PASSWORD STRENGTH PREDICTION USING MACHINE LEARNING ALGORITHMS

Name: V. Divya
HT No: 2203A52130
Batch No: 33

ABSTRACT

The Password Strength Predictor is an innovative tool designed to assess the strength of user-generated passwords. In the digital age, where cyber threats are rampant, the security of personal and sensitive information hinges on the robustness of passwords. This project employs machine learning algorithms to classify passwords into categories such as weak, medium, and strong, based on various criteria including length, complexity, and entropy. The system provides users with instant feedback on the strength of their passwords, along with suggestions for improvement, thereby promoting cybersecurity awareness and enhancing online security. Through the fusion of machine learning techniques and user-friendly interfaces, the Password Strength Predictor aims to empower users with the knowledge and tools necessary to create and maintain secure passwords in an increasingly interconnected world.

Central to the efficacy of the Password Strength Predictor is its ability to provide instantaneous feedback to users regarding the strength of their chosen passwords. Through intuitive and user-friendly interfaces, individuals are promptly informed of the adequacy or inadequacy of their password choices, alongside tailored suggestions for improvement. By bridging the gap between technological sophistication and user accessibility, this tool serves as a catalyst for promoting cybersecurity awareness among users of all backgrounds and levels of technical proficiency.

The fusion of machine learning methodologies with intuitive interfaces represents a cornerstone of the Password Strength Predictor's mission Ultimately, the overarching goal of the Password Strength Predictor is to install a culture of proactive cybersecurity practices, thereby enhancing the resilience of digital ecosystems against evolving threats.

INTRODUCTION

In today's digital era, the majority of our interactions and transactions occur online, the importance of password security had been very secure. The strength of passwords plays a critical role in safeguarding sensitive information from unauthorized access. Therefore, developing effective tools to assess and enhance password strength is more crucial now a days in cybersecurity measures.

The Password Strength Predictor project addresses the critical needs and offers a solution to assess the strength of passwords generated by users. By using machine learning techniques, this project aims to classify passwords such as weak, medium, and strong based on a variety of factors including length, complexity, and decline. To the development of a user-friendly interface, users receive immediate feedback on their strength, along with some related suggestions for enhancing security.

The primary objective of this project is to provide users with a reliable tool that can assess the strength of their chosen passwords and offer suggestions for strengthening them if necessary. By giving every individuals to create stronger passwords, we aim to take risk of unauthorized access to sensitive accounts and data, by enhancing overall cybersecurity posture.

Strong passwords are critical to protecting not only personal accounts, but also corporate networks, government systems, and critical infrastructure. By encouraging the adoption of strong passwords, the project contributes to strengthening the overall resilience of digital infrastructure against cyber threats, thereby protecting critical assets and minimizing the potential for disruptive cyber incidents.

Beyond the immediate benefits, the Password Strength Predictor project has the potential for long-term positive results in cybersecurity. By fostering a culture of password security awareness, the project lays the foundation for sustainable cybersecurity practices that can withstand evolving threats. In addition, as the project evolves and adapts to new trends and techniques in password security, it serves as a dynamic resource for staying ahead of evolving cyber threats.

In conclusion, the Password Strength Predictor project represents a beacon of innovation and empowerment in the ongoing digital security effort. By harnessing the power of machine learning, the project offers users a reliable tool for evaluating and increasing password strength, thereby strengthening defenses against unauthorized access and strengthening cybersecurity resilience. With its user-centric approach, practical recommendations and commitment to continuous improvement, the project embodies the transformative potential of technology in shaping a safer digital future.

METHODOLOGY

To develop a machine learning version for predicting the energy of passwords, step one is to accumulate a complete dataset containing a huge wide variety of passwords in conjunction with their corresponding energy labels. The first step in creating the machine learning version of password energy prediction is to create a dataset with a large number of passwords and associated energy labels. Then, a set of methods are used to capture various characteristics of the password such as the password period, the presence of capital/lowercase/numerical characters, special characters and other relevant information. These characteristics are then used as inputs in the dataset to represent each of the password entries.

The machine learning model is then able to analyze the patterns and correlations related to the password electricity. The next step is to choose the exact set of system mastering rules for the categorization. These features act as inputs for the model to look for patterns and correlations associated with password electricity. Once the data set is ready, the next step is to choose the right set of system mastering rules for classifying the data. Depending on the type of problem and the specific characteristics of the target object (password power), Logistic Regression, Decision Trees, Random Forests, and SVM (Assist Vector Machines) are often used along with type algorithms. The dataset size and complexity, the version's interpretability, and the overall performance may also influence the set of rules chosen.

Once the rules are determined, the data set is broken down into units for testing and training to evaluate the model’s performance. During the training stage, the version reduces the number of false predictions by making iterative changes to parameters as it finds patterns in the data sets. If the model does exceptionally well, you can use it to break down passwords into three categories: medium, strong, and weak, based on their function representations.

In summary, growing a machine gaining knowledge of version for predicting password power entails amassing a dataset, deciding on appropriate capabilities, deciding on a class algorithm, training and comparing the version, and ultimately deploying it for real-world use. By leveraging the energy of gadget studying, this technique allows automatic and scalable evaluation of password strength, thereby enhancing cybersecurity practices in an increasingly more interconnected international.

DATASET AND AUGMENTATION

The process of adding or modifying the original facts of a data set while performing its original characteristics is called data set augmentation. While predicting the password strength, we use extension methods to generate more password samples that follows some set of guidelines. By performing this set of guide lines this improves the quality of the material and strengthens the comprehensible version of the system.

The password composition rules in the material are predefined in the above mentioned case. These settings require at least characters or a combination of letters, numbers, and special characters. Each password is written according to these instructions and stored in SQLite format. Each password uses both uppercase and lowercase letters.

New password attempts can be made using a variety of methods while maintaining the integrity of the original data. One typical strategy is to manipulate or change existing passwords. These methods offer variations that follow established guidelines.

