A Comparative Analysis of Classification Algorithms with Respect to the Categorization of IMDb Data

Karthik Narasimhan

December 2021

1 Abstract

Give the massive cost of film production today, movie production studios want to know if the movie that they produce will be popular with the movie-going public. This paper will compare the performance of K-nearest neighbors and Support Vector Machines as to their ability to predict the IMDb rating of a film. The data used for this paper is a dataset of 1000 movies released between 2006 and 2016 inclusive and their relevant information. Support Vector Machines are primarily used for binary classification so a One-vs-Rest classifier was applied so that it could be used for multi-class classification.

2 Introduction

While filmmaking is an art form, film production is very much a business and every business must get the most essential insights from its data in order to make the most effective and profitable decisions possible. Streaming services, like Netflix and Hulu, make use of machine learning to analyze patterns in the vast amounts of data they receive from their customers' viewing habits. [12] The recommendation systems that recommend movies and TV shows to viewers based on their viewing habits are the result of highly refined classification and prediction algorithms. The more data these algorithms receive, the better it can tailor the experience of the streaming service to the customer's preferences. Netflix, in particular, says that it uses machine learning to "help shape our catalog of movies and TV shows by learning characteristics that make content successful." [11]

William Goldman, the Oscar-winning screenwriter of the films "All the President's Men" and "Butch Cassidy and the Sundance Kid" famously wrote, "Nobody knows anything... Not one person in the entire motion picture field knows for a certainty what's going to work. Every time out it's a guess and, if you're lucky, an educated one." [7] While this may have been true in his time, with

machine learning, we no longer have to be lucky to make an educated guess. Machine learning can help us to make informed predictions as to what an average movie-goer might enjoy.

The aim of this paper is to compare the performance of machine learning classification algorithms with respect to their classification of movie data from the Internet Movie Database (IMDb). This will be done by fitting K-nearest neighbors and a Support Vector Machine (SVM) on a set of training data and then predicting the rating of movies in the test data. Because the user rating will be grouped into 5 classes and SVM cannot be extended to multi-class problems [1], I will apply the One-vs-Rest classifier to SVM. After this, the accuracy, precision, recall and F1 scores will be computed. With over 8 million titles and 11 million person records [8], IMDb is the largest film and television database on the Internet. That said, IMDb charges thousands of dollars for its complete databases. Therefore, this paper will make use of a significantly reduced database of 1000 films released between 2006 and 2016 inclusive, the data for which was found on IMDb, which was downloaded from the data science website Kaggle.

This paper is organized as follows: Section 2 summarizes other research papers on the subject of movie popularity prediction using machine learning, Section 3 describes the machine learning classification algorithms (or classifiers) that I will analyze in comparison to each other, Section 4 will describe my methodology for performing this research including all applicable mathematics, Section 5 will describe my experimental results and analysis, Section 6 will present my conclusions, and Section 7 will provide an overview of potential future work regarding this subject matter.

3 Related Work

In [9], researched trained and tested Naive Bayes, Decision Trees (C4.5) and Logistic Regression with data from IMDb movie data from 2006 to 2016 inclusive to predict the average IMDb user rating of a film. They found that Logistic Regression outperformed the other two classifiers with a 89.98% accuracy.

In [10], researchers collected movie data from The Movie Database (TMDb) and Open Movie Database (OMDb), page view data for movie pages on Wikipedia as a measure of popularity, and likes, shares and comments of movie trailers on YouTube to measure buzz for a movie. The algorithms Random Forest and XGBoost were trained and tested with this data to predict the success or failure of a movie before it released. This prediction was then compared with how well this movie actually performed at the box office. Performance was measured with the movie metadata and extracted features in one set and the movie metadata, extracted features and social media data in another set. They found that XGBoost outperformed Random Forest with a higher Cross Validation Score in both sets of performance metrics.

