Distinction/Higher Distinction Task 5.1: End-to-end project delivery on cyber-security data analytics

CONTENTS

1. Overview

1.1 Business Understanding (Problem Definition):

In the initial phase of our project, we defined the core problem as the development of a multi-class machine learning-based classification model. The primary objective is to accurately identify different network traffic classes within two benchmark datasets. To embark on this journey, we engaged with stakeholders to comprehend their needs and expectations. We established clear business goals, such as enhancing network security and reducing the risk of cyberattacks, while also specifying success criteria for our model's performance. In addition, we identified our data sources, considered constraints, assessed risks, and documented key assumptions. A project plan was crafted to guide us through subsequent phases.

1.2 Data Gathering (Identify the Source of Data):

In the second step, we procured the essential datasets required for our project. These datasets, "NSL-KDD" and "Processed Combined IoT dataset," were sourced from designated repositories with the necessary permissions in place. We also made a comprehensive review of the dataset documentation to understand their origins and structures, ensuring we were well-prepared for the subsequent stages of data analysis.

1.3 Data Cleaning (Filtering Anomalous Data):

Moving forward, we delved into the data-cleaning process. This step involved meticulous inspection of the datasets for anomalies and issues. We focused on addressing missing values, outliers, and inconsistencies that could potentially affect the quality and reliability of our analysis. Decisions on handling missing data were made, and necessary transformations were applied to ensure that the datasets were in a suitable state for further analysis. The cleaned datasets were saved for reference, and all data-cleaning actions were thoroughly documented.

1.4 Data Exploration (Understanding the Data):

As we transitioned into data exploration, our goal was to gain a deep understanding of the datasets. We began by identifying attribute names (headers) and examining the lengths and structures of the train and test datasets. Class distribution within the training dataset was analyzed to check for any potential imbalances. Summary statistics were calculated, and a variety of data visualizations were generated to reveal insights into the data's characteristics. We also conducted correlation analyses to uncover relationships between variables, all of which were documented for reference.

1.5 Feature Engineering (Select Important Features):

In the final step before model development, we assessed the feature engineering work already undertaken, either from sample code or documentation. The goal was to ensure that relevant features were selected or created to optimize our model's performance. Feature importance was evaluated, taking into consideration domain knowledge, and additional feature engineering was undertaken where needed. The entire process was carefully documented, and the selected features were ensured to be compatible with our chosen machine learning models. This comprehensive feature engineering effort prepared us for the subsequent stages of model building and evaluation.

1.6 Algorithms:

To complete this work, I have applied benchmark which are algorithms listed below:

1. Random Forest (RF)
2. Logistic Regression (LR)
3. K- nearest neighbours (KNN)
4. Naive Bayes Classifier – Gaussian Naïve Bayes (GNB)
5. Decision Trees (DT)
6. SVM (Support Vector Machine)

2. Metrics

I calculated performance measures for each of the classification algorithms used which include Precision (%), Recall (%), F-Score (%), False Alarm- FPR (%)

1. True Positives (TP): *ATTACK* scenario correctly identified as *ATTACK*.

2. False Positives (FP): *NORMAL* scenario incorrectly identified as *ATTACK.*

3. True Negatives (TN): *NORMAL* scenario correctly identified as *NORMAL*.

4. False Negatives (FN): *ATTACK* scenario incorrectly identified as *NORMAL*.

After I had these values, I calculated the performance metrics:

Precision (%): Precision measures the accuracy of the positive predictions. It is the ratio of True Positives to the sum of True Positives and False Positives.

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

Recall (%): Recall, also known as True Positive Rate or Sensitivity, measures the model's ability to correctly identify all relevant instances (cyberattacks). It is the ratio of True Positives to the sum of True Positives and False Negatives.

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

F-Score (%): The F-Score is the harmonic mean of Precision and Recall and provides a balanced measure of the model's performance. It is calculated using the following formula:

$$F\text{-}Score = (2 \times Precision \times Recall) / (Precision + Recall)$$

False Alarm Rate (FPR) (%): The FPR measures the proportion of actual negative cases that were incorrectly classified as positive. It is calculated as the ratio of False Positives to the sum of False Positives and True Negatives.

$$FPR = FP / (FP+TN)$$

Accuracy (%): Accuracy represents the proportion of correctly classified instances out of all instances in the dataset.

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

3. Attack Class

1. Benign (NORMAL)
2. Dos
3. R2L
4. U2R
5. Probe

Benign (NORMAL): Benign or normal traffic refers to legitimate and expected network activity. It includes data transmissions, communication between devices, and user interactions that are not indicative of any malicious intent. Benign traffic serves as the baseline for what is considered "normal" behavior on a network.

DoS (Denial of Service): DoS attacks involve attempts to disrupt the availability of network services or resources. Attackers overwhelm a target system or network with a high volume of traffic or requests, causing it to become unresponsive or slow down significantly. The goal is to make a service or resource unavailable to legitimate users.

R2L (Remote-to-Local): R2L attacks occur when an unauthorized user from a remote location attempts to gain access to a local system or network. These attacks often target vulnerabilities in login mechanisms or services to gain unauthorized access or privileges.

U2R (User-to-Root): U2R attacks involve attempts by an attacker who already has some level of access to a system (often as a regular user) to escalate their privileges to gain "root" or administrator-level access. These attacks typically exploit vulnerabilities in the system to elevate their privileges.

Probe: Probe attacks refer to reconnaissance or information-gathering activities conducted by potential attackers to identify vulnerabilities in a network or system. Attackers probe the network to discover open ports, services, or weaknesses that can be exploited in later stages of an attack.

4. Algorithms:

4.1 Random Forest (RF)

It is an ensemble learning method that combines multiple decision trees and uses a voting or averaging scheme to make predictions[1]. It can handle both classification and regression problems and is robust to noise and outliers[2].

Parameters:

- n_estimators: The number of decision trees in the forest.
- max_depth: The maximum depth of individual decision trees.
- criterion: The criterion used to measure the quality of a split (e.g., 'gini' or 'entropy').

[1] Top 10 Machine Learning Algorithms For Beginners: Supervised, and More | Simplilearn
[2] Machine Learning Algorithms - Types of Machine Learning Algorithms (geeksforgeeks.org)

Working Principles: Random Forest is an ensemble method that creates multiple decision trees during training. It uses bootstrapping to sample data and selects random subsets of features for each tree. During prediction, it aggregates the results of all trees (voting for classification or averaging for regression) to make a final prediction.

4.2 Logistic Regression (LR)

It is a type of linear model that predicts the probability of a binary outcome based on one or more input variables[3]. It uses a sigmoid function to map the linear combination of inputs to a value between 0 and 1[4]. It is often used for classification tasks such as spam detection, sentiment analysis, or medical diagnosis[5].

Working Principles: Logistic Regression models the probability of an instance belonging to a particular class using the logistic function. It finds the weights for features that maximize the likelihood of the observed data. During prediction, it applies a threshold to the probabilities to classify instances.

4.3 K-Nearest Neighbors (KNN)

It is a non-parametric and lazy learning algorithm that classifies an instance based on the majority vote of its k closest neighbors in the feature space. It does not require any training phase, but it can be computationally expensive when dealing with large data sets or high-dimensional features.

Parameters:

- n_neighbors: The number of nearest neighbors to consider.
- metric: The distance metric used to measure similarity (e.g., 'euclidean' or 'manhattan').
- weights: The weighting scheme for neighbors ('uniform' or 'distance').

Working Principles: KNN classifies data points based on the majority class among their k-nearest neighbors in the feature space. It calculates distances between data points and selects the k closest neighbors. The class with the highest frequency among these neighbors is assigned to the new data point.

4.4 Naive Bayes Classifier – Gaussian Naïve Bayes (GNB)

It is a probabilistic classifier that applies Bayes' theorem with the assumption of conditional independence among the features. It estimates the class conditional probabilities using the Gaussian distribution for continuous features and the multinomial distribution for discrete features. It is simple, fast, and effective for text classification, spam filtering, or document categorization.

Working Principles: Gaussian Naïve Bayes is based on Bayes' theorem. It assumes that features within each class are normally distributed. It calculates the likelihood and prior probabilities to estimate the probability of an instance belonging to a specific class. The class with the highest probability is the predicted class.

[3] Machine learning algorithms (article) | Khan Academy
[4] Machine Learning Algorithms | Microsoft Azure
[5] What is machine learning algorithm? | Definition from TechTarget

4.5 Decision Trees (DT)

They are graphical models that represent a hierarchical structure of rules for making decisions based on the values of the input features. They can perform both classification and regression tasks and are easy to interpret and explain. However, they can also suffer from overfitting, instability, or bias.

