Introduction:

This study presents models for an easier prediction of the post-release success or failure of a movie using machine learning approaches. Over the years, a significant number of movies are being released, for which it can be overwhelming for the viewers to make a particular choice. The objective of this analysis is to make it simpler for them. On top of that, filmmakers would be able to comprehend the public demands, as it highlights the features that make a movie successful.

Using data, collected from past experiences has helped to train the models for making as accurate predictions as possible. Here, the thoughts of the consumers who have previously watched those movies are helping to make an average viewpoint for the betterment of the film industries worldwide. With the help of it, viewers can avoid the disappointment of unpleasant entertainment, while directors can acquire knowledge about the type of content people enjoy and appreciate.

In this work, various data about movies are key to finding whether they are successful or not. The dataset contains data, such as movie titles, casts, directors, genre, ratings, revenue, releasing year, runtime etc of enlisted movies. Other than title, genre, directors and actors, the rest of the data are in numerical form. After preprocessing categorical and numerical data, Decision Tree, Random Forest and Naive Bayes models are implemented for classification and prediction. Later on, the results are compared to analyze their individual outcome. This study benefits both viewers and movie makers with their time and resources as well as encourages the creation of more quality content.

Dataset Description:

32 variables are included in the dataset, which includes 839 films from 20 years of cinema. There are hundreds of performers, actors and directors names. The remaining 31 variables can serve as

predictors, while "Success" serves as the label. Here, categorical columns are Title, Director and Actors. Others are numerical. We encoded the genre to numerical for better visualization.

Variable Name	**Description**
Title	The movie's name.
genres	Categories for movies like "Animation," "Comedy," "Romance," "Horror," "Sci-Fi," "Action," and "Family"
Director	Name of the Film's Director
Actor	Actors in the film
Year	The calendar year that the film debuts (1916:2016)
Runtime	Time in minutes
Rating	The film's rating
Votes	The number of voters for the film

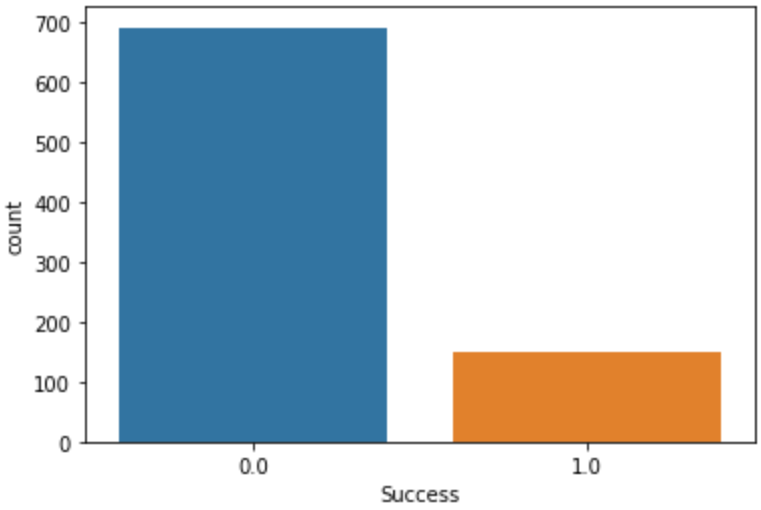
Revenue	How much revenue did the movie generate?
Metascore	Aggregated review of the film
Action	Is the movie from the action genre?
Music	Number of music in film
Musical	Is the movie musical?
Mystery	Is the movie mysterical?
Romance	Is the movie romantic?
Sci-Fi	Is the movie science fictional?
Sports	Is the movie from the sports genre?
Thriller	Is the movie from the thriller genre?
War	Is the movie from the war genre?

Western	Is the movie from the western genre?
Success	Is the movie successful?