In [3], researchers trained and tested the machine learning classification algorithms Bagging, Random Forest, J48, IBK and Naive Bayes with data from

IMDb on Hollywood movies released in 2018. They found that their original dataset was very imbalanced as there were only 3 movies rated as a flop (rating between 0 and 3.5) and 138 average movies. After applying the classifiers, the accuracy was very low. They decided to apply Synthetic Minority Over-sampling Technique which balanced the dataset. After predicting the user ratings with the test data, they found that accuracies significantly improved and that Random Forest gave the highest accuracy while Naive Bayes gave the lowest accuracy.

4 Machine Learning Classifiers

4.1 K-Nearest Neighbors

K-Nearest Neighbors (KNN) is a supervised machine learning classification algorithm. KNN is trained by selecting K number of neighbors, calculating the Euclidean distance of K number of neighbors, find the number of neighbors for each category and then assign each new data point to the category for which the number of neighbors is the greatest. KNN is tested by calculating the Euclidean distance between the test data and training points and then selecting the K number of points which is closest to the test data. KNN then calculates the probability of the test data belonging to the classes of the training data and assigns the test data to the class that has the highest probability. [5]

The formula for the Euclidean distance between A_1 and B_1 is:
$\sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2}$

4.2 Support Vector Machine

Support Vector Machine are supervised machine learning classification algorithms. The purpose of SVM is to maximize distance to the closest example of each type. We do this by maximizing the margin between the data points and the hyperplane. [6] The margin is twice the absolute value of distance b of the closest example to the hyperplane. The discriminant function for a hyperplane is $g(x) = xw$ If $g(x) = 0$, then we are on the hyperplane, $g(x) > 0$ refers to one side of the hyperplane and $g(x) < 0$ refers the other side of the hyperplane.

Let x_+^* x_-^* be the samples closest to the hyperplane (these are the support vectors) and designate that $g(x_+^*) = 1$ and $g(x_-^*) = -1$
$width = x_+^* - x_-^*$
The distance of x from the hyperplane is:
$d(x|w) = \frac{xw}{||w||}$
Therefore:
$dx_+^* = \frac{1}{||w||}$
The margin is therefore:
$margin = J(w) = d(x_+^*) - d(x_-^*) = \frac{2}{||w||}$
To minimize this function:
$J = \frac{||w||}{2} = \frac{1}{2}w^T w$

The loss attributed to a positive sample is: $max(0, 1 - x_+ w$
The loss attributed to a negative sample is: $max(0, 1 - x_- w$
Let $y_+ = 1$ and $y_- = -1$ and generalize this as: $max(0, 1 - yxw$
Our objective function is to minimize:

$$J(w) = \frac{1}{2} w^T w + C \sum_{i=1}^{N} max(0, 1 - Y_i X_i w) \tag{1}$$

where C is a blending hyperparameter. $\frac{1}{2} w^T w$ is the max margin and $C \sum_{i=1}^{N} max(0, 1 - Y_i X_i w)$ is the minimum error function or hinge loss function.
To find w, do gradient descent on the full formula.

$$\frac{dJ}{dw} = \begin{cases} w & \text{if } Y_i X_i w = 0 \\ w - c Y_i X_i^T & \text{otherwise} \end{cases} \tag{2}$$

4.3 One-vs-Rest and One-vs-One

One-vs-Rest (1vR) is a heuristic method for using binary classification algorithms for multi-class classification. The multi-class dataset is split into multiple binary classification problems and the classification algorithm is trained on each binary classification problem, [4] which generates a trained classifier for each class that the algorithm was trained on. When test data is passed to the entire model, it is considered input for all of the trained classifiers. If there is a probability that the test data belong to a certain class, the class's classifier returns a +1 while the other classifiers return a -1. The prediction is the class with the greatest probability score. [2]
One-vs-One(1v1) instead splits a multi-class classification into binary classification problems. The dataset is split into one dataset for each class vs every other class. [4]

5 Methodology

5.1 Data Acquisition

This database was downloaded from the data science website Kaggle. The data in this database was collected from IMDb. It contains information about 1000 movie titles released between 2006 and 2016 inclusive, including title, year of release, IMDb user rating, genre, description, director, actors, runtime, MPAA rating, number of IMDb votes, revenue and the metascore from Metacritic.