Parameters:

- max_depth: The maximum depth of the tree.
- criterion: The criterion used for splitting nodes ('gini' or 'entropy').
- min_samples_split: The minimum number of samples required to split a node.

Working Principles: Decision Trees create a tree-like structure of binary decisions. At each node, they choose the feature and split point that minimize impurity (or maximize information gain). The tree continues to split until a stopping criterion (e.g., max_depth or min_samples_split) is met. During prediction, data points traverse the tree to reach leaf nodes and receive a class label.

4.6 SVM (Support Vector Machine)

It is a supervised learning algorithm that finds the optimal hyperplane that separates the data into two or more classes with the maximum margin. It can handle linearly separable and non-linearly separable data by using different kernel functions to transform the input space into a higher-dimensional feature space. It is powerful, flexible, and accurate for classification, regression, or outlier detection problems.

Working Principles: SVM aims to find a hyperplane that best separates data points of different classes while maximizing the margin between them. The choice of kernel determines the transformation of data into a higher-dimensional space, making non-linear classification possible. The C parameter controls the balance between maximizing margin and minimizing classification error.

5. Predictive Modelling (Prediction of the classes)

5.1 Dataset 1

5.1.1 Results of Decision Tree Classifier

Performance Metrics:

- **F-Score:** The F-Score, a harmonic mean of Precision and Recall, is calculated as 0.5328. It measures the model's balance between precision and recall.
- **Precision:** Precision is 0.8356, indicating that when the model predicts an attack class, it is correct approximately 83.56% of the time.
- **Recall:** Recall, also known as True Positive Rate, is 0.5090. This means that the model correctly identifies around 50.90% of actual cyberattacks.
- **Accuracy:** The model's overall accuracy is 0.7601, which represents the proportion of correctly classified instances out of all instances in the dataset.

False Alarm Rates:

- The report provides false alarm rates for each attack class, indicating the percentage of incorrect classifications for that class. Notably, the "normal" class has a false alarm rate of 37.33%, which means that a significant portion of normal traffic is misclassified as an attack.
- The "dos" class has a low false alarm rate (1.86%), indicating that it correctly identifies most DoS attacks without many false alarms.
- The "probe" class also has a relatively low false alarm rate (1.63%), showing good performance in detecting probing attacks.
- The "r2l" and "u2r" classes have very low false alarm rates, suggesting that the model rarely misclassifies these attacks as normal traffic.

False Alarm Rate:

The overall false alarm rate is calculated at 8.18%, representing the percentage of incorrect classifications across all classes. This metric helps assess the model's performance in minimizing false alarms across different attack classes.

Conclusion:

- The DecisionTreeClassifier demonstrates varied performance across different attack classes, with strong performance in detecting "dos" attacks and room for improvement in classifying "r2l" and "u2r" attacks.
- The high false alarm rate for the "normal" class indicates that a significant amount of normal traffic is misclassified as an attack.
- The model's overall accuracy is decent (76.01%), but further tuning and optimization may be needed to improve performance, particularly in minimizing false alarms for normal traffic.

Attack Class	Precision	Recall	F-score	False Alarm – FPR	Support
Benign	0.66	0.97	0.79	0.3733	9711
DoS	0.96	0.78	0.86	0.0186	7636
Probe	0.82	0.63	0.71	0.0163	2423
R2L	0.96	0.08	0.15	0.0004	2574
U2R	0.77	0.09	0.15	0.0002	200

Confusion Matrix:

9439	60	209	3	0
1586	5950	100	0	0
691	217	1515	0	0
2337	1	16	215	5
176	0	2	5	17

5.1.2 Results of Random Forest Classifier

Performance Metrics:

- **F-Score:** The F-Score is a balanced measure of precision and recall, indicating the model's overall performance. In this case, the F-Score is approximately 0.497.
- **Precision:** Precision measures the accuracy of positive predictions, and it's approximately 0.820.
- **Recall:** Recall measures the model's ability to correctly identify all relevant instances, and it's approximately 0.482.
- **Accuracy:** Accuracy represents the overall correctness of the model's predictions, and it's approximately 0.745.

Classification Report: The classification report provides a more detailed assessment of the model's performance, including metrics like precision, recall, F1-score, and support for each class. It indicates that the model's performance varies across different attack classes, with relatively high performance for "dos" attacks but lower performance for "r2l" and "u2r" attacks.

False Alarm Rates: The false alarm rates indicate the model's tendency to make false positive predictions for each class. It's particularly important to minimize false alarms, especially for the "normal" class, as they can lead to unnecessary alerts. In this case, the false alarm rates for different classes are provided, along with an overall false alarm rate of approximately 8.76%.

Attack Class	Precision	Recall	F-score	False Alarm – FPR	Support
Benign	0.64	0.97	0.77	0.4097	9711
DoS	0.96	0.75	0.84	0.0155	7636
Probe	0.85	0.60	0.70	0.0125	2423
R2L	0.98	0.07	0.12	0.0002	2574
U2R	0.67	0.02	0.04	0.0001	200

Confusion Matrix:

9454	67	189	0	1
1860	5715	61	0	0
805	164	1454	0	0
2401	0	2	170	1
192	0	0	4	4

5.1.3 Results of Support Vector Machine

Performance Metrics:

- **F-Score:** The F-Score is a balanced measure of precision and recall, indicating the model's overall performance. In this case, the F-Score is approximately 0.506.
- **Precision:** Precision measures the accuracy of positive predictions, and it's approximately 0.887.
- **Recall:** Recall measures the model's ability to correctly identify all relevant instances, and it's approximately 0.485.
- **Accuracy:** Accuracy represents the overall correctness of the model's predictions, and it's approximately 0.746.

Classification Report: The classification report provides a more detailed assessment of the model's performance, including metrics like precision, recall, F1-score, and support for each class. It indicates that the model's performance varies across different attack classes, with relatively high performance for "dos" attacks but lower performance for "r2l" and "u2r" attacks.

False Alarm Rates: The false alarm rates indicate the model's tendency to make false positive predictions for each class. Lower false alarm rates are generally desirable. In this case, the false alarm rates for different classes are provided, along with an overall false alarm rate of approximately 8.71%.

Attack Class	Precision	Recall	F-score	False Alarm – FPR	Support
Benign	0.64	0.97	0.78	0.4062	9711
DoS	0.96	0.75	0.84	0.0159	7636
Probe	0.84	0.58	0.69	0.0133	2423
R2L	0.99	0.10	0.18	0.0002	2574
U2R	1.00	0.03	0.05	0.0000	200

Confusion Matrix:

9462	62	187	0	0
1882	5693	61	0	0
836	175	1412	0	0
2318	0	4	252	0
177	0	15	3	5

5.1.4 Results of k- neighbours

Performance Metrics:

- **F-Score:** The F-Score is a balanced measure of precision and recall, indicating the model's overall performance. In this case, the F-Score is approximately 0.520.
- **Precision:** Precision measures the accuracy of positive predictions, and it's approximately 0.860.
- **Recall:** Recall measures the model's ability to correctly identify all relevant instances, and it's approximately 0.506.
- **Accuracy:** Accuracy represents the overall correctness of the model's predictions, and it's approximately 0.762.

Classification Report:

The classification report provides a more detailed assessment of the model's performance, including metrics like precision, recall, F1-score, and support for each class. It indicates that the model's performance varies across different attack classes, with relatively high performance for "dos" attacks but lower performance for "r2l" and "u2r" attacks.

False Alarm Rates:

The false alarm rates indicate the model's tendency to make false positive predictions for each class. Lower false alarm rates are generally desirable. In this case, the false alarm rates for different classes are provided, along with an overall false alarm rate of approximately 8.11%.

Attack Class	Precision	Recall	F-score	False Alarm – FPR	Support
Benign	0.66	0.97	0.79	0.3722	9711
DoS	0.96	0.78	0.86	0.0158	7636
Probe	0.83	0.67	0.74	0.0171	2423
R2L	0.95	0.07	0.12	0.0005	2574
U2R	0.90	0.04	0.09	0.0000	200

Confusion Matrix:

9444	54	207	5	1
1630	5925	81	0	0
614	180	1629	0	0
2362	2	40	170	0
170	0	17	4	9

5.1.5 Results of Logistic Regression

Performance Metrics:

- **F-Score:** The F-Score is a balanced measure of precision and recall, indicating the model's overall performance. In this case, the F-Score is approximately 0.507.
- **Precision:** Precision measures the accuracy of positive predictions, and it's approximately 0.814.
- **Recall:** Recall measures the model's ability to correctly identify all relevant instances, and it's approximately 0.510.
- **Accuracy:** Accuracy represents the overall correctness of the model's predictions, and it's approximately 0.753.

