Group 77 Final Report

Movie Judge

View on GitHub Download .zip Download .tar.gz

Group 77 Final Report

Introduction/Background

We want to create a model that can take in information such as the summary/synopsis of the movie, director name, potential cast, genre, themes, and projected release date to derive whether it will be a hit (IMDB >= 6.5) or a miss (IMDB < 6.5). The data set below that have been found on Kaggle have this information along with ratings that can be used for training. As detailed in similar projects [1, 2], data processing for this project is crucial [4] and a canonical algorithm like random forest is best suited for this task [5].

Kaggle IMDb Data Set

Problem Definition/Motivation

Each year, thousands of movies are released. Movies are normally pitched with an idea/synopsis, a full script, or just a well-known name with a good track record backs it. Currently, studio execs mainly evaluate these pitches based on feel and potentially some good movies are being passed up for flops; we want to prevent this.

Methods

Pre-Processing

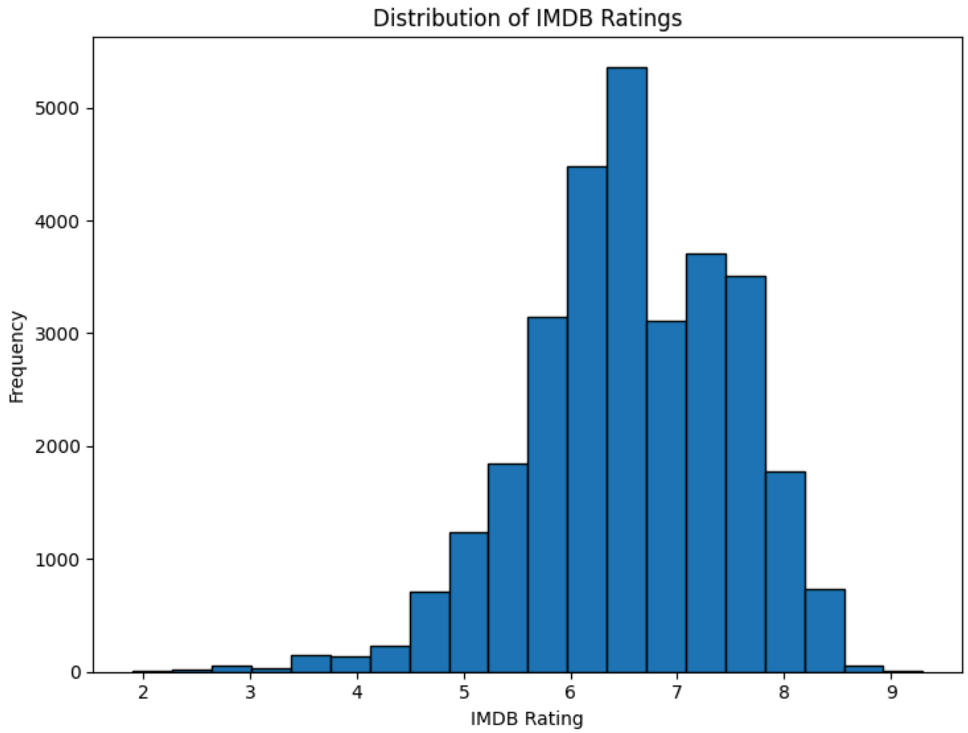
The original data set has approximately 1 million data points and 42 columns. Our initial step was to prune the data. Irrelevent columns like “did the movie have a poster”, “budget”, and “revenue” were dropped. This reduced the dataset to 11 columns. Additionally, the target variable (IMDB Rating) was transformed to a binary representation (0 or 1) with 0 being ratings below 6.5 and 1 being ratings greater or equal to 6.5.

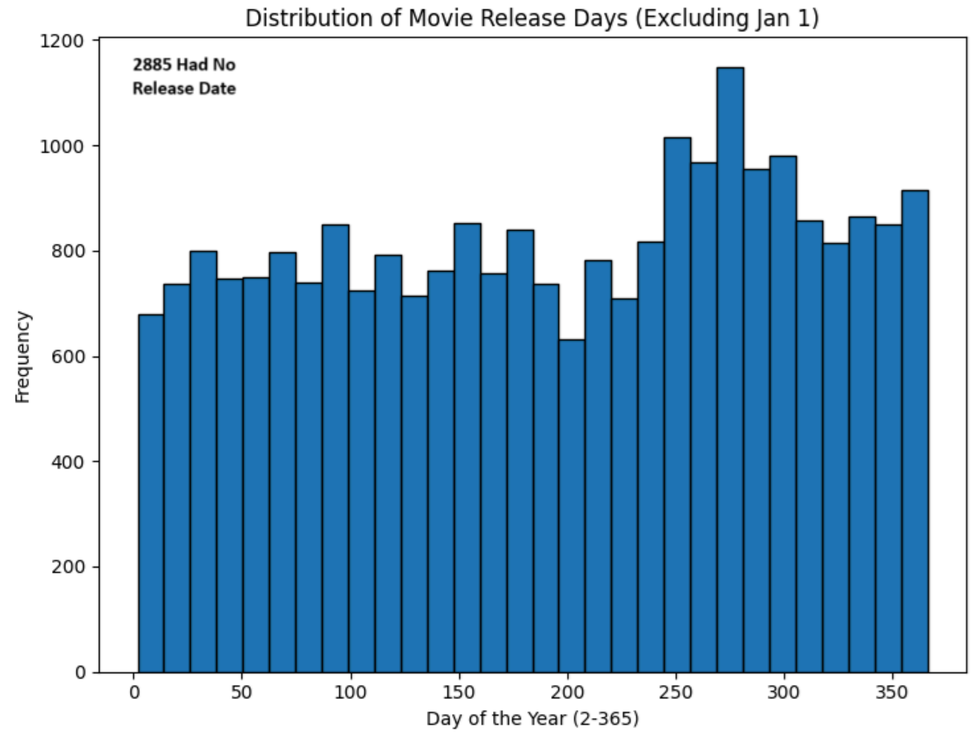
Handling Missing Values

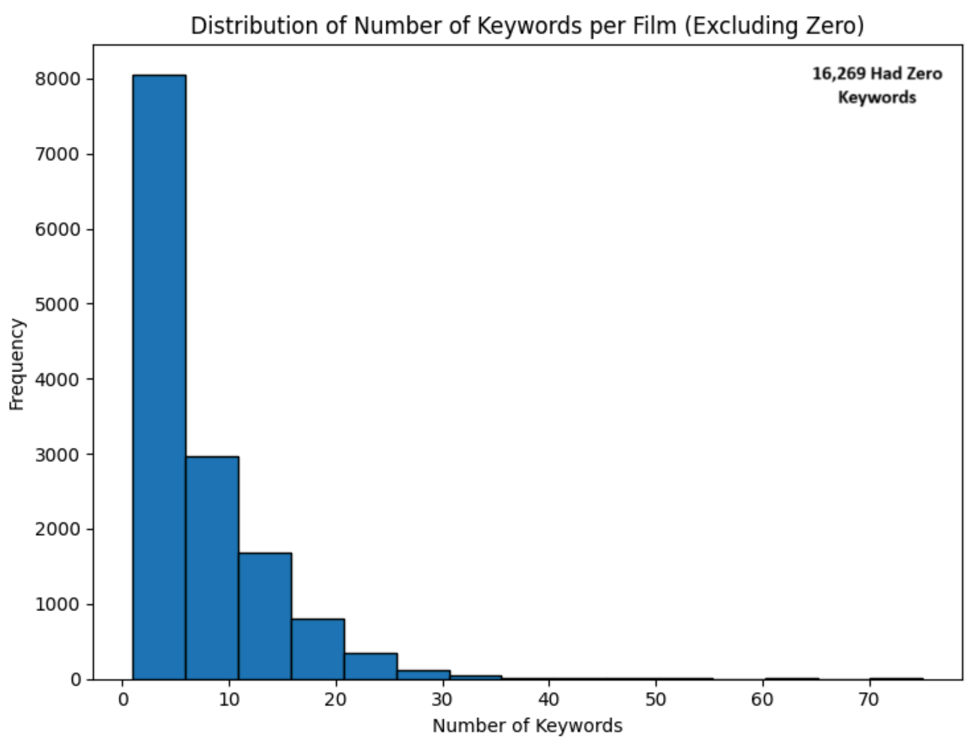
The dataset had a lot of missing data. The dataset was a large sum of various movies across the world; some titles were not registering properly and had missing crucial data such as a rating, overview, or production info. To isolate movies that had enough data we found the “Star1” column to be useful. Movies that had this metric tended to have all other values as well. By cutting rows that did not have a “Star1” value, the dataset was reduced to approximately 30k values.