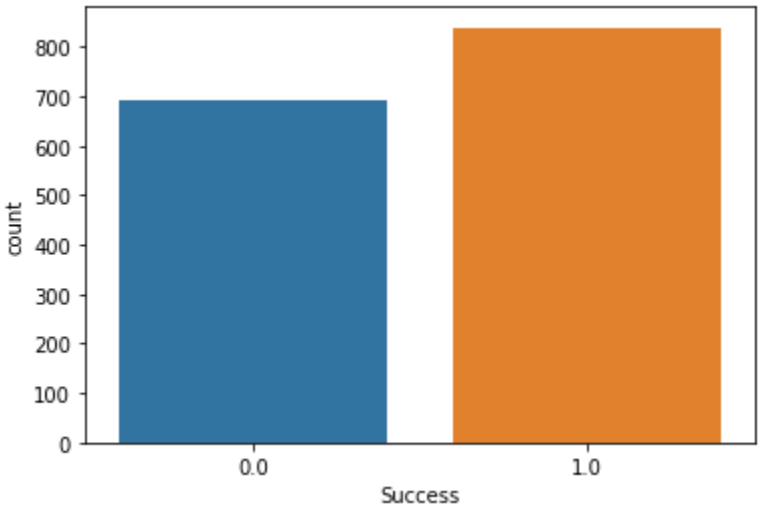
Data Preprocessing:

After importing data, the next step is to preprocess the data. Raw data may contain None value, invalid value for our prediction which is unsuitable for our study. So, data preprocessing is an important step in any data analysis study.

In our model, The dataset downloaded for this prediction has the following multiple features. Some are categorical and others are numerical values. That's why we need to preprocess the data very carefully. Firstly, we visualize the dataset to get a good understanding of it. We see 'Rating based on years', 'Votes based on years', 'Revenue based on years'. After that, we check which columns have null values. We find the columns with null values and as our dataset is small in size, we fill the null values with median values. Now, we have to preprocess the label column. We find that the **label** column **'Success'** has 0 for 690 times and 1 for 149 times. (Here 0 refers to failure and 1 refers to Success).



As the success and failure numbers are not equal, we try to make it close to equal. After that,



Then,we find the **correlation** matrix. From that, we find that **Runtime, Rating, Votes, Revenue, MetaScore** columns are highly correlated with success than the others. So we drop other columns. Also, we only take rows with more than 50 Votes.

Split data into training and test sets:

To split our training dataset and create a test dataset, we will use Scikit-learn's train_test_split() method where we will dedicate **20%** of the training samples to the test set.

Model Used:

Decision Tree:

Decision trees are powerful and popular tools for classification and prediction.

It is a classifier in the form of a tree structure which has a decision node, leaf node, edge and path. Here, leaf nodes specify the target attribute's value.

Basically, we take the feature as the root node which gives the lowest impurity (lowest Gini index), Here,

Gini index = 1 - summation of (Pi)^2 where i>=1 and i<=n

In our model, We have used GridSearchCV with the Decision Tree because passing all sets of hyperparameters manually is tough for us. Hence, GridsearchCV passes all combinations of hyperparameters one by one into the model and checks the result. In the end, it returns us the best result. We have used,

max_depth=list of range 10 to 15 and max_features = list of range 0 to 5

Random Forest:

Random forests are a supervised Machine learning algorithm that is widely used in Machine learning. Random forests basically build a number of Decision Trees. It is basically a set of decision tree.

In our model, the result is not relying on one particular decision tree, rather it checks predictions from each tree and based on that, it predicts the final output.

So, basically the workflow of our code of ***Random Forest***, when we give the ***movies_success*** dataset to the random forest classifier. Then it divides the dataset into subsets and is given to each decision tree. From each decision tree we get a prediction result, then based on the majority

of results, the Random Forest classifier predicts the final decision for our movie's success prediction.

Naive Bayes:

Third model we have used is Naive Bayes. It predicts a feature on the basis of all other feature's occurrences by using the principle of ***bayes theorem.***

$$P(A|B) = (P(B|A)*P(A)) / P(B)$$

Here,

P(A|B) Posterior probability: Probability of hypothesis A on the observed event B.

P(B|A) Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.

P(A) Prior Probability: Probability of hypothesis before observing the evidence.

P(B) Marginal Probability: Probability of Evidence.

In our model, we have used the ***GaussianNB*** classifier to fit it to the training dataset.

After every model, we find the accuracy, recall, precession and f1 score to measure the performance.