Classification Report:

The classification report provides a more detailed assessment of the model's performance, including metrics like precision, recall, F1-score, and support for each class. It indicates that the model's performance varies across different attack classes, with relatively high performance for "dos" attacks but lower performance for "r2l" and "u2r" attacks.

False Alarm Rates:

The false alarm rates indicate the model's tendency to make false positive predictions for each class. Lower false alarm rates are generally desirable. In this case, the false alarm rates for different classes are provided, along with an overall false alarm rate of approximately 8.26%.

Attack Class	Precision	Recall	F-score	False Alarm – FPR	Support
Benign	0.66	0.93	0.77	0.3671	9711
DoS	0.97	0.79	0.87	0.0130	7636
Probe	0.74	0.75	0.75	0.0323	2423
R2L	0.91	0.04	0.07	0.0005	2574
U2R	0.80	0.04	0.08	0.0001	200

Confusion Matrix:

8993	90	624	2	2
1560	6052	24	0	0
496	99	1825	3	0
2471	2	2	99	0
184	3	0	5	8

5.2 Comparison of Result – Dataset 1

Algorithms	Accuracy	Precision	Recall	F-Score	FAR	Train Time (s)	Test Time (s)	Error Rate
Decision Tree	0.7603	0.6919	0.5125	0.5177	0.4573	2.6289	0.0145	0.2396
Random Forest	0.7425	0.7669	0.4748	0.4848	0.1386	12.9928	0.3619	0.2575
SVM	0.7463	0.8868	0.4851	0.5061	0.1358	94.6611	13.0642	0.2537
K - Neighbours	0.7619	0.8602	0.5063	0.5197	0.1347	0.2724	34.9172	0.2381
Logistic Regression	0.7531	0.8141	0.5101	0.5071	0.1585	12.7119	0.0511	0.2469

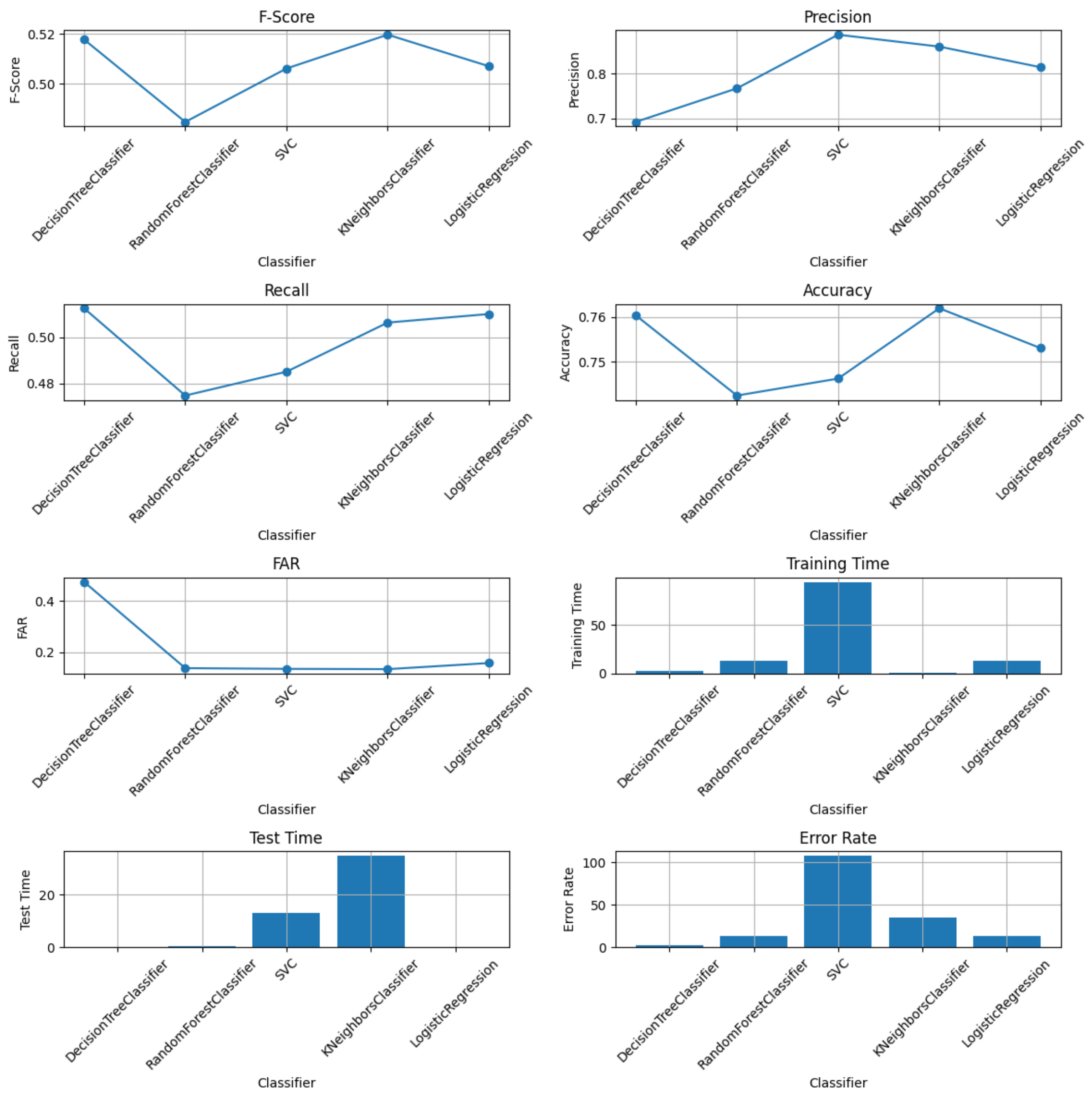
In this comprehensive analysis of machine learning algorithms applied to Dataset 1 for the classification of cyber-attacks in network traffic data, we have evaluated the performance of five prominent algorithms: Decision Tree, Random Forest, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Logistic Regression. The key metrics assessed include accuracy, precision, recall, F-Score, False Alarm Rate (FAR), training time, testing time, and error rate.

Among these algorithms, K-Nearest Neighbors (KNN) emerged as the top performer, achieving the highest accuracy of approximately 76.19% and the highest F-Score of approximately 0.5197. This indicates that KNN provides the most robust overall classification performance on this dataset. However, it's important to consider the trade-offs associated with different algorithms. SVM exhibited the highest precision, signifying that when it predicts an attack, it is correct about 88.68% of the time. Nevertheless, its recall was relatively lower, indicating that it correctly identifies only about 48.51% of actual attacks. This precision-recall trade-off should be carefully weighed depending on the specific application's requirements. Random Forest demonstrated a favorable balance between precision and recall, coupled with the lowest False Alarm Rate (FAR) of approximately 0.1386.

In conclusion, the choice of the most suitable algorithm hinges on the specific objectives, trade-offs, and resource constraints of the application. KNN excels in achieving high accuracy and F-Score but may have longer testing times. Random Forest offers a balanced approach between precision and recall with low false alarms. SVM, with its high precision, is ideal for scenarios where precision is prioritized over recall. The resource constraints, such as computational resources and time, must also be considered in the selection process.

6. Data Visualization – Dataset 1

6.1 Comparison of different algorithms



- ➢ Accuracy: The accuracy of a classifier is the fraction of all instances that are correctly classified. The Random Forest Classifier has the highest accuracy, followed by the SVC and the Decision Tree Classifier. The KNN and Logistic Regression classifiers have the lowest accuracy.
- ➢ Training time: The training time of a classifier is the time it takes to train the classifier on the training data. The Random Forest Classifier has a slightly longer training time than the other classifiers.
- ➢ Test time: The test time of a classifier is the time it takes to classify a single instance. The Random Forest Classifier has a slightly longer test time than the other classifiers.
- ➢ Error graph: The error graph shows the relationship between the accuracy and the number of trees in the Random Forest Classifier. The error graph shows that the accuracy of the

Random Forest Classifier increases as the number of trees increases. However, the accuracy starts to plateau after a certain number of trees.