IMPLEMENTATION

WE Implement different machine learning algorithms or models to the dataset like:

KNN Classification: K-Nearest Neighbours is a simple set of rules that classifies a facts point primarily based on the majority class of its k-nearest neighbours. It calls for no schooling segment but may be computationally luxurious throughout inference, mainly with large datasets.

Logistic Regression: Despite its name, logistic regression is a linear version used for binary type. It learns the connection between the independent variables and the opportunity of a selected final results. It's rapid to teach and interpret however assumes a linear courting between capabilities and the log-odds of the outcome.

CNN Classification: Convolutional Neural Networks are mainly used for picture class obligations. They encompass convolutional layers that routinely study features from the enter data. CNNs have accomplished ultra-modern performance in diverse computer vision duties however require a big amount of records and computational assets for education.

SVM (Support Vector Machine): SVM is a powerful classification set of rules that unearths the hyperplane that quality separates the lessons in the function area. It can handle excessive-dimensional information and is powerful in cases in which the quantity of functions exceeds the quantity of samples. SVMs can use special kernel functions to address non-linear relationships between functions.

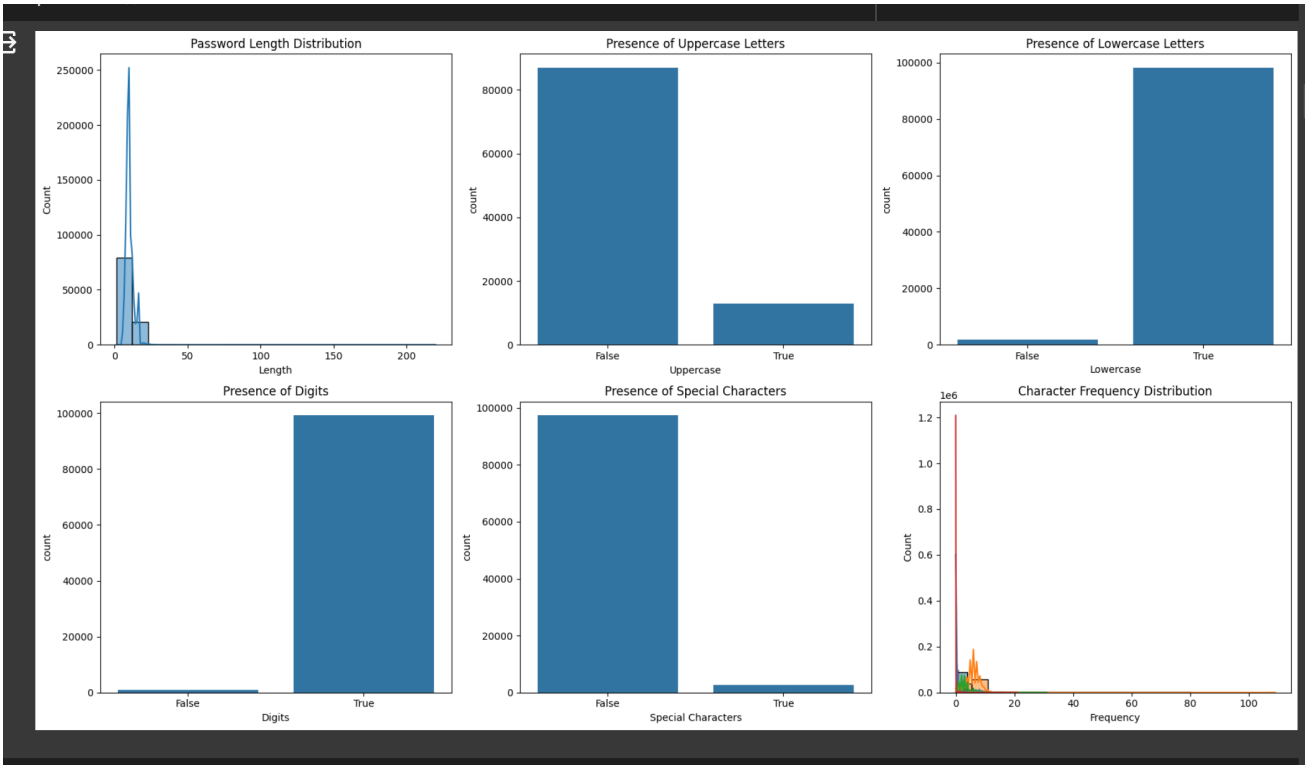
Decision Tree: Decision Trees recursively partition the function space into subsets, making selections based totally on function values. They are smooth to interpret and visualize, making them useful for information feature importance. However, they generally tend to overfit the education information if not pruned nicely.