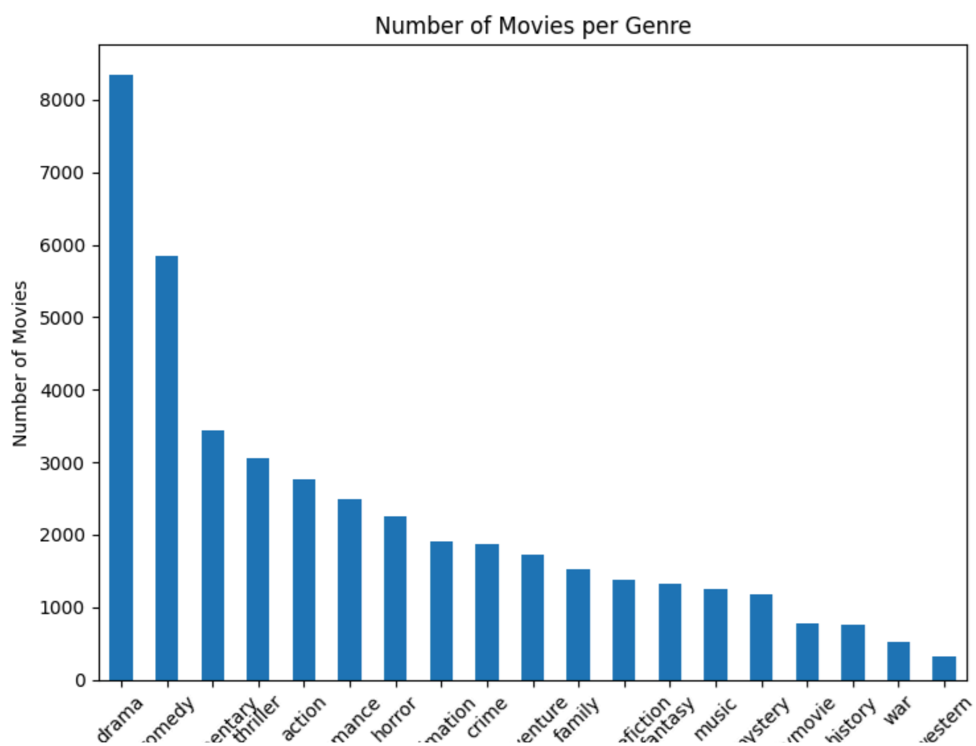
Visualizing the Data

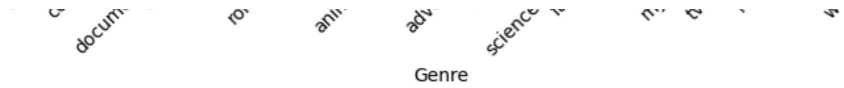
To get an understanding of the data and spread of values in the 30k dataset, several distributions were graphed.


Distribution of IMDB Ratings
Frequency
IMDB Rating
0
1000
2000
3000
4000
5000
2
3
4
5
6
7
8
9


Distribution of Movie Release Days (Excluding Jan 1)
2885 Had No
Release Date
Frequency
Day of the Year (2-365)
0
200
400
600
800
1000
1200
0
50
100
150
200
250
300
350


Distribution of Number of Keywords per Film (Excluding Zero)
16,269 Had Zero
Keywords
Frequency
Number of Keywords
0
1000
2000
3000
4000
5000
6000
7000
8000
0
10
20
30
40
50
60
70


Number of Movies per Genre
Number of Movies
0
1000
2000
3000
4000
5000
6000
7000
8000
drama
thriller
action
horror
crime
family
fantasy
music
history
war



Target Encoding

Single value columns (‘Director’, ‘Star1’, ‘Star2’, ‘Writer’, ‘Music_Composer’) were target encoded due to the large number of possible values.

Multi-Category Target Encoding

Multi-category columns (‘genres_list’, ‘keywords’) were encoded individually by splitting arrays up and encoding. For instance, genres were split into their types (action, comedy, adventure, etc) and then encoded for each movie.

String Embeddings

For the movie overviews, string embeddings were used. From the Sentence Transformer package, the bert-base-nli-mean-tokens transformer was used.

The Models

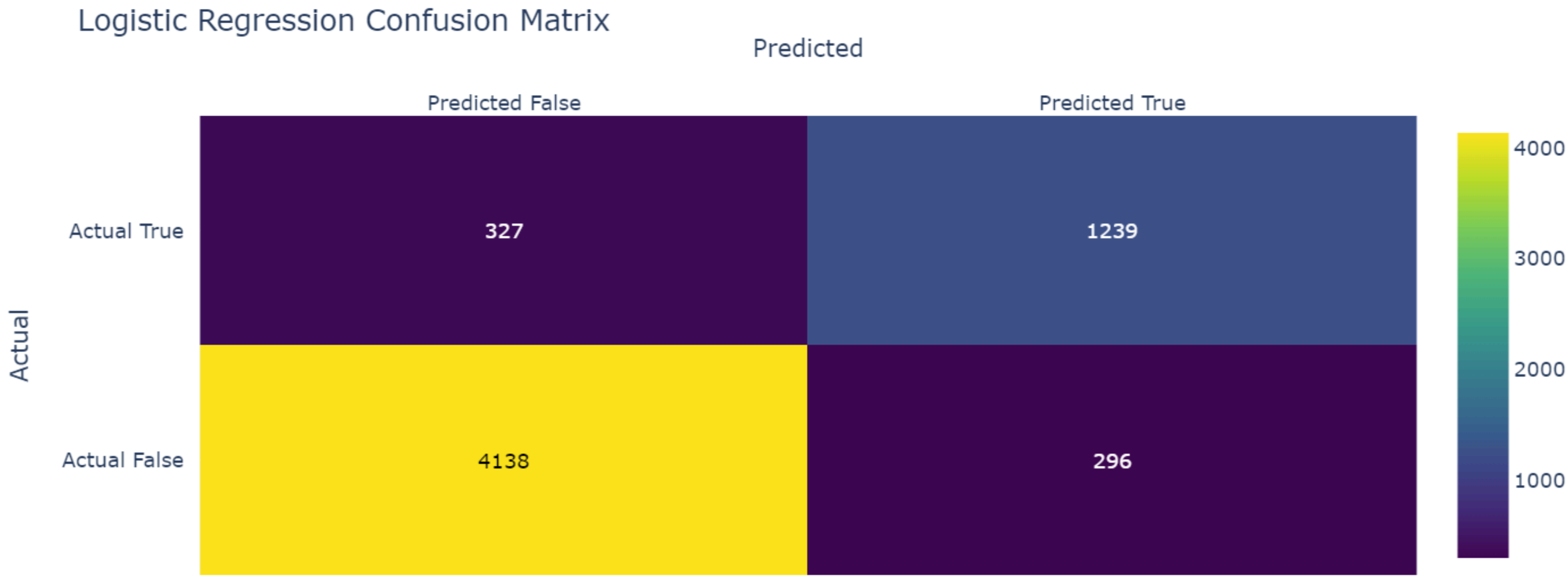
For the actual model implementations we used logistic regression to implement a binary classification. This was done with the scikit-learn logistic regression package [3].