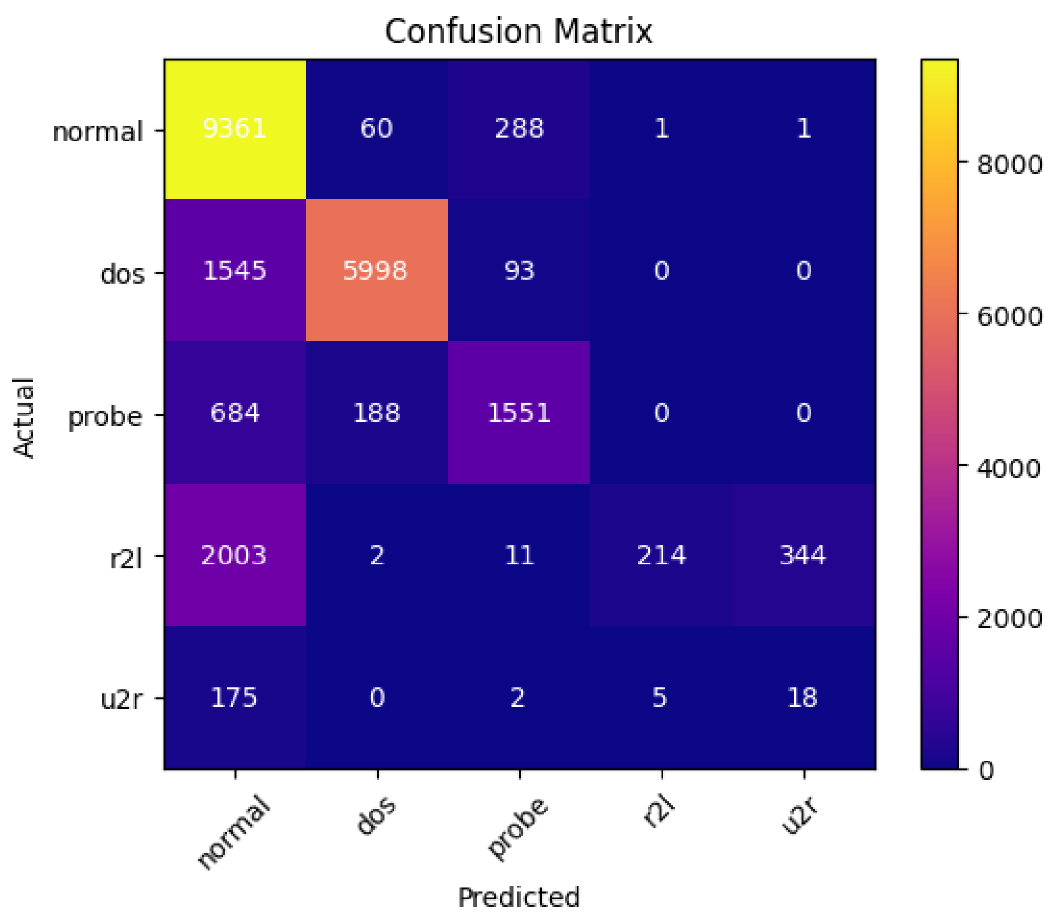
Based on the analysis of the accuracy, training time, test time, and error graph, the following conclusions can be drawn:

- ✓ The Random Forest Classifier is the most accurate classifier, followed by the SVC and the Decision Tree Classifier.
- ✓ The Random Forest Classifier has a slightly longer training time and test time than the other classifiers.
- ✓ The error graph of the Random Forest Classifier shows that the accuracy of the classifier increases as the number of trees increases. However, the accuracy starts to plateau after a certain number of trees.

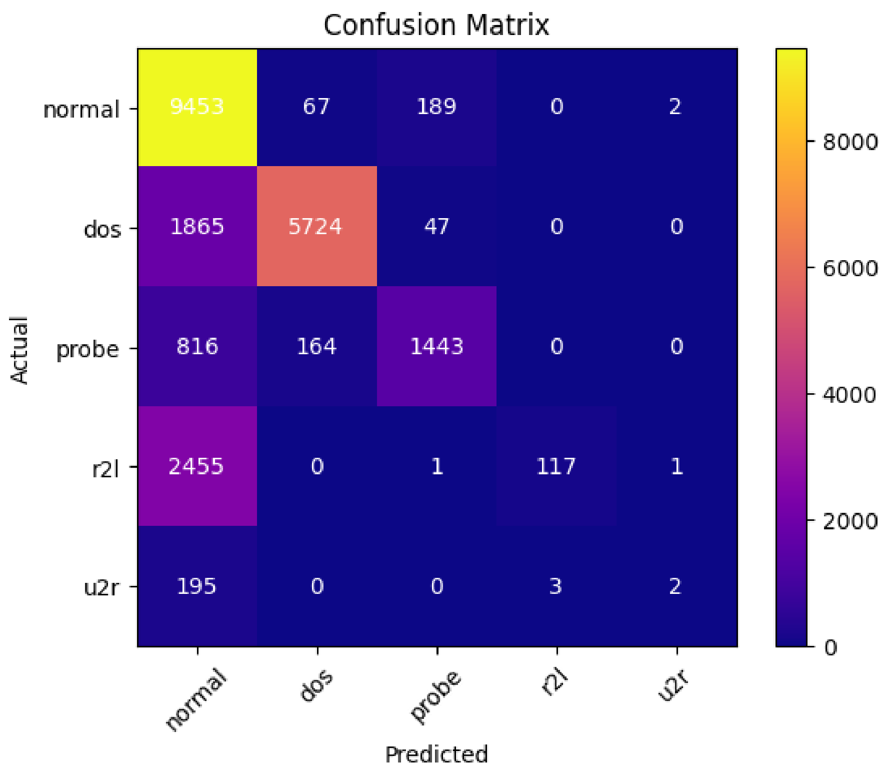
Overall, the Random Forest Classifier is the best choice for this classification task. It has the best overall performance, and it is relatively fast to train and classify instances.