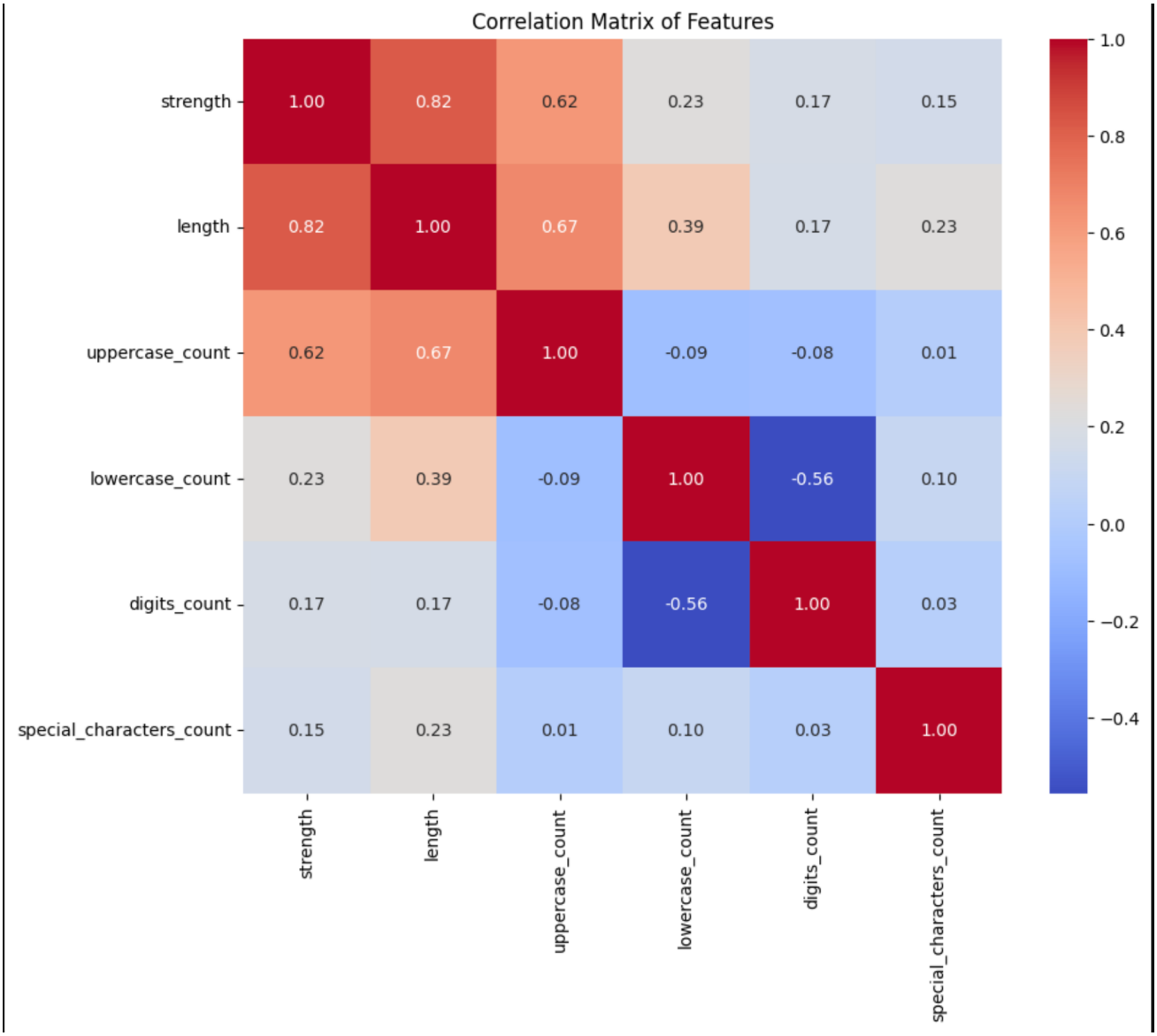
Multi-Layer Perceptron (MLP): MLP is a type of artificial neural community composed of more than one layers of nodes (or neurons) which could study complex patterns in the statistics. It is a versatile version capable of approximating any non-linear characteristic given sufficient facts and computational sources. Training MLPs regularly calls for careful tuning of hyperparameters to keep away from overfitting.

Results:

Feature Extraction:

	password	strength	length	uppercase	lowercase	digits	special_characters	uppercase_count	lowercase_cou
0	zxe870819	1	9	False	True	True	False	0	
1	xw46454nr23l	1	12	False	True	True	False	0	
2	soporte13	1	9	False	True	True	False	0	
3	accounts6000webhost.com	2	23	False	True	True	True	0	
4	c443balg	1	8	False	True	True	False	0	




Correlation Matrix of Features
strength
length
uppercase_count
lowercase_count
digits_count
special_characters_count
1.00 0.82 0.62 0.23 0.17 0.15
0.82 1.00 0.67 0.39 0.17 0.23
0.62 0.67 1.00 -0.09 -0.08 0.01
0.23 0.39 -0.09 1.00 -0.56 0.10
0.17 0.17 -0.08 -0.56 1.00 0.03
0.15 0.23 0.01 0.10 0.03 1.00
strength
length
uppercase_count
lowercase_count
digits_count
special_characters_count
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

- **Random Forest:**

Random Forest classification is an ensemble learning method that constructs a multitude of decision trees during training. This method is used in various fields, including finance, healthcare, and marketing, due to its different types and effectiveness.

	precision	recall	f1-score	support
0	1.00	1.00	1.00	2700
1	1.00	1.00	1.00	14852
2	1.00	1.00	1.00	2448
accuracy			1.00	20000
macro avg	1.00	1.00	1.00	20000
weighted avg	1.00	1.00	1.00	20000