Results

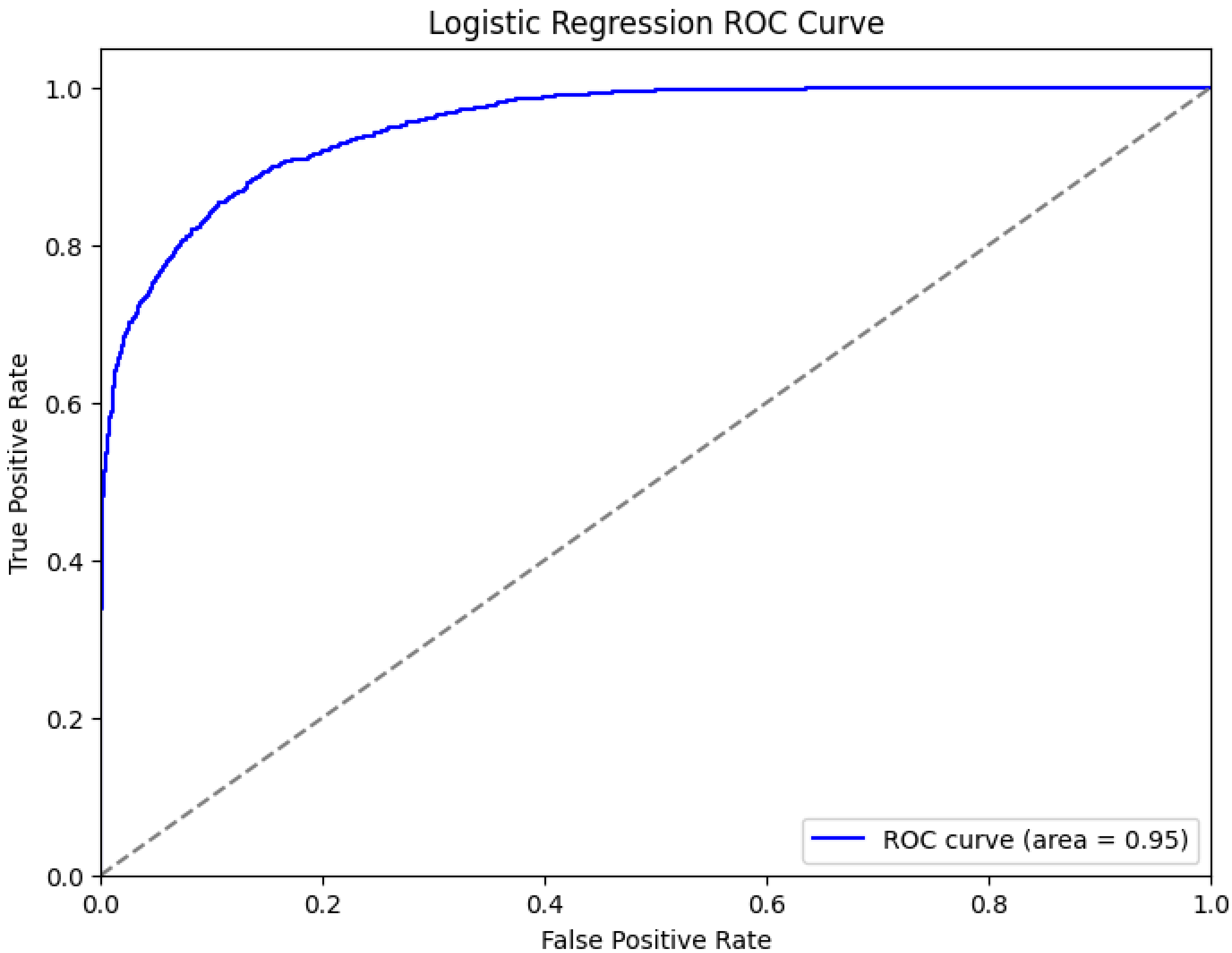
Logistic Regression

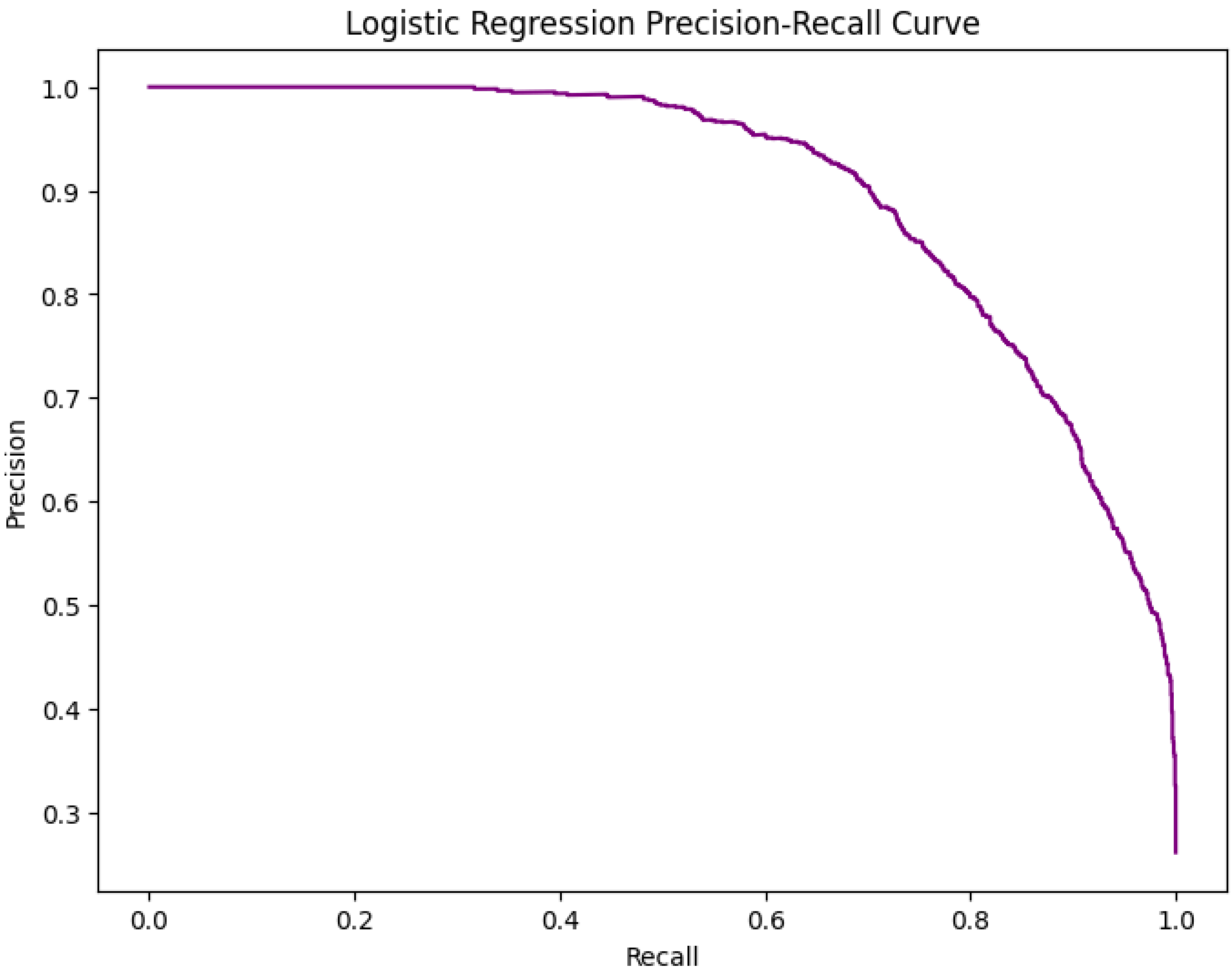
After training and fitting our model with the processed dataset, we got an overall accuracy of 89.6%.