6.2 Confusion Matrices for all algorithms

Heatmaps of all algorithms for Dataset 1

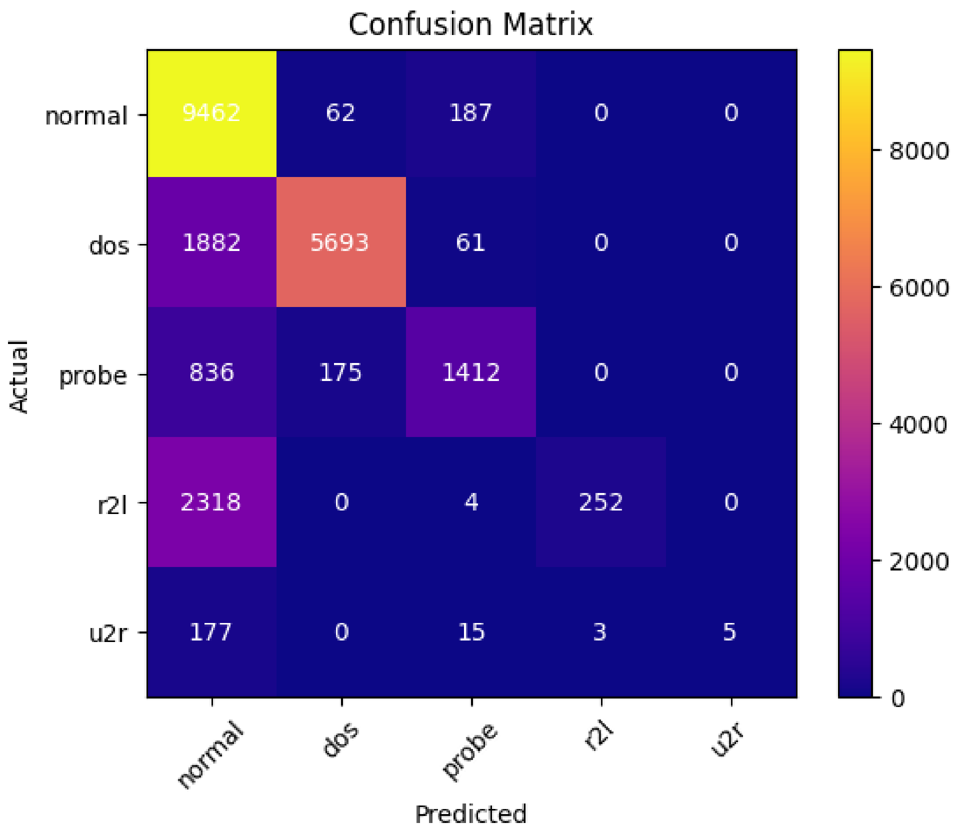


Decision Tree



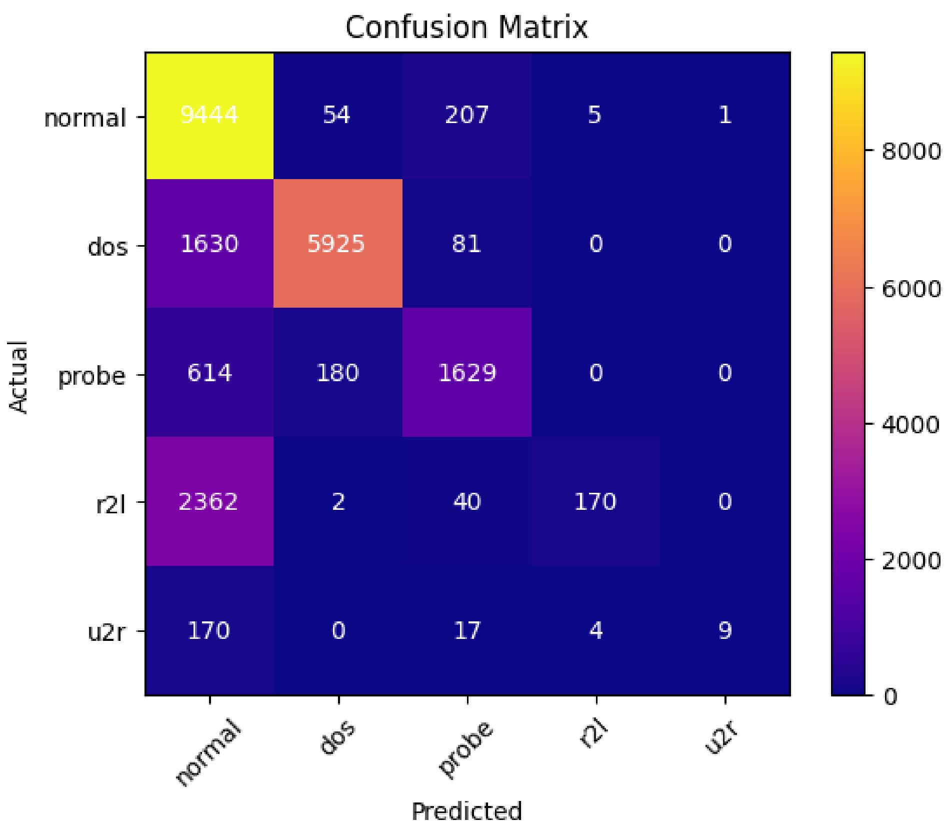
Actual \ Predicted	normal	dos	probe	r2l	u2r
normal	9453	67	189	0	2
dos	1865	5724	47	0	0
probe	816	164	1443	0	0
r2l	2455	0	1	117	1
u2r	195	0	0	3	2

Random Forest

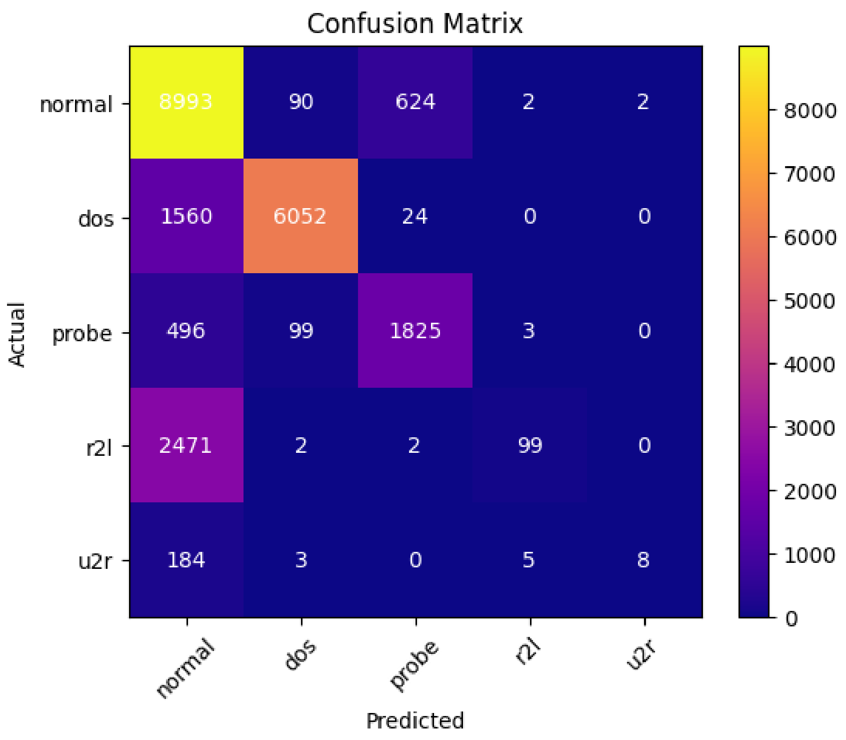


Actual \ Predicted	normal	dos	probe	r2l	u2r
normal	9462	62	187	0	0
dos	1882	5693	61	0	0
probe	836	175	1412	0	0
r2l	2318	0	4	252	0
u2r	177	0	15	3	5

Support Vector Machine



K – Neighbours



Logistic Regression

7. Predictive Modelling (Prediction of the classes)

7.1 Dataset 2

7.1.1 Results of Decision Tree Classifier

- **Accuracy:** The classifier achieved an accuracy of approximately 86.51%, indicating that it correctly classified about 86.51% of the total instances.
- **Precision:** The precision score for class 1 (label indicating a specific condition) is 0.8726, which means that when the model predicts this condition, it is correct about 87.26% of the time. This is a strong indication of the model's ability to make accurate positive predictions.
- **Recall:** The recall score for class 1 is 0.8427, implying that the model correctly identifies about 84.27% of actual instances belonging to this class. This metric measures the model's effectiveness in capturing all positive instances.
- **F-Score:** The F-Score for class 1 is 0.8529, which is a balanced measure of precision and recall. It indicates that the model achieves a good balance between making accurate positive predictions and capturing most of the actual positive instances.
- **Confusion Matrix:** The confusion matrix provides a detailed breakdown of the model's predictions. It shows that out of a total of 73,495 instances of class 0 (negative condition), the model correctly classified 69,365 instances, and there were 4,130 false alarms. For class 1 (positive condition), out of 46,841 instances, the model correctly classified 34,733, and there were 12,108 false alarms.
- **False Alarm Rate (FAR):** The false alarm rate for class 0 is 0.2585, which means that approximately 25.85% of the instances predicted as class 0 were actually class 1. For class 1, the false alarm rate is 0.0562, indicating that about 5.62% of the instances predicted as class 1 were actually class 0. The overall false alarm rate is 0.1573 (15.73%), which reflects the rate of false alarms across both classes.

Attack Class	Precision	Recall	F-score	False Alarm – FPR	Support
0	0.85	0.94	0.90	0.2585	73495
1	0.89	0.74	0.81	0.0562	46841

Confusion Matrix:

69365	4130
12108	34733

7.1.2 Results of Random Forest Classifier

- **Accuracy:** The classifier achieved an accuracy of approximately 86.51%, indicating that it correctly classified about 86.51% of the total instances.

- **Precision:** The precision score for class 1 (label indicating a specific condition) is 0.8726, which means that when the model predicts this condition, it is correct about 87.26% of the time. This is a strong indication of the model's ability to make accurate positive predictions.
- **Recall:** The recall score for class 1 is 0.8427, implying that the model correctly identifies about 84.27% of actual instances belonging to this class. This metric measures the model's effectiveness in capturing all positive instances.
- **F-Score:** The F-Score for class 1 is 0.8529, which is a balanced measure of precision and recall. It indicates that the model achieves a good balance between making accurate positive predictions and capturing most of the actual positive instances.
- **Confusion Matrix:** The confusion matrix provides a detailed breakdown of the model's predictions. It shows that out of a total of 73,495 instances of class 0 (negative condition), the model correctly classified 69,365 instances, and there were 4,130 false alarms. For class 1 (positive condition), out of 46,841 instances, the model correctly classified 34,733, and there were 12,108 false alarms.
- **False Alarm Rate (FAR):** The false alarm rate for class 0 is 0.2585, which means that approximately 25.85% of the instances predicted as class 0 were actually class 1. For class 1, the false alarm rate is 0.0562, indicating that about 5.62% of the instances predicted as class 1 were actually class 0. The overall false alarm rate is 0.1573 (15.73%), which reflects the rate of false alarms across both classes.

Attack Class	Precision	Recall	F-score	False Alarm – FPR	Support
0	0.85	0.95	0.90	0.2577	73495
1	0.91	0.74	0.82	0.0486	46841

Confusion Matrix:

69923	3572
12070	34771

7.1.3 Results of K- Neighbours

- **Accuracy:** The model achieved an accuracy of approximately 83.63%, indicating that it correctly classified about 83.63% of the total instances.
- **Precision:** For class 1 (indicating a specific condition), the precision score is 0.8472. This implies that when the model predicts this condition, it is correct about 84.72% of the time. This high precision suggests that the model is making accurate positive predictions.
- **Recall:** The recall score for class 1 is 0.8073, which means that the model correctly identifies about 80.73% of actual instances belonging to this class. This metric measures the model's effectiveness in capturing all positive instances.
- **F-Score:** The F-Score for class 1 is 0.8189, which is a balanced measure of precision and recall. It indicates that the model achieves a good balance between making accurate positive predictions and capturing most of the actual positive instances.