Performance:

The easiest performance metric to understand is accuracy. Accuracy indicates whether a series of measurements are correct on average

$$Accuracy = (TP + TN)/(TP + TN + FP + FN)$$

Precision is the proportion of accurately predicted positive observations to all of the expected positive observations

$$Precision = TP/(TP + FP)$$

Recall is the ratio of correctly predicted positive observations to all observations in the 999actual class.

$$Recall = TP / (TP + FN)$$

There is a difference between Precision and Recall

The F1-score combines the precision and recall of a classifier into a single metric by taking their harmonic mean. If a classifier has a higher recall value and another has a higher precision value, then F1 can measure which one is better,

$$Fl\ score = (2*Precision*Recall)/(Precision + Recall)$$

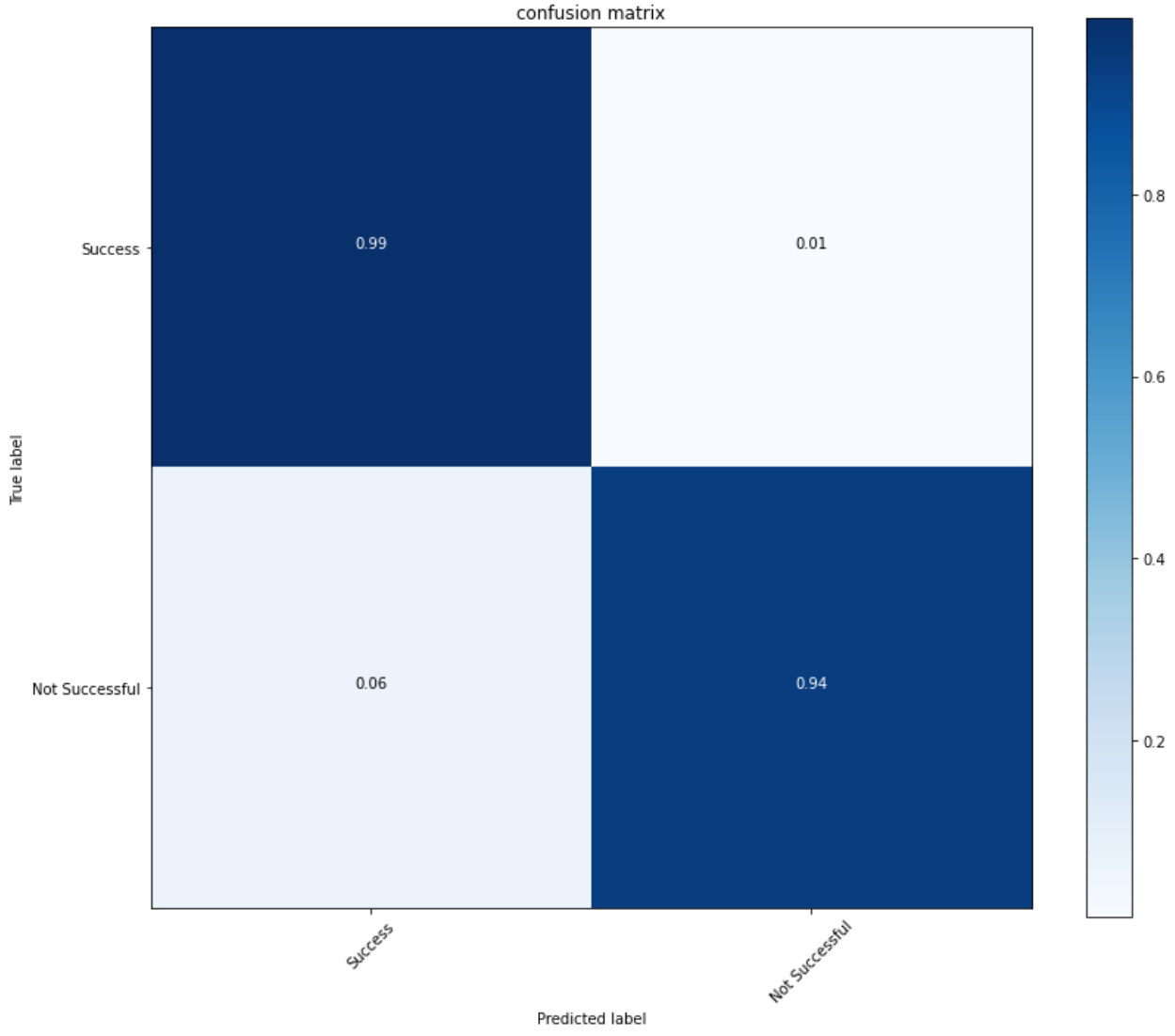
Here, TP represents true positive, TN refers to true negative, FP and FN means false positive and false negative respectively.