The results are represented in the a confusion matrix below.



To further assess the robustness of the model, we plotted a ROC curve and precision recall curve.

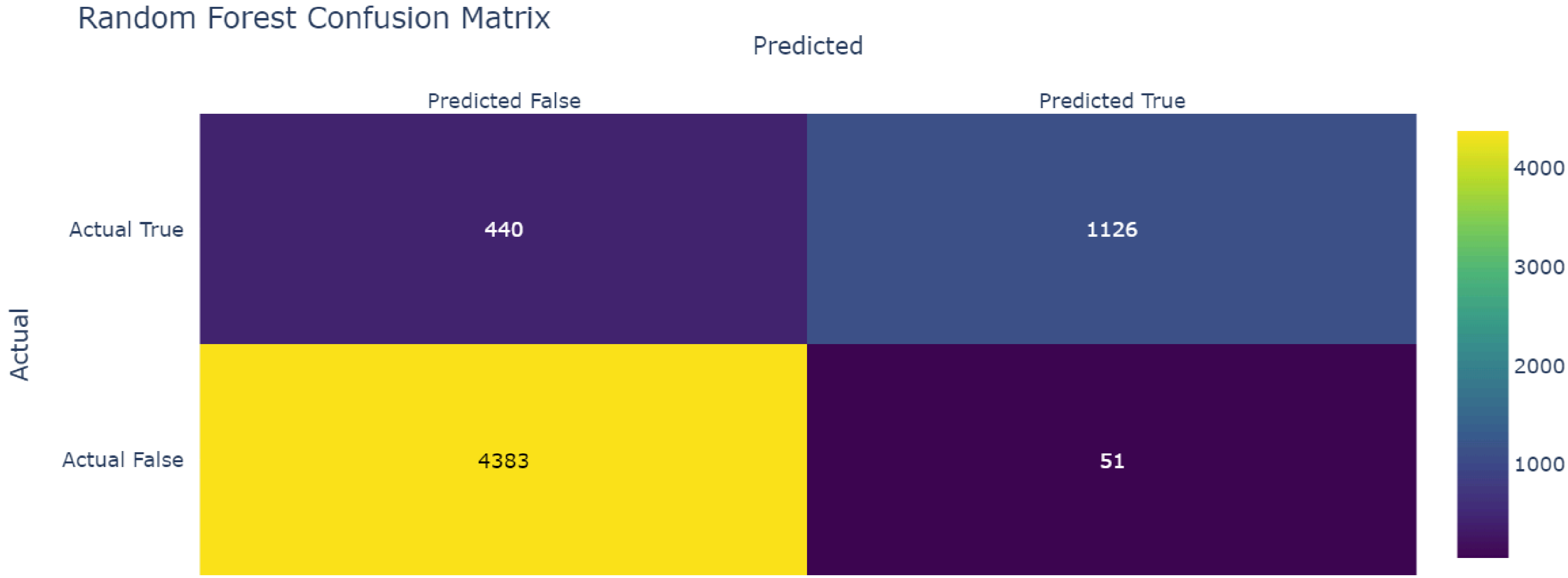

Logistic Regression ROC Curve
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC curve (area = 0.95)