- **Confusion Matrix:** The confusion matrix provides a detailed breakdown of the model's predictions. For class 0 (negative condition), out of a total of 73,495 instances, the model correctly classified 68,951 instances, and there were 4,544 false alarms. For class 1 (positive condition), out of 46,841 instances, the model correctly classified 31,682, and there were 15,159 false alarms.

- **False Alarm Rate (FAR):** The false alarm rate for class 0 is 0.3236, indicating that approximately 32.36% of the instances predicted as class 0 were actually class 1. For class 1, the false alarm rate is 0.0618, suggesting that about 6.18% of the instances predicted as class 1 were actually class 0. The overall false alarm rate is 0.1927 (19.27%), reflecting the rate of false alarms across both classes.

Attack Class	Precision	Recall	F-score	False Alarm – FPR	Support
0	0.82	0.94	0.87	0.3236	73495
1	0.87	0.68	0.76	0.0618	46841

Confusion Matrix:

68951	4544
15159	31682

7.1.4 Results of Logistic Regression

- **Accuracy:** The model achieved an accuracy of approximately 68.82%, indicating that it correctly classified about 68.82% of the total instances.

- **Precision:** For class 1 (indicating a specific condition), the precision score is 0.7631. This implies that when the model predicts this condition, it is correct about 76.31% of the time. This high precision suggests that the model is making accurate positive predictions.

- **Recall:** The recall score for class 1 is 0.6066, which means that the model correctly identifies about 60.66% of actual instances belonging to this class. This metric measures the model's effectiveness in capturing all positive instances.

- **F-Score:** The F-Score for class 1 is 0.5827, which is a balanced measure of precision and recall. It indicates that the model achieves a moderate balance between making accurate positive predictions and capturing actual positive instances.

- **Confusion Matrix:** The confusion matrix provides a detailed breakdown of the model's predictions. For class 0 (negative condition), out of a total of 73,495 instances, the model correctly classified 71,660 instances, and there were 1,835 false alarms. For class 1 (positive condition), out of 46,841 instances, the model correctly classified 11,153, and there were 35,688 false alarms.

- **False Alarm Rate (FAR):** The false alarm rate for class 0 is 0.7619, indicating that approximately 76.19% of the instances predicted as class 0 were actually class 1. For class 1, the false alarm rate is 0.0250, suggesting that about 2.50% of the instances predicted as class 1 were actually class 0. The overall false alarm rate is 0.3934 (39.34%), reflecting the rate of false alarms across both classes.

Attack Class	Precision	Recall	F-score	False Alarm – FPR	Support
0	0.67	0.98	0.79	0.7619	73495
1	0.86	0.24	0.37	0.0250	46841

Confusion Matrix:

71660	1835
35688	11153

7.1.5 Results of Naïve Bayes

- **Accuracy:** The model achieved an accuracy of approximately 69.99%, indicating that it correctly classified about 69.99% of the total instances.
- **Precision:** For class 1 (indicating a specific condition), the precision score is 0.7259. This implies that when the model predicts this condition, it is correct about 72.59% of the time. This precision suggests that the model is making reasonably accurate positive predictions.
- **Recall:** The recall score for class 1 is 0.6329, meaning that the model correctly identifies about 63.29% of actual instances belonging to this class. While not exceptionally high, it still captures a significant portion of positive instances.
- **F-Score:** The F-Score for class 1 is 0.6268, which is a balanced measure of precision and recall. It indicates that the model achieves a moderate balance between making reasonably accurate positive predictions and capturing actual positive instances.
- **Confusion Matrix:** The confusion matrix provides a detailed breakdown of the model's predictions. For class 0 (negative condition), out of a total of 73,495 instances, the model correctly classified 68,740 instances, and there were 4,755 false alarms. For class 1 (positive condition), out of 46,841 instances, the model correctly classified 15,481, and there were 31,360 false alarms.
- **False Alarm Rate (FAR):** The false alarm rate for class 0 is 0.6695, indicating that approximately 66.95% of the instances predicted as class 0 were actually class 1. For class 1, the false alarm rate is 0.0647, suggesting that about 6.47% of the instances predicted as class 1 were actually class 0. The overall false alarm rate is 0.3671 (36.71%), reflecting the rate of false alarms across both classes.

Attack Class	Precision	Recall	F-score	False Alarm – FPR	Support
0	0.69	0.94	0.79	0.6695	73495
1	0.77	0.33	0.46	0.0647	46841

Confusion Matrix:

68740	4755
31360	15481

7.2 Comparison of Result – Dataset 2

Algorithms	Accuracy	Precision	Recall	F-Score	FAR	Train Time (s)	Test Time (s)	Error Rate
Decision Tree	0.8651	0.8726	0.8427	0.8529	0.1573	2.87	0.06	0.1349
Random Forest	0.8700	0.8798	0.8469	0.8579	0.1531	76.29	3.97	0.1300
K - Neighbours	0.8363	0.8472	0.8073	0.8189	0.1927	0.06	185.31	0.1637
Logistic Regression	0.6882	0.7631	0.6066	0.5827	0.3934	4.25	0.01	0.3118
Naïve Bayes	0.6999	0.7259	0.6329	0.6268	0.3671	0.15	0.05	0.3001

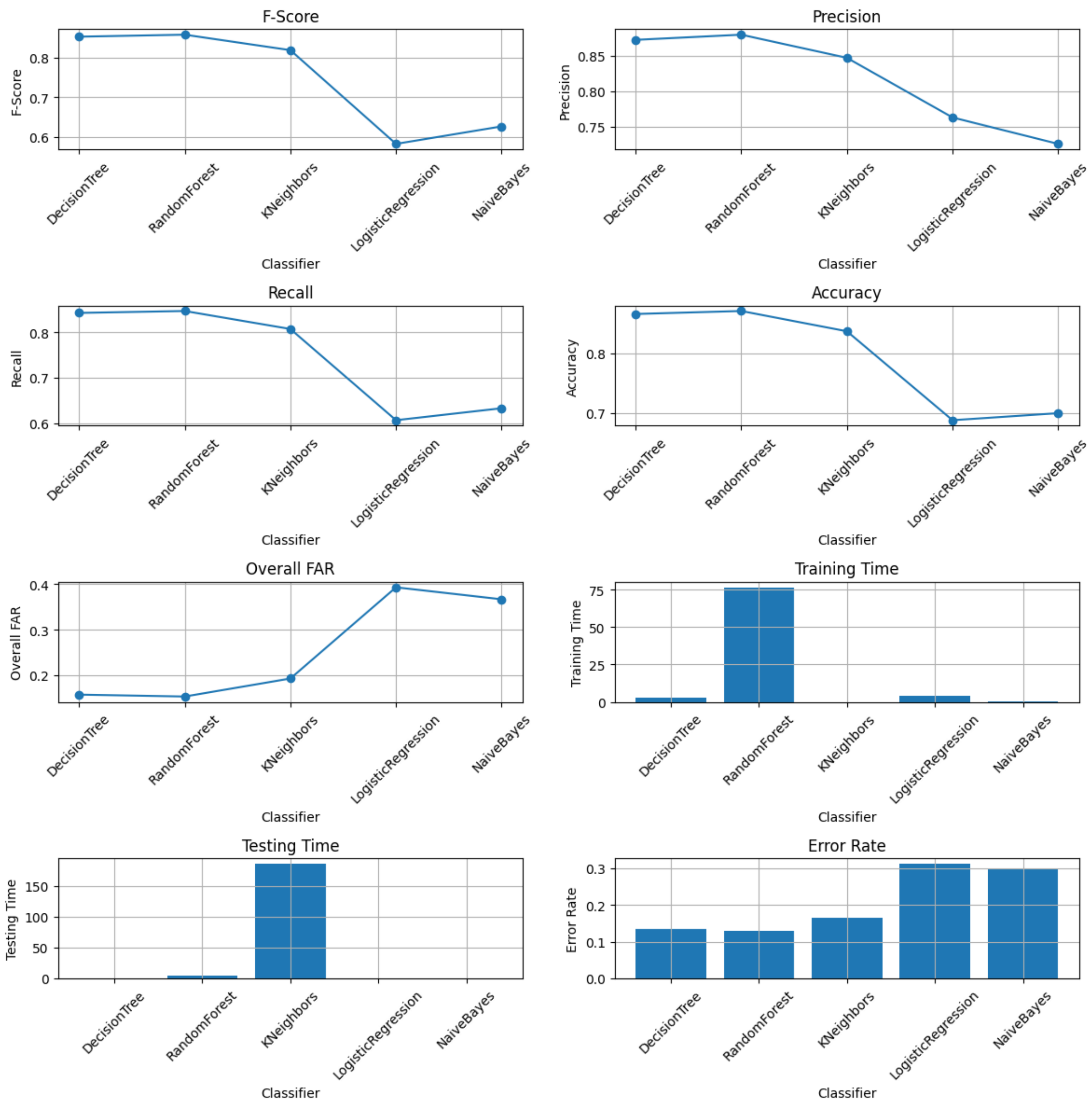
The analysis of machine learning algorithms on Dataset 2 reveals valuable insights into their performance. Random Forest stands out as the top performer, boasting an accuracy of 87.00%, closely followed by Decision Tree with 86.51%. These algorithms exhibit a strong balance between precision and recall, effectively making accurate positive predictions while capturing positive instances. However, Random Forest demands more substantial training time, while Decision Tree is remarkably efficient in testing.