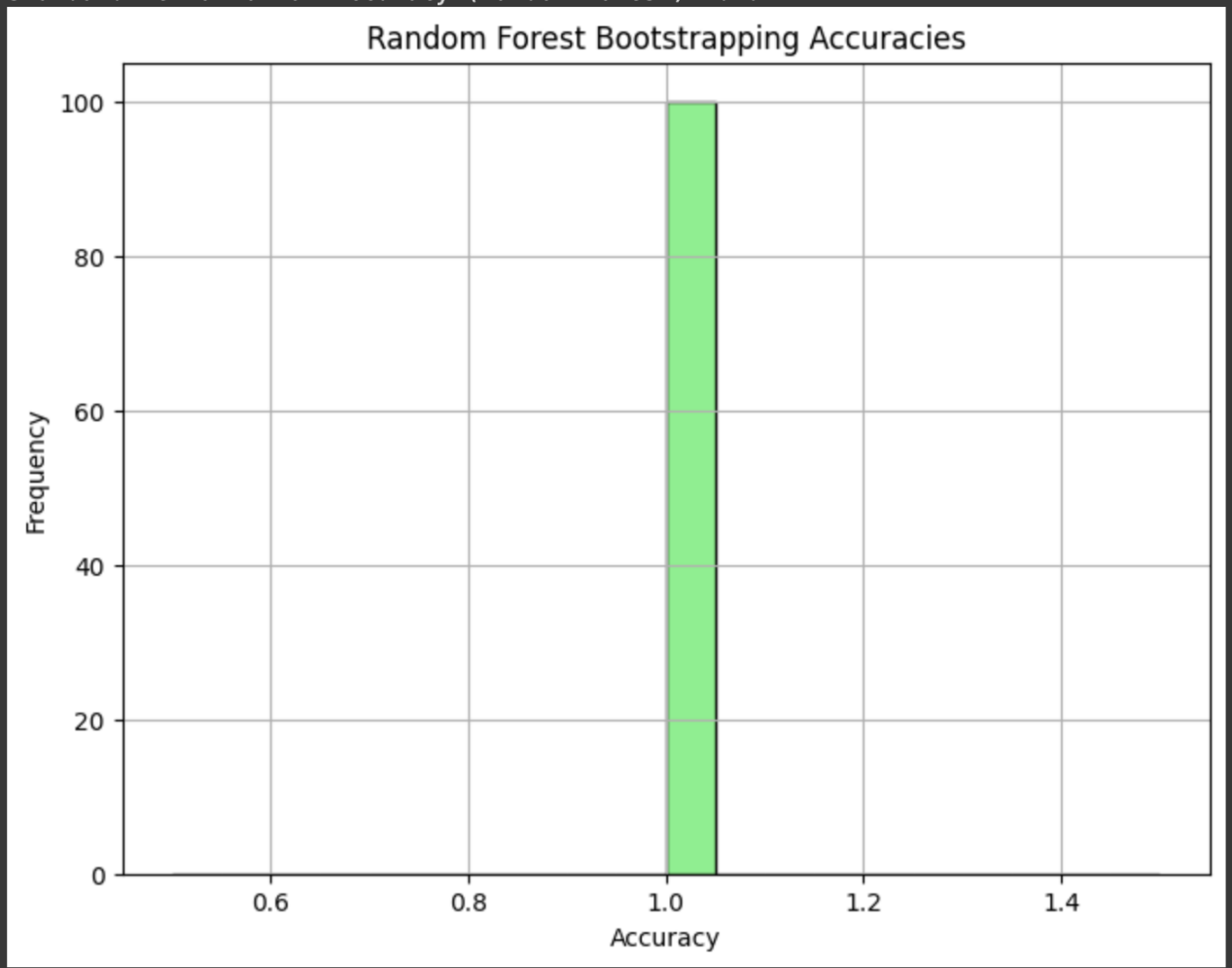
cv_scores: [1. 1. 1. 1. 1.]

1.0

Bootstrapping:

In random forest, bootstrap sampling is used to create multiple subsets of the original dataset. This bootstrap sample is used to train individual decision trees within the forest. This technique introduces randomness and diversity into the ensemble, enhancing the model's robustness and generalization performance by reducing overfitting.

```
Mean Accuracy (Random Forest): 1.0
Standard Deviation of Accuracy (Random Forest): 0.0
```