5.2 Data Preprocessing

Many of the fields in the database were missing. Most incomplete fields were in the Revenue column. This is because the Revenue column only included information about the revenue earned at the US box office and many of the films listed are non-American films that did not get a US release. Other times, the incorrect metascore would be listed. I decided to correct the database as much I could by using information from Metacritic and IMDb to fill in any missing fields. I decided that any rows that contained missing quantitative information would not be included in the project. After importing the data in the dataframe, I removed all rows that contained a missing cell. After doing this, 907 rows remained, which is over 90% of the original database.

5.3 Feature Extraction

I extracted only quantitative information from the original dataset into the set of features that were set to the x_data variable, except for the IMDb user rating, which was extracted to the label variable. These included Runtime, Revenue, Votes, Metascore. Although you cannot learn much about a movie from these pieces of information, they are still very much relevant. Runtime gives a rough estimate as to the production value of a film. This is because movies cost money to make, and lower-budget movies must be short out of necessity. Large budget movies can afford to be longer. Revenue tells us how much money a film made. Number of votes tell us how engaged the online film community was with the movie. Metascore tells us how much critics liked the movie.

5.4 Data Transformation

The IMDb user rating was chosen as the label for this research paper. The user rating values in the original dataset were values between 0 and 10, rounded to one decimal place. To make classification easier for the chosen classifiers, I simplified the user rating into 5 classes. Scores between 0 and 2 were changed to 1, scores between 2 and 4 were changed to 2, scores between 4 and 6 were changed to 3, scores between 6 and 8 were changed to 4 and scores between 8 and 10 were changed to 5.

6 Experiments and Results

I evaluated KNN and SVM with a One-to-Many Classifier according to accuracy, recall, precision and F1 score. Accuracy, recall, precision and F1 score are determined by the number of True Positives, False Positives, True Negatives and False Negatives.

	Predicted Negative	Predicted Positive
Actual Negative	True Negative	False Positive
Actual Positive	False Negative	True Positive

The following are formulas describing each measurement: Accuracy is the ratio of correct observations to all observations.

$$Accuracy = \frac{TruePositive+TrueNegative}{TruePositive+FalsePositive+FalseNegative+TrueNegative}$$

Precision is the ratio of correct positive observations to all positive observations.

$$Precision = \frac{TruePositive}{TruePositive+FalsePositive}$$

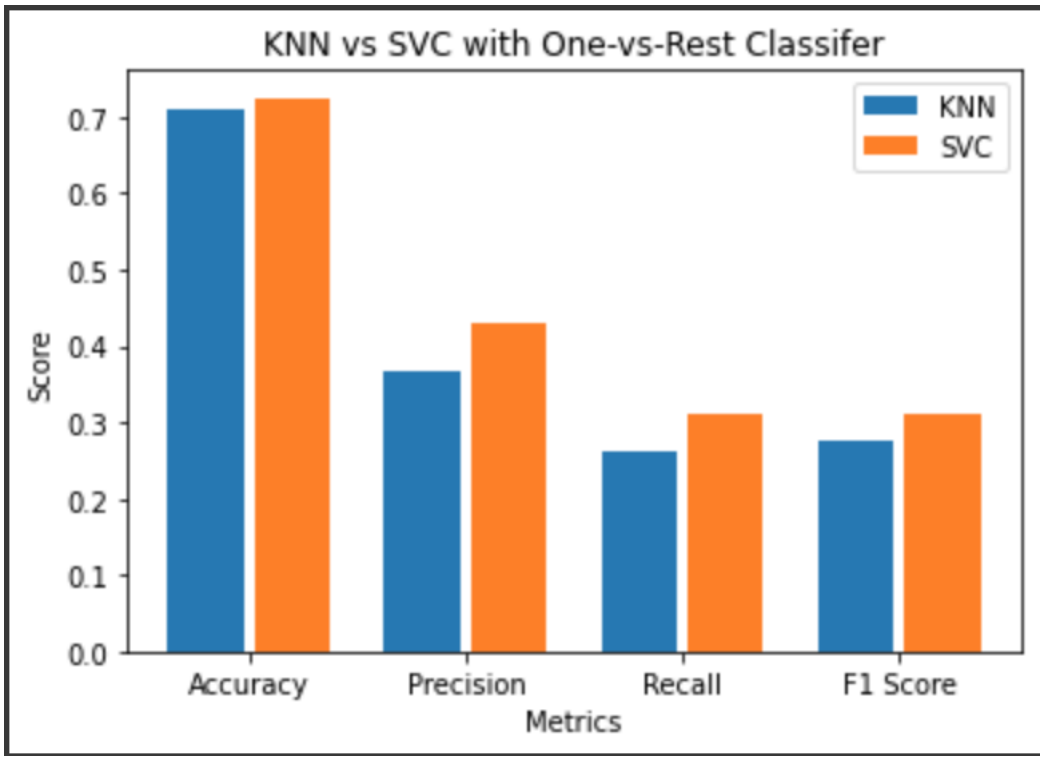
Recall is the ratio of correct positive observations to all observations.