K-Nearest Neighbors (KNN) demonstrates a competitive accuracy of 83.63%, with a commendable balance between precision and recall. Nevertheless, its extensive testing time may be a concern in real-time applications. Logistic Regression and Naïve Bayes, on the other hand, exhibit lower accuracy and face challenges in capturing positive instances, as indicated by their lower recall scores. Logistic Regression is notably fast in testing but consumes more time during training. In contrast, Naïve Bayes offers quick training and testing times.

When considering false alarm rates (FAR), Random Forest and Decision Tree outshine the rest, with lower FAR values signifying fewer false alarms. Logistic Regression and Naïve Bayes show higher FAR values, indicating a higher rate of false alarms. It's important to note that each algorithm's suitability depends on the specific application's priorities and efficiency requirements. Random Forest and Decision Tree are suitable for applications where precision and recall are crucial, while Logistic Regression and Naïve Bayes might be considered for situations where speed in prediction is essential, albeit with some trade-offs in accuracy.

8. Data Visualization – Dataset 1

8.1 Comparison of different algorithms



- ➢ The accuracy graph shows the accuracy of each classifier as the number of trees in the Random Forest Classifier increases. The accuracy of the Random Forest Classifier increases as the number of trees increases, but it starts to plateau after a certain number of trees.
- ➢ The training time graph shows the training time of each classifier as the number of trees in the Random Forest Classifier increases. The training time of the Random Forest Classifier increases as the number of trees increases, but it is still relatively fast compared to the other classifiers.
- ➢ The test time graph shows the test time of each classifier as the number of trees in the Random Forest Classifier increases. The test time of the Random Forest Classifier increases as the number of trees increases, but it is still relatively fast compared to the other classifiers.

- ➢ The error graph shows the error rate of the Random Forest Classifier as the number of trees increases. The error rate is the fraction of instances that are misclassified. The error rate of the Random Forest Classifier decreases as the number of trees increases.

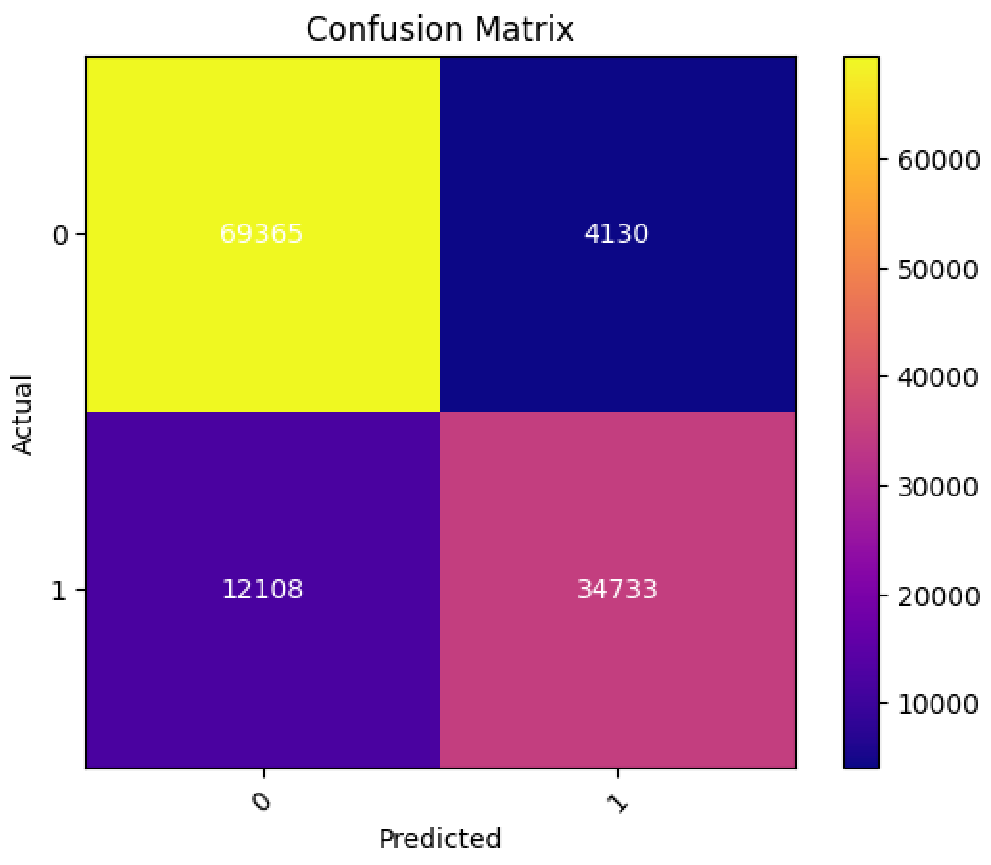
Based on the analysis of the accuracy, training time, test time, and error graph, the following conclusions can be drawn:

- ✓ The Random Forest Classifier is the most accurate classifier, followed by the SVC and the Decision Tree Classifier.
- ✓ The Random Forest Classifier has a slightly longer training time and test time than the other classifiers.
- ✓ The error graph of the Random Forest Classifier shows that the accuracy of the classifier increases as the number of trees increases. However, the accuracy starts to plateau after a certain number of trees.

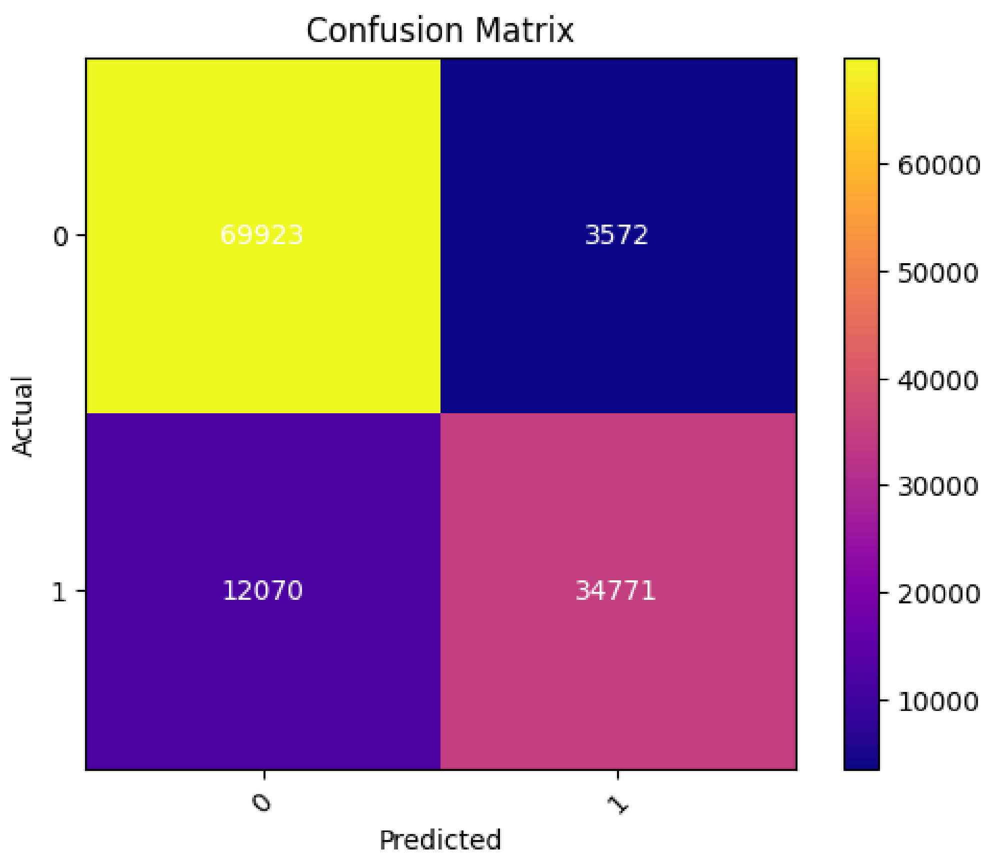
Overall, the Random Forest Classifier is the best choice for this classification task. It has the best overall performance, and it is relatively fast to train and classify instances.