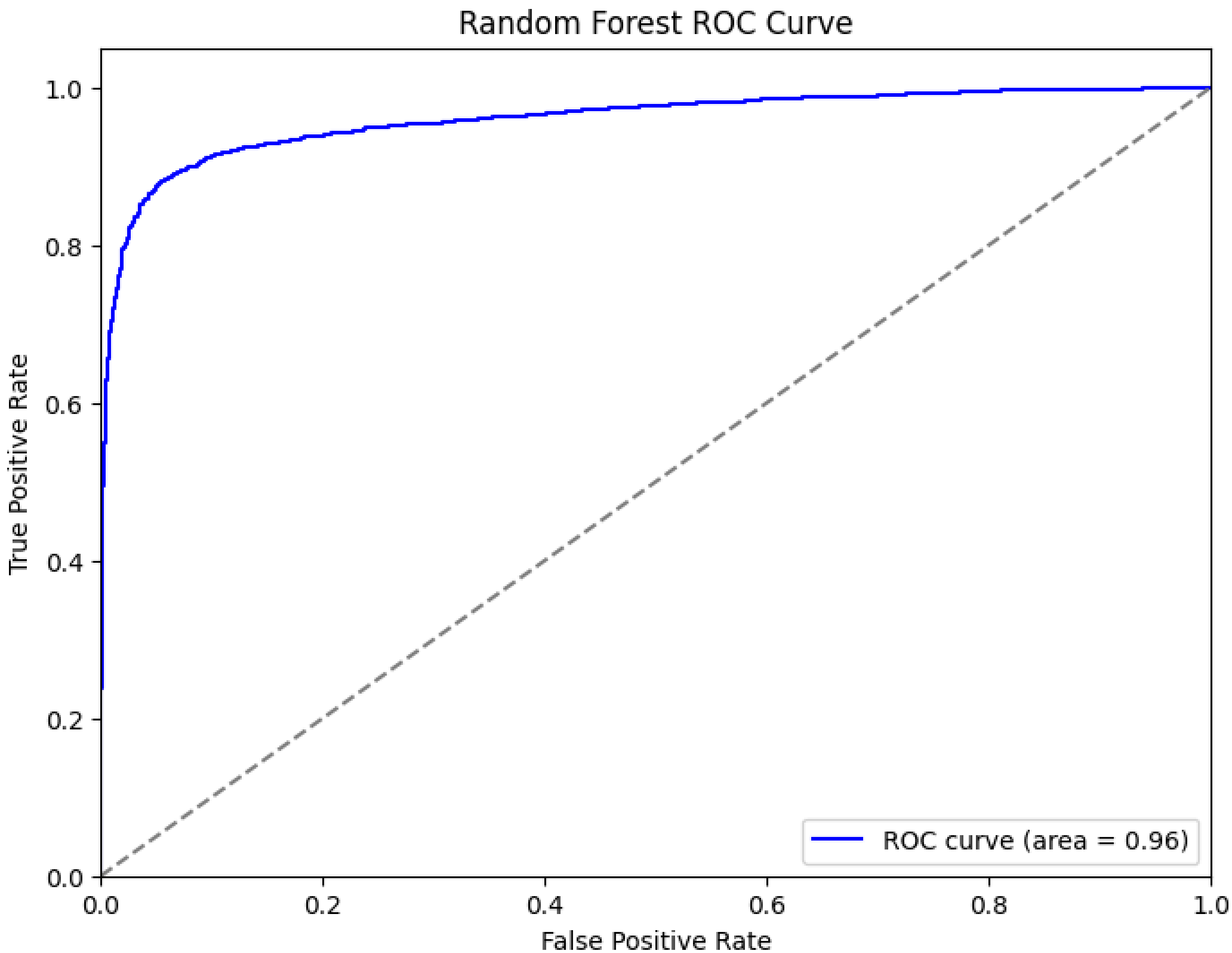
Random Forest

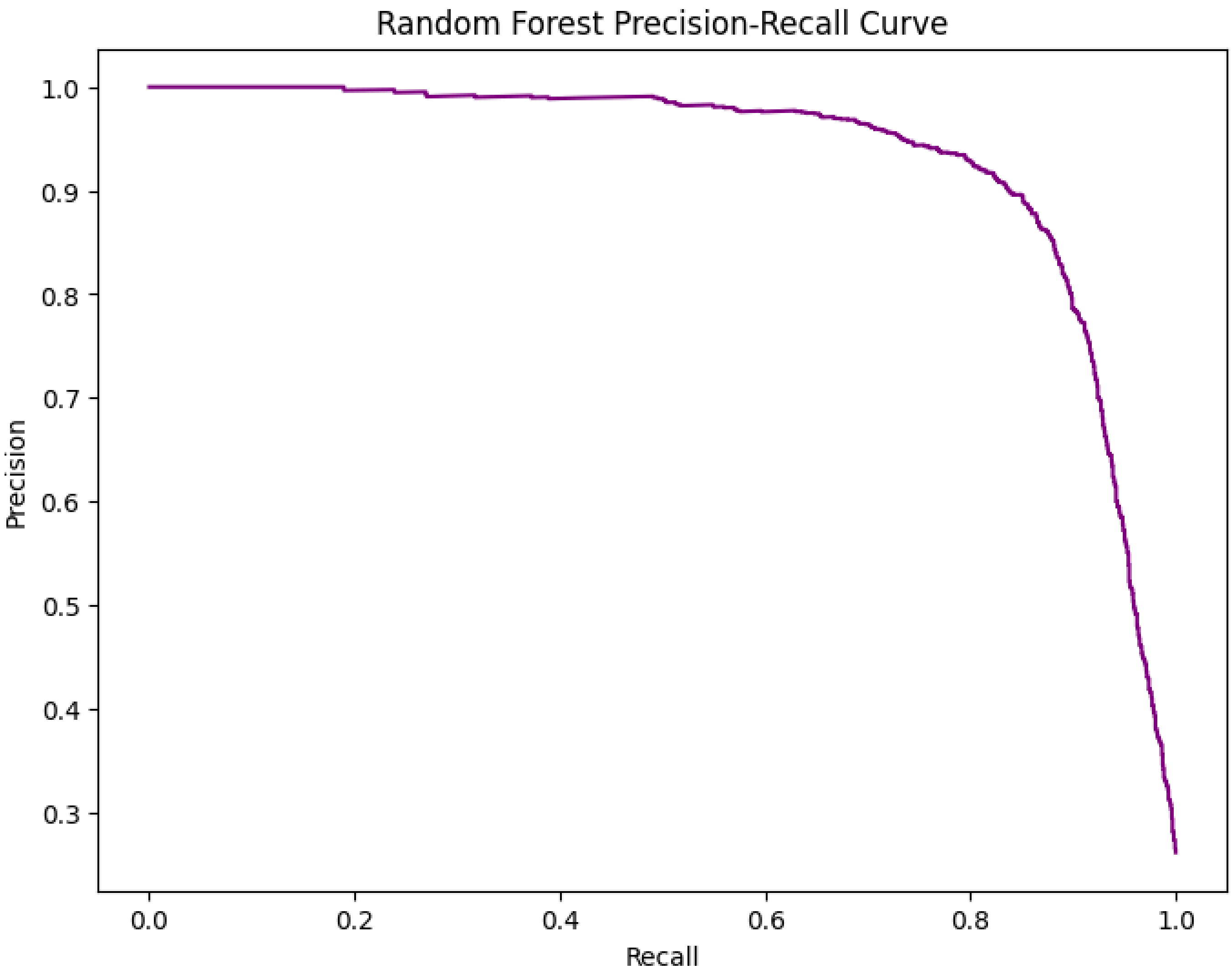
After training and fitting our model with the processed dataset, we got an overall accuracy of 91.8%.

The results are represented in the a confusion matrix below.



To further assess the robustness of the model, we plotted a ROC curve and precision recall curve.

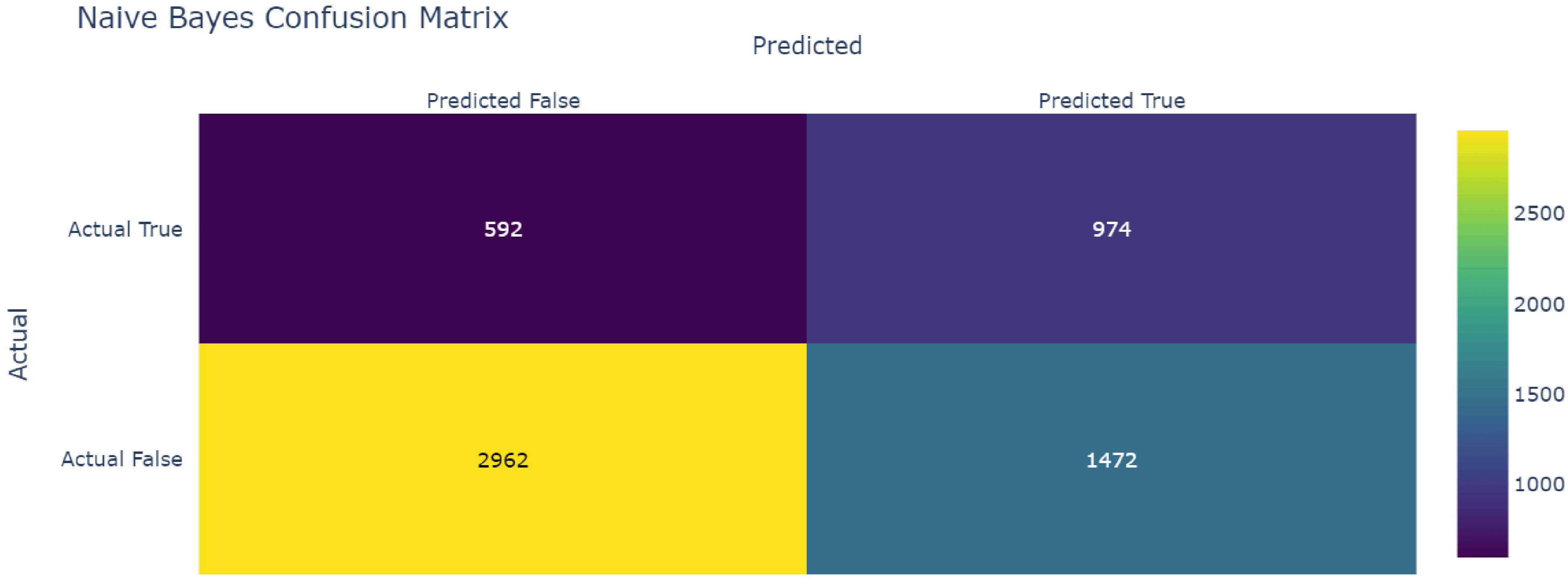

Random Forest ROC Curve
True Positive Rate
False Positive Rate
ROC curve (area = 0.96)
0.0
0.2
0.4
0.6
0.8
1.0