Model	**Accuracy(%)**	**Recall(%)**	**Precision(%)**	**F1 Score(%)**
Decision Tree	98.21	96.4	98.2	97
Random Forest	99.4	98.4	99.4	99
Naive Bayes	89.88	87.6	89.9	84.5

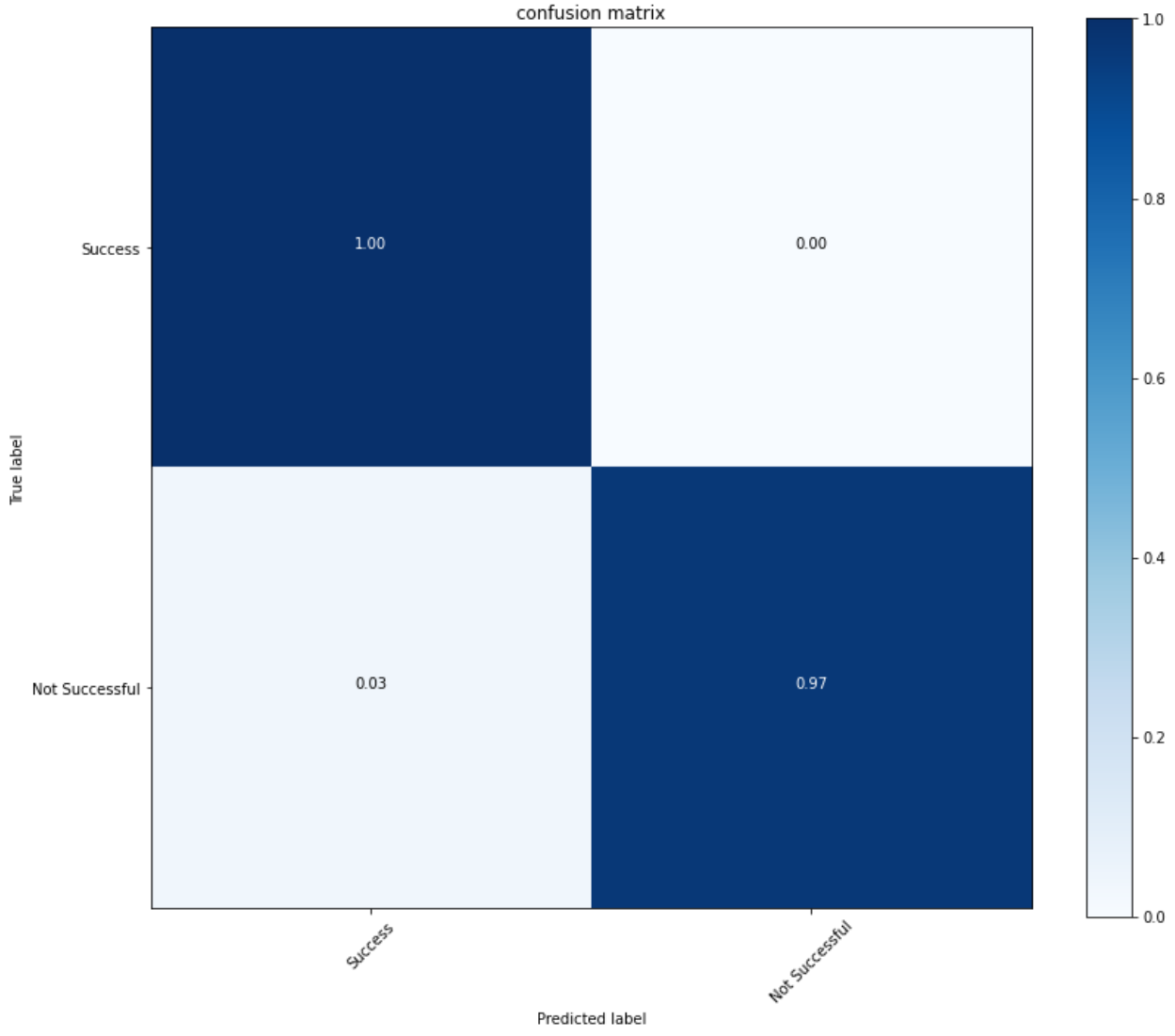
Confusion Matrix:

Confusion matrix is basically a describing technique for the performance of a classification algorithm. Calculating the confusion matrix for each model, we get a better idea of what our three classification models are getting right and what types of errors it is making.