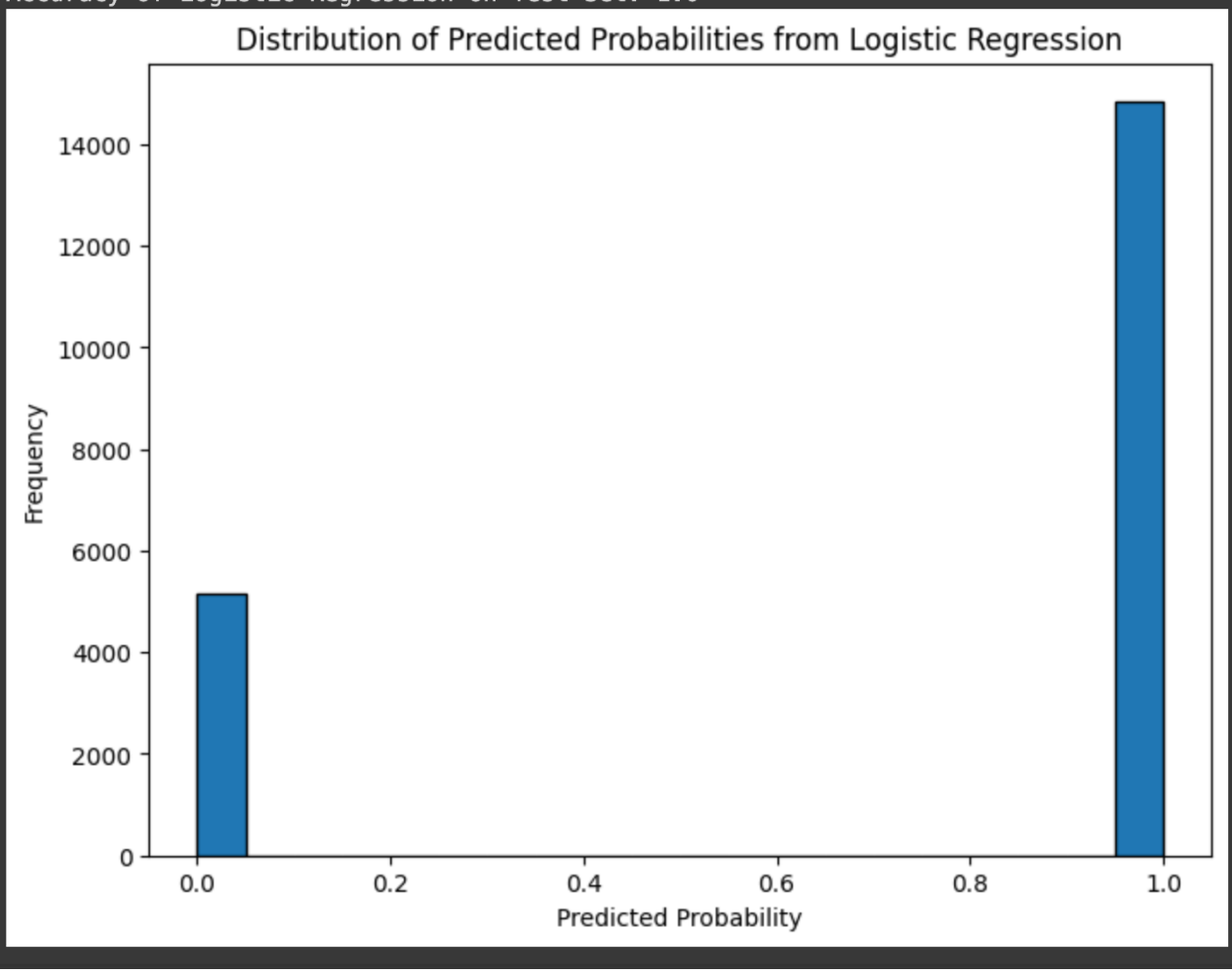
- **LOGISTIC REGRESSION:**

Classification tasks often use logistic regression. This technique looks at input data to predict if an event will happen. It uses a special math function to relate the input data to the event's likelihood. Logistic regression gives probabilities and decision boundaries. It works well for datasets where outcomes separate into groups. Many fields like finance, marketing, and healthcare use logistic regression. The method is simple, effective, and easy to understand.

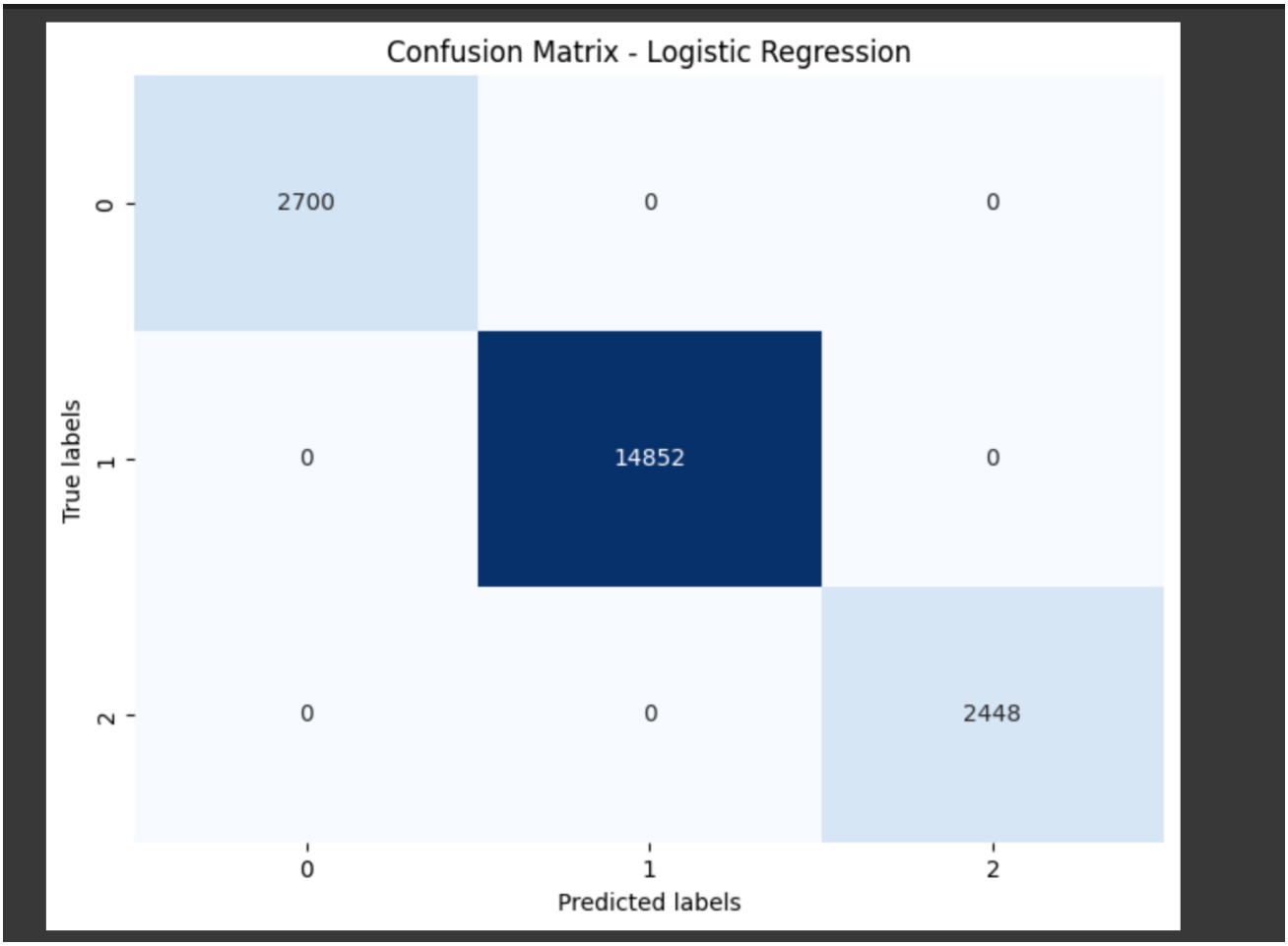
Logistic Regression - Train Accuracy: 1.0

Logistic Regression - Test Accuracy: 1.0

```
Accuracy of Logistic Regression on Test Set: 1.0
```



Confusion Matrix For Logistic Regression:



The confusion matrix plays an important role in logistic regression to evaluate the performance of model. The confusion matrix gives the detailed information about the predicted and true labels of data.

$$\boldsymbol{Accuracy} = \frac{(\textbf{True Positives (TP)}+\textbf{True Negatives (TN)})}{(\boldsymbol{True\ Positives\ (TP)}+\boldsymbol{True\ Negatives\ (TN)}+\boldsymbol{False\ Positives\ (FP)}+\boldsymbol{False\ Negatives\ (FN)})} \textbf{....[1]}$$