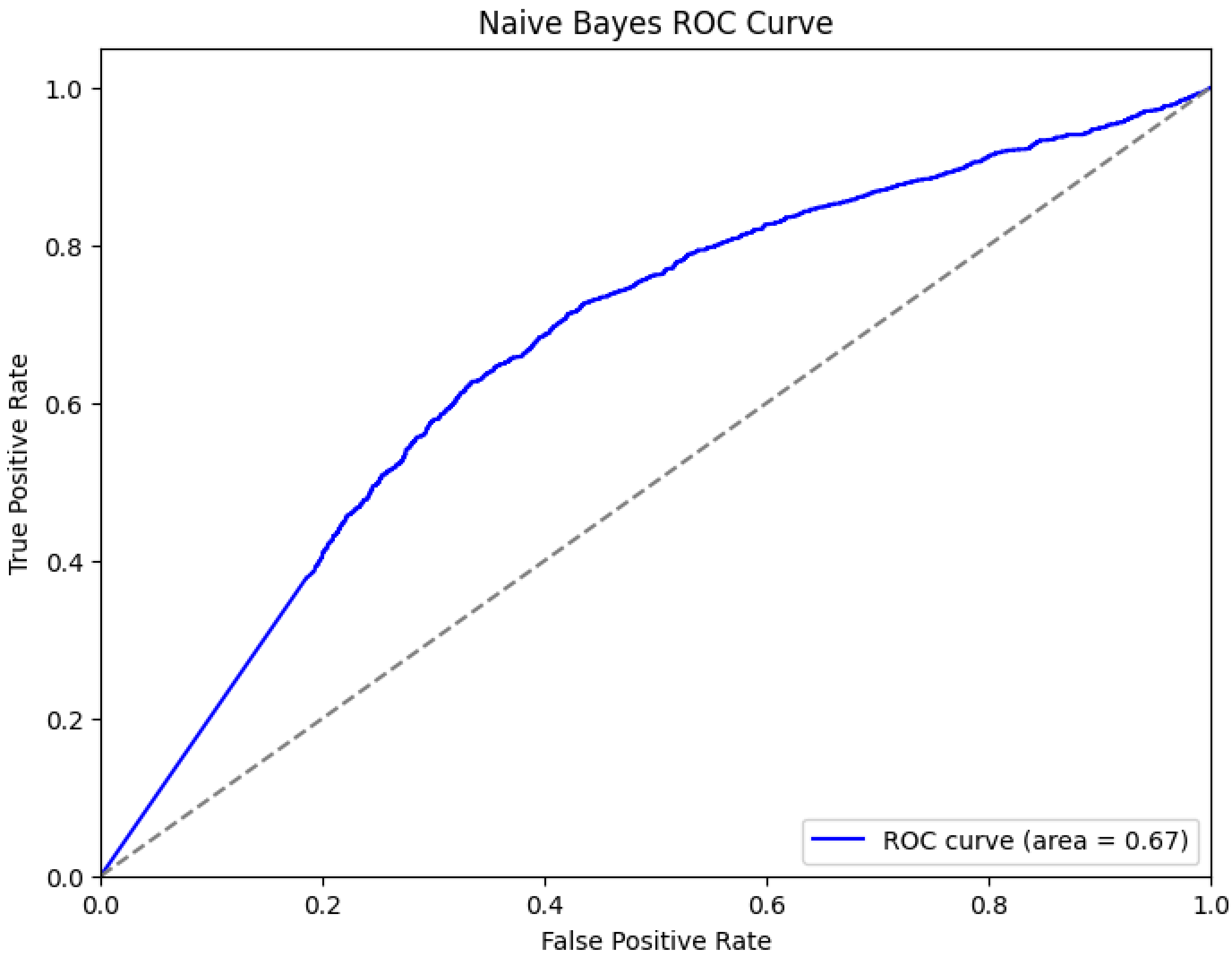
Naive Bayes Network

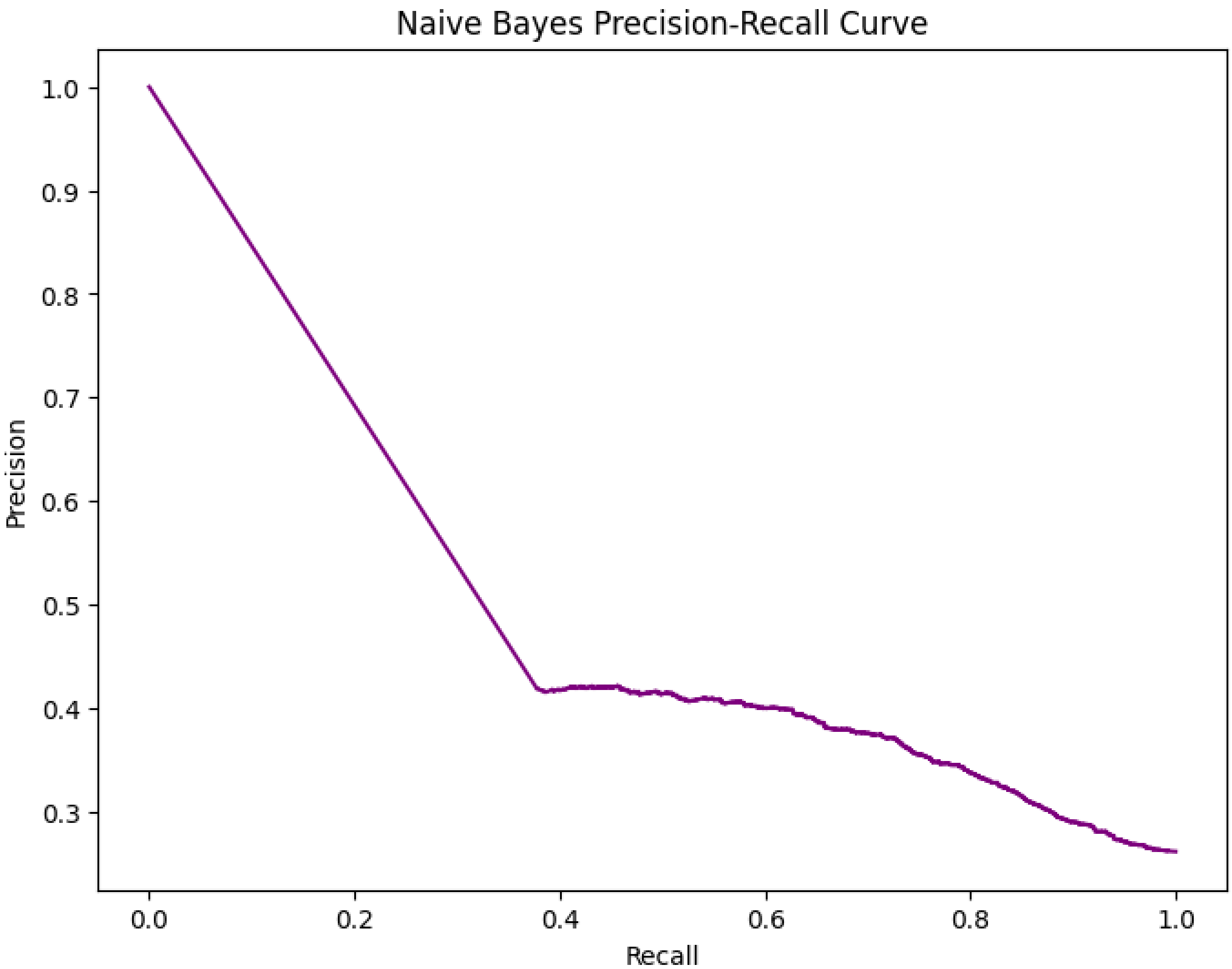
After training and fitting our model with the processed dataset, we got an overall accuracy of 65.6%.