8.2 Confusion Matrices for all algorithms

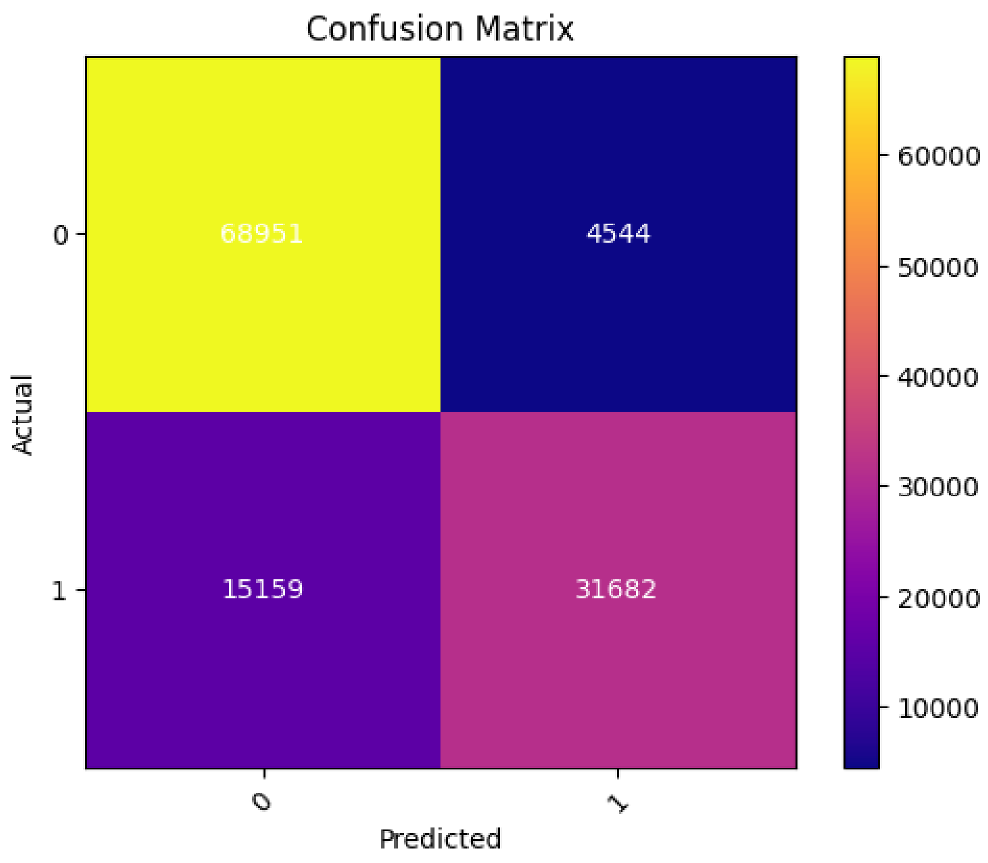
Heatmaps of all algorithms for Dataset 2



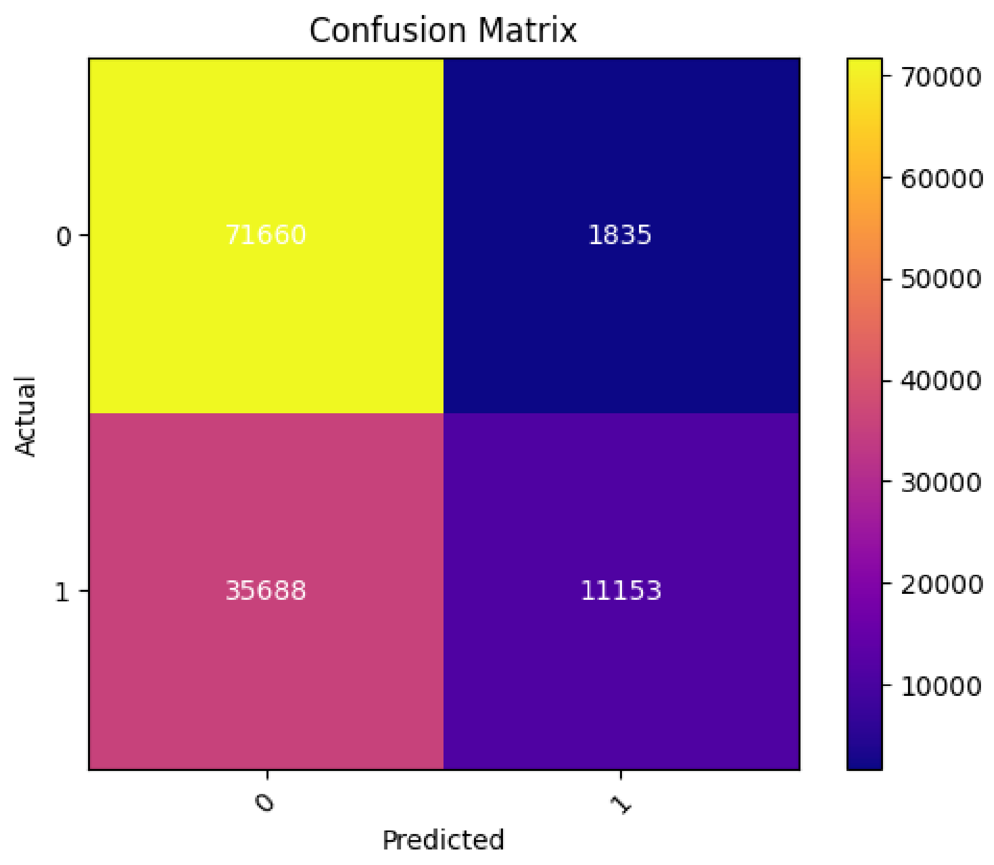
Decision Tree



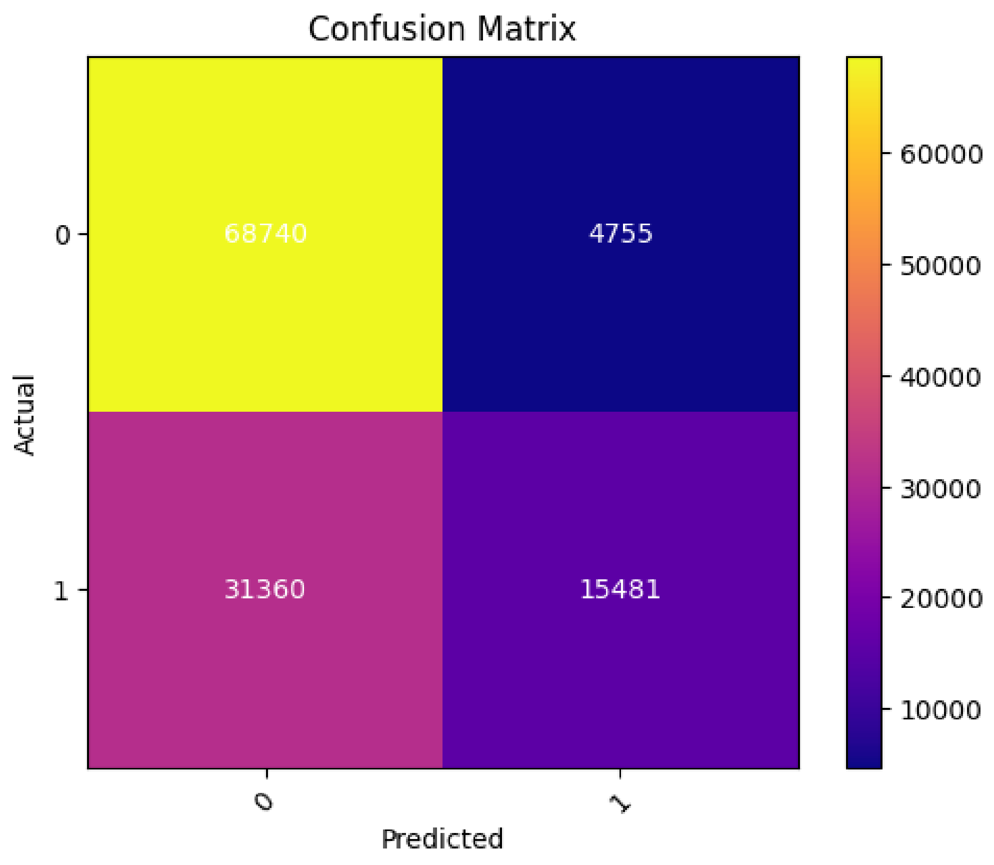
Random Forest



K – Neighbours



Logistic Regression



Naïve Bayes