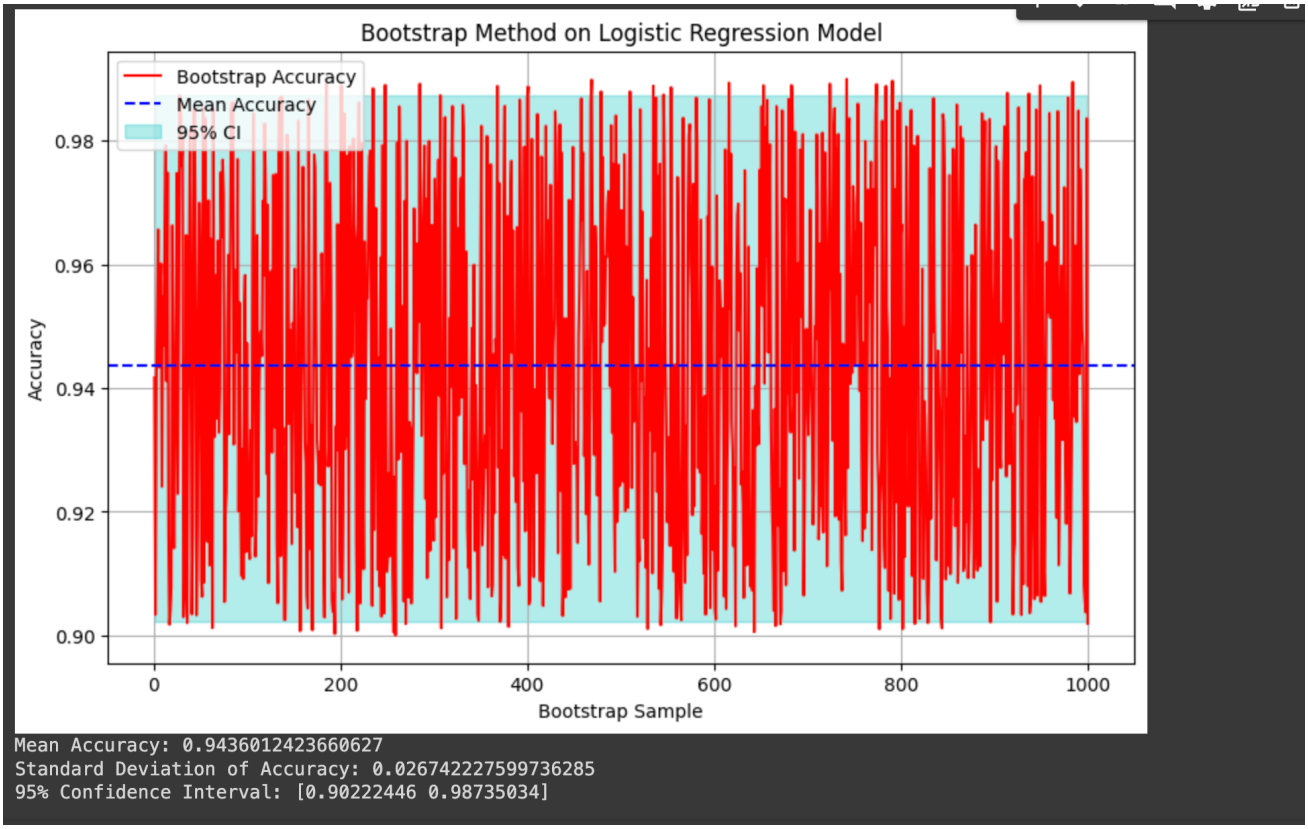
$$\boldsymbol{precision} = \frac{\boldsymbol{TruePositive}}{(\boldsymbol{TruePositive}+\boldsymbol{FalsePositive})} \textbf{...[2]}$$

$$\boldsymbol{Recall} = \frac{\boldsymbol{TruePositive}}{(\boldsymbol{TruePositive}+\boldsymbol{FalseNegative})} \textbf{.....[3]}$$

$$\textbf{F1}=\frac{(2\times\boldsymbol{Precision}\times\boldsymbol{Recall})}{(\boldsymbol{Precision}+\boldsymbol{Recall})} \textbf{,.....[4]}$$

Bootstrapping:

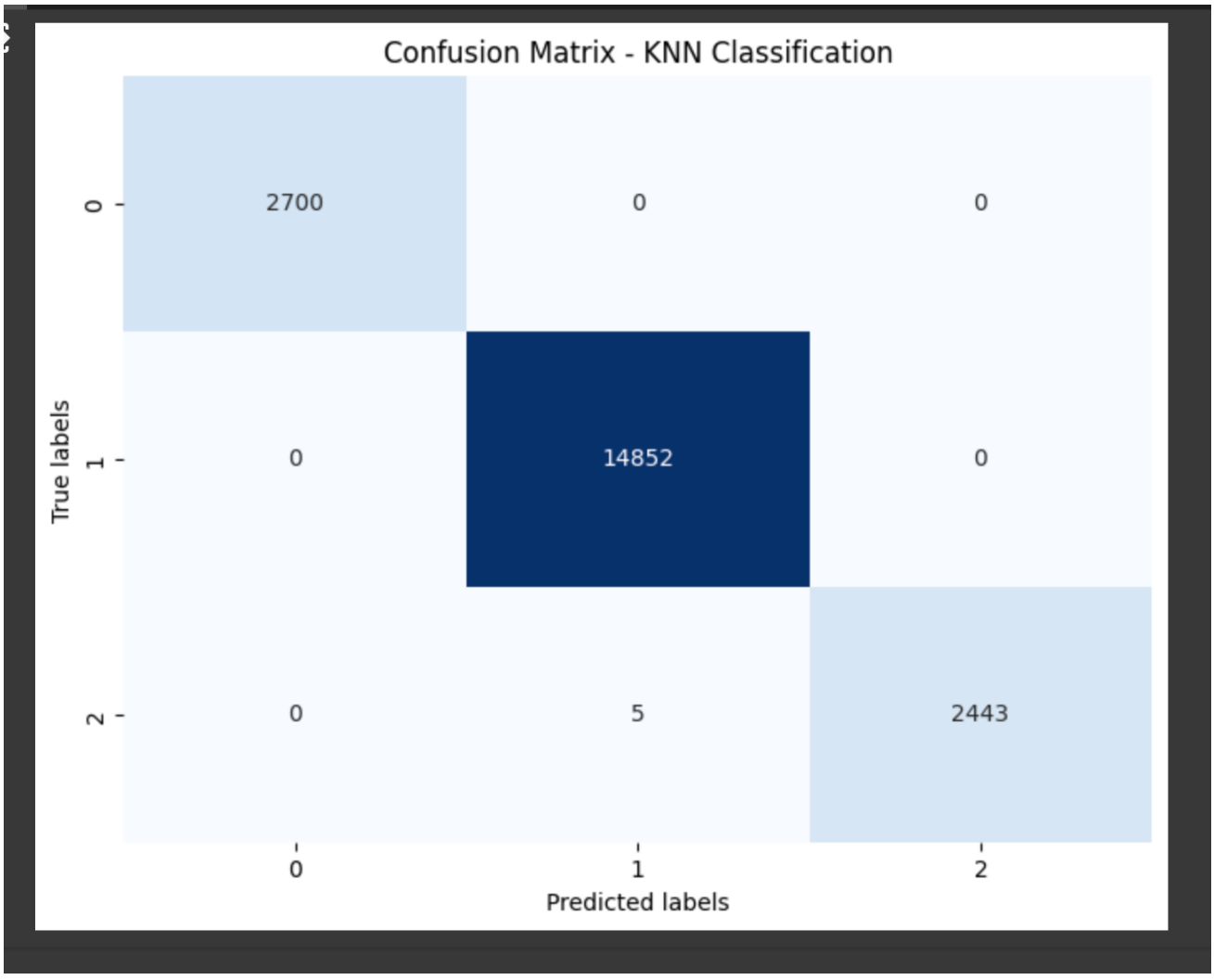
Bootstrap is a method in which we repeatedly collect samples from data allowing us to estimate how uncertain the statics can be.It helps to understand the reliability of results without the mean accuracy.95% confidence interval dashed line.



- **KNN Classification:**

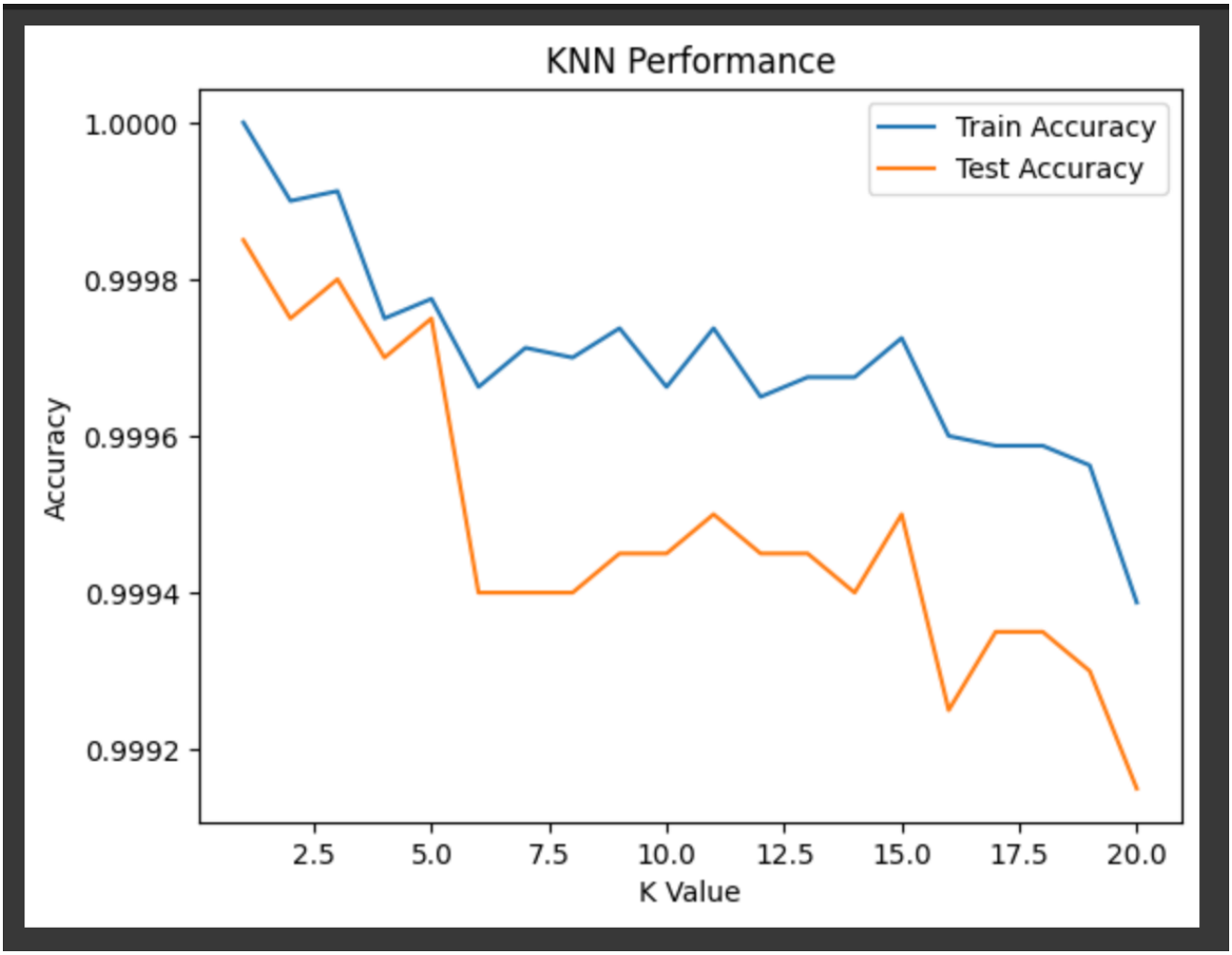
The KNN classification, classifies the data points according to their accurate to other points in the feature space can be done using a simple but powerful K-Nearest Neighbors (KNN) method. It classifies a data point by a majority vote of its k nearest neighbours in the feature space, making it a non-parametric and lazy learning algorithm.