The results are represented in the a confusion matrix below.



To further assess the robustness of the model, we plotted a ROC curve and precision recall curve.

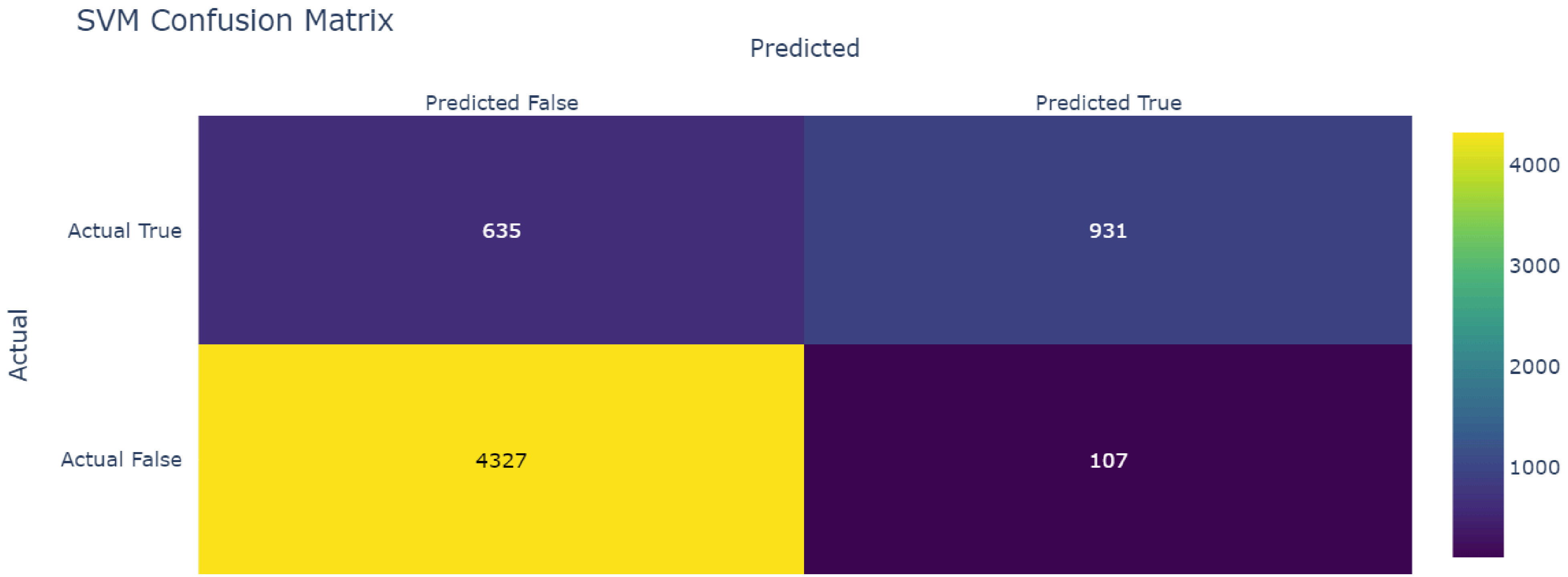

Naive Bayes ROC Curve
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC curve (area = 0.67)



