Using Different Supervised Learning Algorithms to Predict Video Game Critic Scores

Kelvin Tiongson
kelvinjtiongson@lewisu.edu
DATA-51000-001, Summer 2020
Data Mining and Analytics
Lewis University

I. Introduction

The goal of this analysis was to use different supervised learning algorithms and identify which algorithm was best at predicting video game critic scores. The data that was used was taken from Kaggle which provided the video game sales, critic scores, user scores, ratings, and development information [1]. The data was transformed in order to correctly fit the models. The analysis uses four different supervised learning algorithms. We will be able to identify which algorithm is or is not effective.

The future sections of this report describe the dataset, methodology, results along with a discussion, and a conclusion. Section II contains a description of the dataset, frequency distribution, and descriptive statistics of the global sales, critic score, and user score of the video game. The methodology for analysis is presented in section III. The supervised learning methods chosen in this analysis were k-Nearest Neighbors, decision trees, random forest, and AdaBoost. In section IV, the reports and results of the analysis are discussed. Finally, section V provides conclusions about the methods used that are effective or not effective in predicting video game critic scores.

II. Data Description

Table I describes the dataset from Kaggle. This dataset consisted of fifteen columns. The goal of the analysis was to predict critic scores by Metacritic based on certain attributes of the data. Metacritic scores are scores that summarize the many entertainment reviews for a video game [2]. From Table I, the attributes used for modeling were the critic score, platform, rating, global sales, and the user score. The target feature was the critic score. The attributes that were used were normalized so that their scale was the same and one feature was not weighing more than another.

TABLE I. VIDEO GAME SALES

Attribute	**Type**	**Example Value**	**Description**
NAME	Nominal (string)	“Super Mario”	Name of the video game
PLATFORM	Nominal (string)	“Wii”	Video game platform
YEAR_OF_RELEASE	Numeric (real)	2011	Year the game was released
PUBLISHER	Numeric (string)	“Nintendo”	Video game publisher
NA_SALES	Numeric (real)	41.36	Video game sales in North America (in millions)
EU_SALES	Numeric (real)	28.96	Video game sales in Europe (in millions)
JP_SALES	Numeric (real)	3.77	Video game sales in Japan (in millions)
OTHER_SALES	Numeric (real)	8.45	Video game sales in other locations (in millions)
GLOBAL_SALES	Numeric (real)	82.54	Total sales (in millions)
CRITIC_SCORE	Numeric (real)	76	Aggregate score compiled by Metacritic staff
CRITIC_COUNT	Numeric (real)	51	Number of critics used in comping up with the critic score
USER_SCORE	Numeric (real)	8.3	Score by Metacritic’s subscribers
USER_COUNT	Numeric (real)	324	Number of users who gave the user score
RATING	Nominal (string)	“E”	ESRB rating

Table II shows the descriptive statistics of the columns of the data set. Rather than looking at all of the sales of the video game, we focused on the global sales column from Table I. From Table II, the average year the games were sold were around 2006. The average global sales of a video game can be expected to be around 0.52 million. The video game sales are closely populated to the mean, but the maximum sale shows that there is one outlier. The critic and user score quartiles show that most games are close to the mean but there are some games that have a really low score. The number of user counts show that there are a lot of users who like to give their input on a video game. There are more user counts than critic counts.

TABLE II. DESCRIPTIVE STATISTICS OF THE NUMERIC FEATURES

	Year_of_Release	**Global_Sales**	**Critic_Score**	**Critic_Count**	**User_Score**	**User_Count**
count	17408	17416	8336	8336	7798	7798
mean	2006.63	0.52	68.91	26.19	7.12	162.67
std	5.90	1.53	13.96	18.99	1.50	562.84
min	1976	0.01	13	3	0	4
25%	2003	0.05	60	11	6.4	10
50%	2008	0.16	71	21	7.5	25
75%	2011	0.45	79	36	8.2	81
max	2017	82.54	98	113	9.7	10766

III. METHODOLOGY

In order to prepare my data for the algorithms, I had to transform the attributes that were being used. The attributes that were used from Table II were global sales, critic score, platform, user score, and game rating. All of the platforms were able to be categorized into four platforms. Most of the platforms were able to be categorized under "Nintendo", "PlayStation", or "Xbox". The platforms that did not belong under the three company platforms were given a value of "Other". From Table III, we are able see that there are seven different values for the game rating attribute. However, three of the ratings could be labeled as outliers. "E10+" is a rating for games that is meant for everyone ages 10 and up. "K-A" is a game rating that is for kids to adults [3]. "RP" is a game rating that is pending, and "AO" is a game rating that is for adults only [4]. I replaced all "E10+" ratings with "E" since they were both ratings for games that are playable for everyone. I replaced "RP" ratings with "T" since "T" is the mode. "AO" was replaced with "M" for mature rated games and "K-A" was replaced with "T" for teen rated games.