$$Recall = \frac{TruePositive}{TruePositive+FalseNegative}$$

F1 Score is the weighted average of precision and recall.

$$F1Score = 2 * \frac{Recall*Precision}{Recall+Precision}$$

KNN received an accuracy of 70.8%, a precision score of 36.8%, a recall score of 26.3% and an F1 score of 27.6%. SVM received an accuracy of 72.5%, a precision score of 43.0%, a recall score of 31.3% and an F1 score of 30.9%.



7 Conclusions and Future Work

In this paper, I compared the performance of K-Nearest Neighbors with a Support Vector Machine with a One-to-Many Classifier applied. The results show that the SVM had a superior performance to KNN in every performance measure. I believe this is because SVM is better at handling outliers than KNN. This prediction of movie ratings can be improved by adding description as a feature. Since many movie-goers choose to see a film because its premise, it would be useful and valuable to include Description as a feature to see if it has an impact on successful movie rating prediction.

References

[1] Mustafa Murat Arat. *Multiclass Classification - One-vs-Rest / One-vs-One*. URL: `https://mmuratarat.github.io/2019-10-02/multi-class-classification`. (accessed: 12.10.2021).

[2] Amey Band. *Multi-class Classification — One-vs-All One-vs-One*. URL: `https://towardsdatascience.com/multi-class-classification-one-vs-all-one-vs-one-94daed32a87b`. (accessed: 12.10.2021).

[3] Warda Bristi, Zakia Zaman, and Nishat Sultana. "Predicting IMDb Rating of Movies by Machine Learning Techniques". In: July 2019, pp. 1–5. DOI: 10.1109/ICCCNT45670.2019.8944604.

[4] Jason Brownlee. *One-vs-Rest and One-vs-One for Multi-Class Classification*. URL: https://machinelearningmastery.com/one-vs-rest-and-one-vs-one-for-multi-class-classification/. (accessed: 12.10.2021).

[5] Antony Christopher. *K-Nearest Neighbor*. URL: https://medium.com/swlh/k-nearest-neighbor-ca2593d7a3c4. (accessed: 12.10.2021).

[6] Rohith Gandhi. *Support Vector Machine — Introduction to Machine Learning Algorithms*. URL: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47. (accessed: 12.10.2021).

[7] William Goldman. *Adventures in the Screen Trade*. Grand Central Publishing, 1989. ISBN: 0446391174.

[8] IMDb. *Press Room*. URL: https://www.imdb.com/pressroom/stats/. (accessed: 12.10.2021).

[9] Manish Jaiswal et al. "Prediction and Analysis of Movie Success with Machine Learning Approach". In: Feb. 2020.

[10] Zahabiya Mhowwala, A. Razia, and Sujala D. "Movie Rating Prediction using Ensemble Learning Algorithms". In: *International Journal of Advanced Computer Science and Applications* 11 (Jan. 2020). DOI: 10.14569/IJACSA.2020.0110849.

[11] Netflix. *Machine Learning*. URL: https://research.netflix.com/research-area/machine-learning. (accessed: 12.10.2021).

[12] Frankie Wallace. *How Data Science is Used Within the Film Industry*. URL: https://www.kdnuggets.com/2019/07/data-science-film-industry.html. (accessed: 12.10.2021).