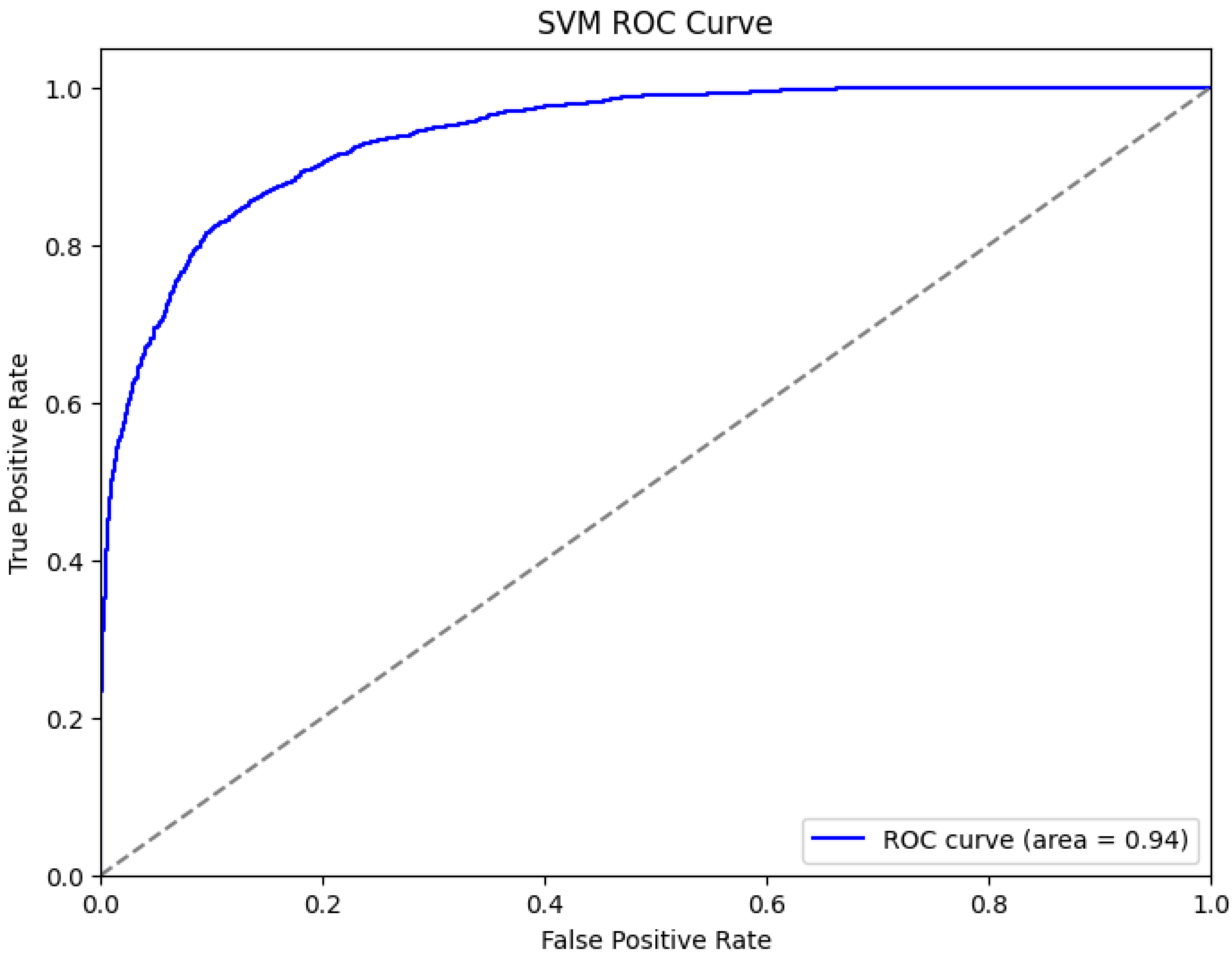
SVM

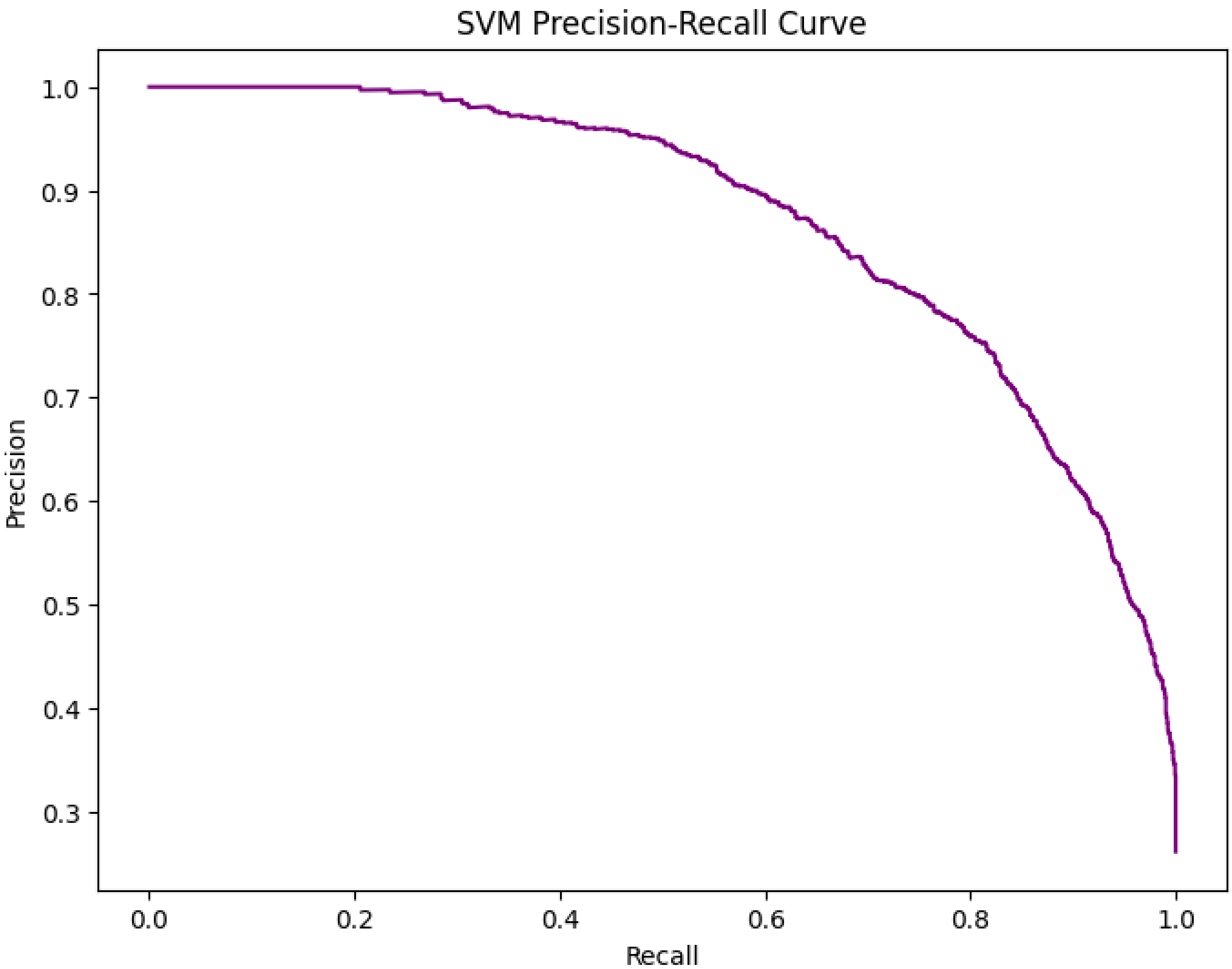
After training and fitting our model with the processed dataset, we got an overall accuracy of 87.6%.

The results are represented in the a confusion matrix below.



To further assess the robustness of the model, we plotted a ROC curve and precision recall curve.


SVM ROC Curve
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC curve (area = 0.94)



Discussion

We assesed the model with a confusion matrices as shown above. Since we use binary classification, a random guess has a 50% accuracy. Because of this, we wanted to create a model that performed well with 80-95% accuracy.

Logistic Regression

Logistic regression is a simple linear model that is known to avoid overfitting. This gives more stable and understandable predcitions whereas complex modles may not. This led to an accuracy value of 0.896. The ROC and precision-recall curves also further prove that the model is reliable.

Random Forest

Random forest performs wells on models with large dimensionality or data with missing values which was perfect for our large movie dataset with all the various columns and embeddings. This led to a accuracy value of 0.918 which was our best precision recorded. The ROC and precision-recall curves also further prove that the model is reliable.

Naive Bayes

Naive Bayes did not perform as well on our dataset as the other models. This may be because Naive Bayes assumes that all the features are independent which may have affected its accuracy. This led to a value of 0.656 which is very close to just having the classification be a random guess. The ROC and precision-recall curves also show that the model is not reliable.

SVM

Due to the failure of the Naive Bayes model we implemented a support vector machine (SVM) model. SVM is versatile and supports various kernel functions to adapt to complex data distributions. For this model we used the radial basis function (rbf) kernel as it is able to capture complex non-linear data which is needed for the movie dataset. With this, the accuracy value was 0.876. The ROC and precision-recall curves also enforce that the model is reliable.

Overall Conclusion

Through all this we have determined that the Random Forest classifier is the best model for our dataset. It handles the large dimensionality and complexity of our dataset. Logistic regression and SVM were both robust and had good accuracy; Naive bayes was by far the worst and did not handle the complexity and relationship between features in the dataset.

Next Steps

To further improve the data more filtering and scaling before feeding it into the models can be done. We also noticed some technical issues with the source of the data where some data was incorrect, this may have impacted our results. A UI can be created to feed in new data to the model to provide a binary classification of it that new movie will be rated well or not.

References

1. “Using Machine Learning to Predict Movie Reviews,” Medium, https://medium.com/@Coursesteach/using-machine-learning-to-predict-movie-reviews-82b0ab1db313

2. V. Onumaku, “IMDB Movie Ratings Prediction with Machine Learning.,” Medium, https://medium.com/@Onumaku_chibuike/imdb-movie-ratings-prediction-with-machine-learning-7bdaf843c268

3. D. Jurafsky, Language Modeling, https://web.stanford.edu/~jurafsky/slp3/slides/LM_4.pdf (accessed Oct. 3, 2024).

4. Z. Balfagih, “Decoding Cinematic Fortunes: A Machine Learning Approach to Predicting Film Success,” 2024 21st Learning and Technology Conference (L&T), Jeddah, Saudi Arabia, 2024, pp. 144-148, doi: 10.1109/LT60077.2024.10468906.

5. T. Sharma, R. Dichwalkar, S. Milkhe and K. Gawande, “Movie Buzz - Movie Success Prediction System Using Machine Learning Model,” 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS), Thoothukudi, India, 2020, pp. 111-118, doi: 10.1109/ICISS49785.2020.9316087.

Contribution Table

Name	Contribution
Priya Soneji	GitHub Pages, Data Analysis and Discussion, SVM Implementation
Joshua Mao	Logistic Regression Implementation, Video Editing
Eric Wen	Random Forest Implementation

Name	Contribution
Evan Douglass	Methods, Data Preprocessing
Matthew Kim	Intro, Naive Bayes Implementation

Gantt Chart

Gantt Chart Excel Sheet

Final Presentation Video

Video