TABLE III. COUNTS OF RATING ATTRIBUTE

Rating	**Count**
T	2489
E	2162
M	1489
E10+	968
RP	2
AO	1
K-A	1

The values of critic score varied between ten and one hundred. The values of the user scores varied between one and ten. Metacritic codes their scores into three bins. Games scored 75 and above are considered "good", games scored between 50-74 are "average", and games below 49 are "bad" [5]. So, I replaced the values of the critic score with their respective Metacritic label. I did the same thing for the user scores. There were several data that were missing. When doing the analysis, I focused on rows that did not have any missing data. Although there was a big amount of missing data, I was still left with more than seven thousand rows of data. The dataset was split into two sets. One set was the training set and the other being the testing set. The training set was seventy percent of the original dataset and the testing set was thirty percent of the original dataset.

Finally, after cleaning the data and splitting the dataset, I was able to model the training set with the supervised learning algorithms. Starting out, I used the constant model method to get a baseline standard for implementing the rest of the supervised learning model methods. The constant model gave me a classification score of 0.51. Moving forward, we know that the rest of the models should perform similar to or greater than 0.51. The next algorithm that was implemented was the k-Nearest Neighbors algorithm. When implementing k-Nearest Neighbors on the data the distance metric that was chosen was Euclidean distance. The initial k chosen was 3 which gave a classification accuracy of 0.638. In order to try and improve the classification accuracy, I changed the k all the way up to k = 20. The best score I received was k = 15 with a classification accuracy of 0.669. When implementing the decision tree algorithm, the only parameter I had to change to give me the best classification accuracy was the minimum number of instances in leaves. This number was set to 25, which gave me a classification accuracy of 0.672. When implementing random forest, I had to adjust the number of trees and growth control. The number of trees that gave me the best classification accuracy was 50. The growth control parameter that was adjusted was not allowing the model to split subsets smaller than 49. Adjusting these two numbers for random forest gave me a classification accuracy of 0.675. This was the best accuracy score among the group. The last model method that was used was AdaBoost. This modeling method was the worst of the other supervised learning methods. The best accuracy score that I could obtained with AdaBoost was 0.612. This was created by adjusting the learning rate parameter to 0.99998. The classification score of each model is shown in Figure 1.

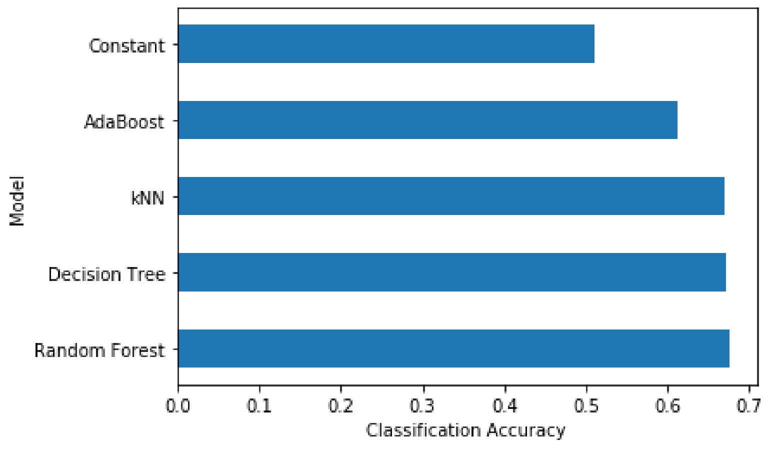

Figure 1. Classification accuracy with the training set

IV. Results and Discussion

From Figure 1, it is evident that random forest was our best model in terms of predicting video game critic scores. Decision tree and k-Nearest Neighbors were not too far behind, while AdaBoost and Constant did not perform as well as the others. Constant was the baseline model, so it was assumed for this to be the worst of the models. After modeling the training set, I then used the model methods with their adjusted parameters on the test set. Figure 2 shows the performance of each model with the classification accuracy. Similar to Figure 1, random forest was the best model on the test set with a classification accuracy of 0.686. The random forest model on the test set performed 0.13 better than on the training set. However, the k-Nearest Neighbors model was 0.03 better at 0.671 than the decision tree at 0.668. Then the last two models were the same as Figure 1 with AdaBoost coming in fourth and Constant being last.