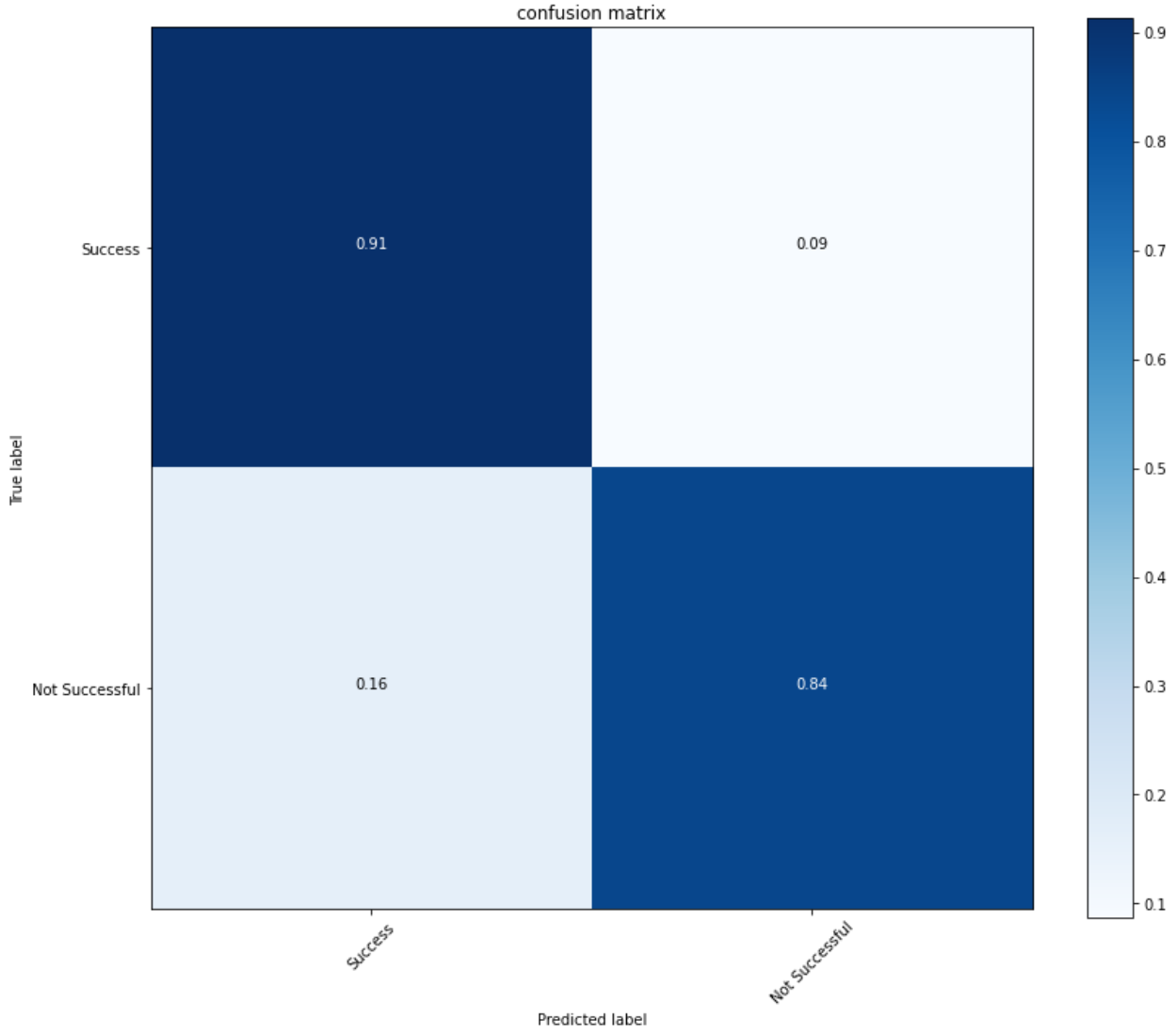
Confusion Matrix of Decision Tree:



Confusion Matrix of Random Forest:



Confusion Matrix of Naive Bayes:

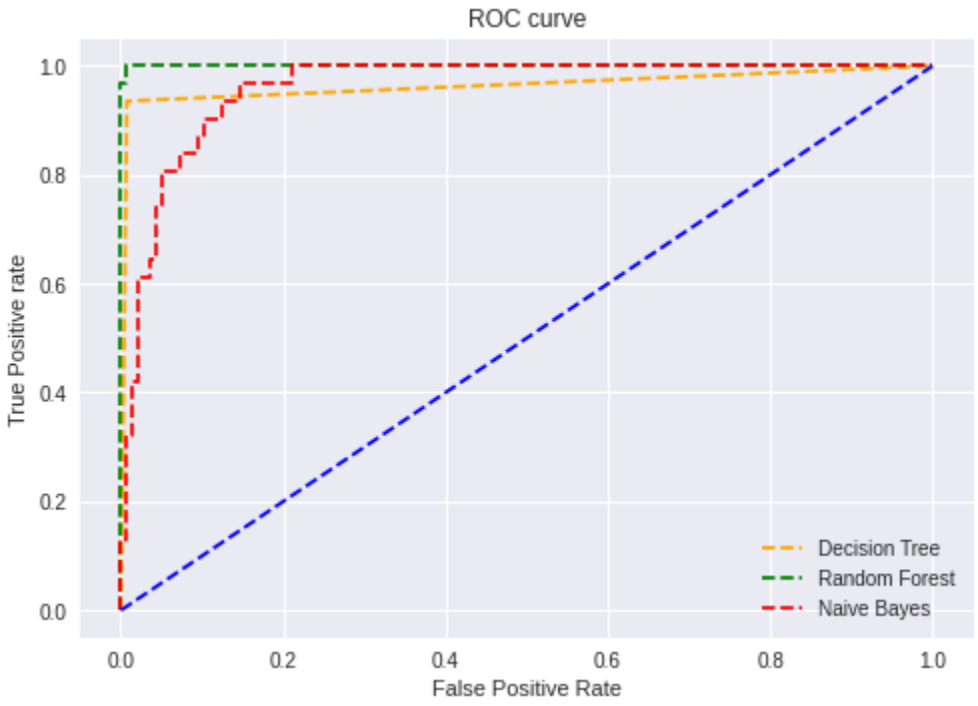


AUC and ROC:

AUC VALUES (Area Under Curve):

Decision Tree(%)	Random Forest(%)	Naive Bayes(%)
96.41	99.97	95.99

ROC CURVE (Error Curve):



Comparison:

From the Area under the ROC curve we can see, Random forest gives us the best result. The error curve or the ROC curve also supports the statement of Random forest being the best model among the models we used.

References:

D. (2022, April 10). ***How to predict the success of a movie using data analytics?*** **DataUntold.** https://datauntold.com/predict-movie-success/

D. (2022, April 10). ***How to predict the success of a movie using data analytics?*** **DataUntold.** https://datauntold.com/predict-movie-success/

Solutions, E. (2016, November 11). ***Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures*****. Exsilio Blog.** https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/

sklearn.model_selection.GridSearchCV**. (2007–2022). Scikit-Learn.** https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html?fbclid=IwAR2pGfHhjeYbhZt2tr12IyIfAtY_Ty0YGoA7PZ0En3mCTSrkylMkd1OktAU

Yiu, T. (2021, December 10). ***Understanding Random Forest - Towards Data Science*****. Medium.** https://towardsdatascience.com/understanding-random-forest-58381e0602d2?fbclid=IwAR3bPLXIhlimAscRpzHkL-U9-ZUwAoQJ7ZdDWIAIihVSvorDm51Ngmhgqc4

Just a moment. . . **(2021, January 13). Machine Learning Mastery.** https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/?fbclid=IwAR3MP3mAfn-RLHKXqjxtxzexrRxQamiCbSdx60QfHT53Z0nS8gIh-119_48#:%7E:text=ROC%20Curves%20and%20AUC%20in%20Python,-We%20can%20plot&text=The%20AUC%20for%20the%20ROC,probabilities%20for%20the%201%20class