konstantin, and B.K. Szymanski, "Community detection via maximization of modularity and its variants," *IEEE Transactions on Computational Social Systems,* vol. 1, pp. 46-65, 2014.

[13] T.N. Dinh and M.T. Thai, "Community detection in scale-free networks: approximation algorithms for maximizing modularity," *IEEE Journal on Selected Areas in Communications*, vol. 31, pp. 997-1006, 2013.

[14] T.N. Dinh, X. Li, and M.T. Thai, "Network Clustering via Maximizing Modularity: Approximation Algorithms and Theoretical Limits." *in Data Mining (ICDM), 2015 IEEE International Conference on*. IEEE, 2015.