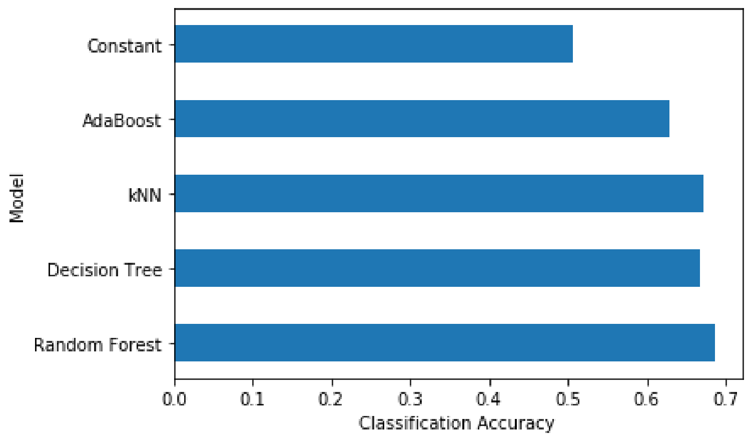

Figure 2. Classification accuracy with the test set

Figure 3, 4, and 5 shows the ROC analysis of each model classifying the target feature as average, bad, or good. Figure 3 shows the ROC analysis for classifying the target as average. Figure 4 shows the ROC analysis for classifying the target as bad. Figure 5 shows the ROC analysis for classifying the target as good. For each figure, there are five lines that represent the five models used in the analysis. Their legend is represented in Table III. In Table III, the color represents the model that matches with Figures 3, 4, 5. From the three figures, you can see that the area under the curve favors more towards random forest. This supports the classification accuracy that random forest is the ideal model method of choice between the methods used. Figures 3, 4, and 5 also show that the random forest model will perform the best between the other four models used in this analysis. However, decision tree and k-Nearest Neighbors are not far off from random forest.

TABLE IV. LEGEND FOR FIGURE 3, 4, 5

Color	**Model**
🟩	Constant
🟩	AdaBoost
🟧	kNN
🟪	Decision Tree
🟥	Random Forest

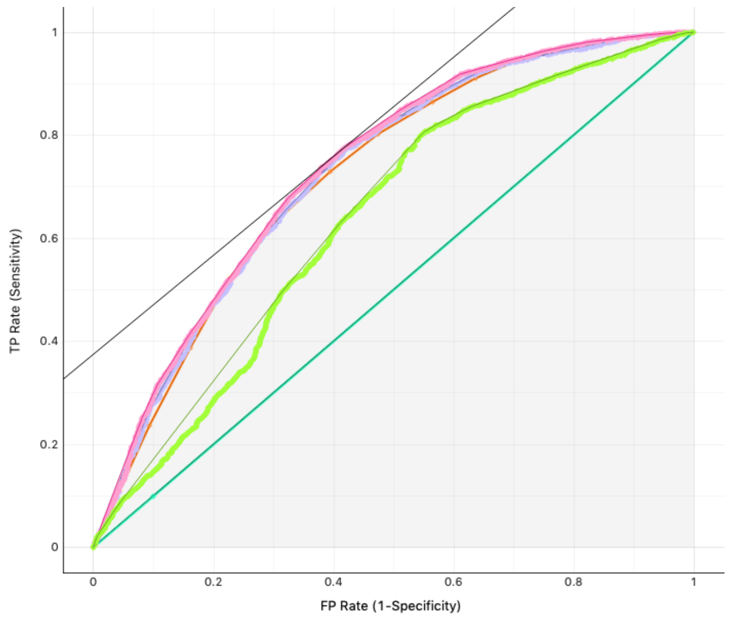

Figure 3. ROC analysis for “average”

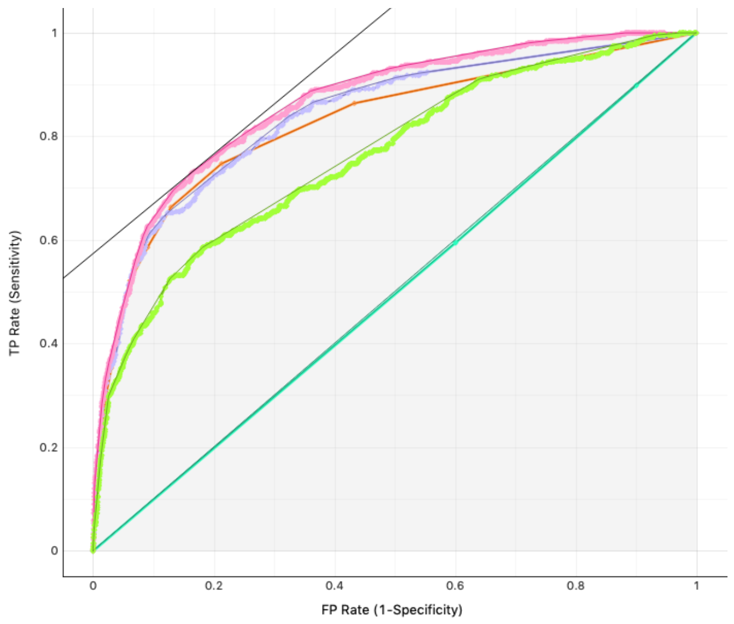

Figure 4. ROC analysis for “bad”

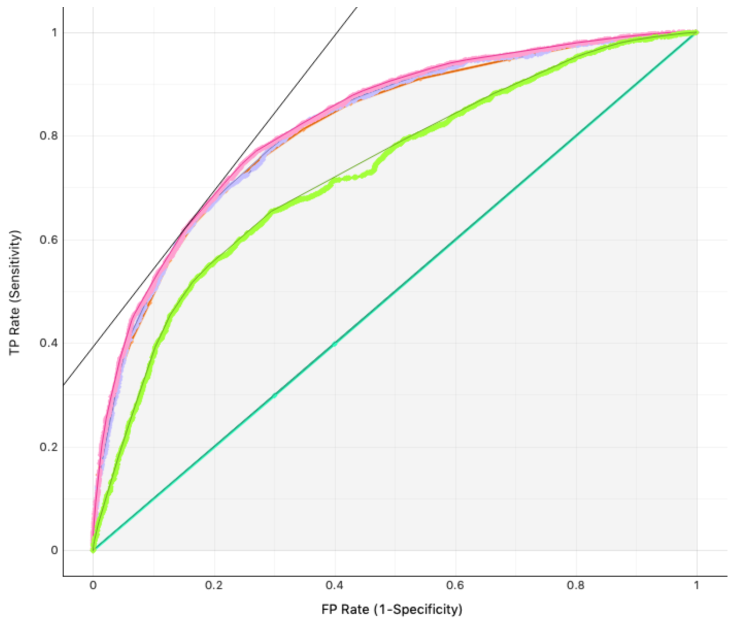

Figure 5. ROC analysis for “good”

V. Conclusions

I computed the descriptive statistics of the numerical attributes. In order to model the data, I transformed the data to so that the categories would fit within the respective model methods. I normalized the global sales attribute so that the attribute would not outweigh other attributes. I categorized the platform attribute so that the values are Nintendo, PlayStation, Xbox or Other. I also categorized the game rating attribute so that the values would take only one of the three values M, T, or E. I gave the critic score and user score values that matched up with Metacritic’s video game rating of average, bad, or good. Then I split the data into two sets. Seventy percent of the dataset were used as the training set. The remaining thirty percent were used as the test set. After this, the data was ready to be modeled. When modeling the data, certain parameters were adjusted for each in order to obtain the highest classification accuracy. Once I was able to grab the highest classification accuracy with the training set, I used the same models on the test set. Finally, I graphed the ROC analysis when classifying the target as average, bad, or good.

Random forest was the model with highest classification accuracy on the training set and test set. Decision trees and k-Nearest Neighbors were right behind random forest in terms of their classification accuracy. For both training and test set, AdaBoost had the second to worst classification accuracy and Constant had the worst. The ROC analysis for each classification of the critic score supplemented that random forest was the best algorithm to use to predict a critic score as average, bad, or good. From the four supervised learning algorithms used, we can expect AdaBoost to not be effective. Decision trees and k-Nearest Neighbors will be effective but not as effective as random forest. Random forest will be the most effective supervised learning algorithm.

References

[1] https://www.kaggle.com/kendallgillies/video-game-sales-and-ratings
[2] https://www.metacritic.com/about-metacritic
[3] https://videogamegeek.com/videogamerating/2220/esrb-k
[4] https://www.esrb.org/ratings-guide/
[5] https://www.metacritic.com/about-metascores