Confusion Matrix for KNN classification:



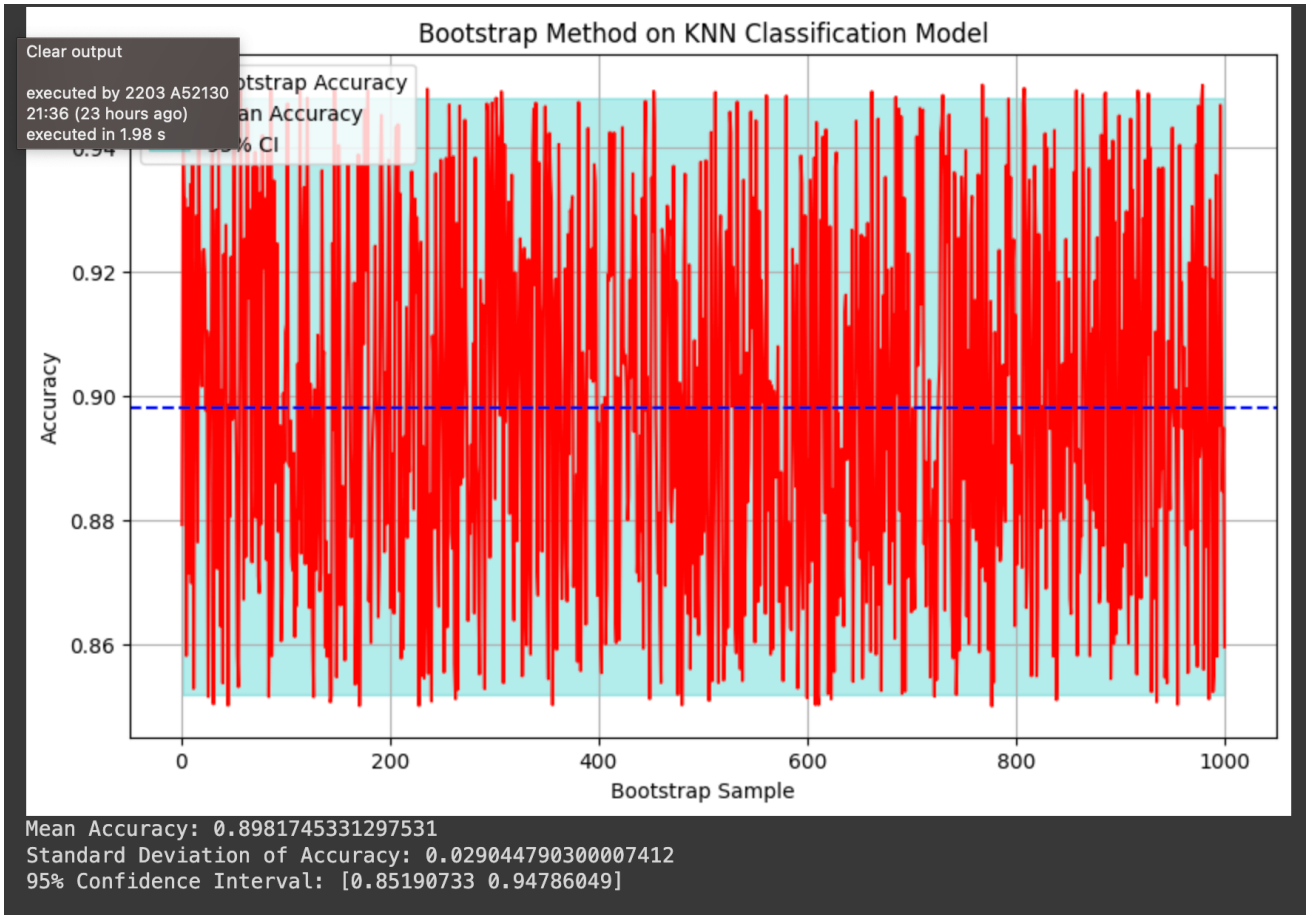
KNN - Train Accuracy: 0.999775

KNN - Test Accuracy: 0.99975



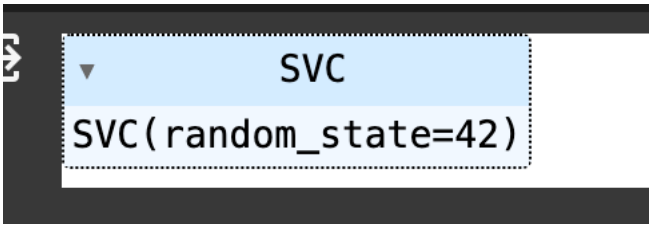
Bootstrapping:

KNN bootstrapping involves resampling the dataset with replacement to create multiple bootstrap samples, then applying KNN classification on each sample and aggregating the results, typically using techniques like bagging or boosting, to improve the model's performance and robustness.


Clear output
executed by 2203 A52130
21:36 (23 hours ago)
executed in 1.98 s
Bootstrap Method on KNN Classification Model
Accuracy
0.92
0.90
0.88
0.86
0
200
400
600
800
1000
Bootstrap Sample
Mean Accuracy: 0.8981745331297531
Standard Deviation of Accuracy: 0.029044790300007412
95% Confidence Interval: [0.85190733 0.94786049]

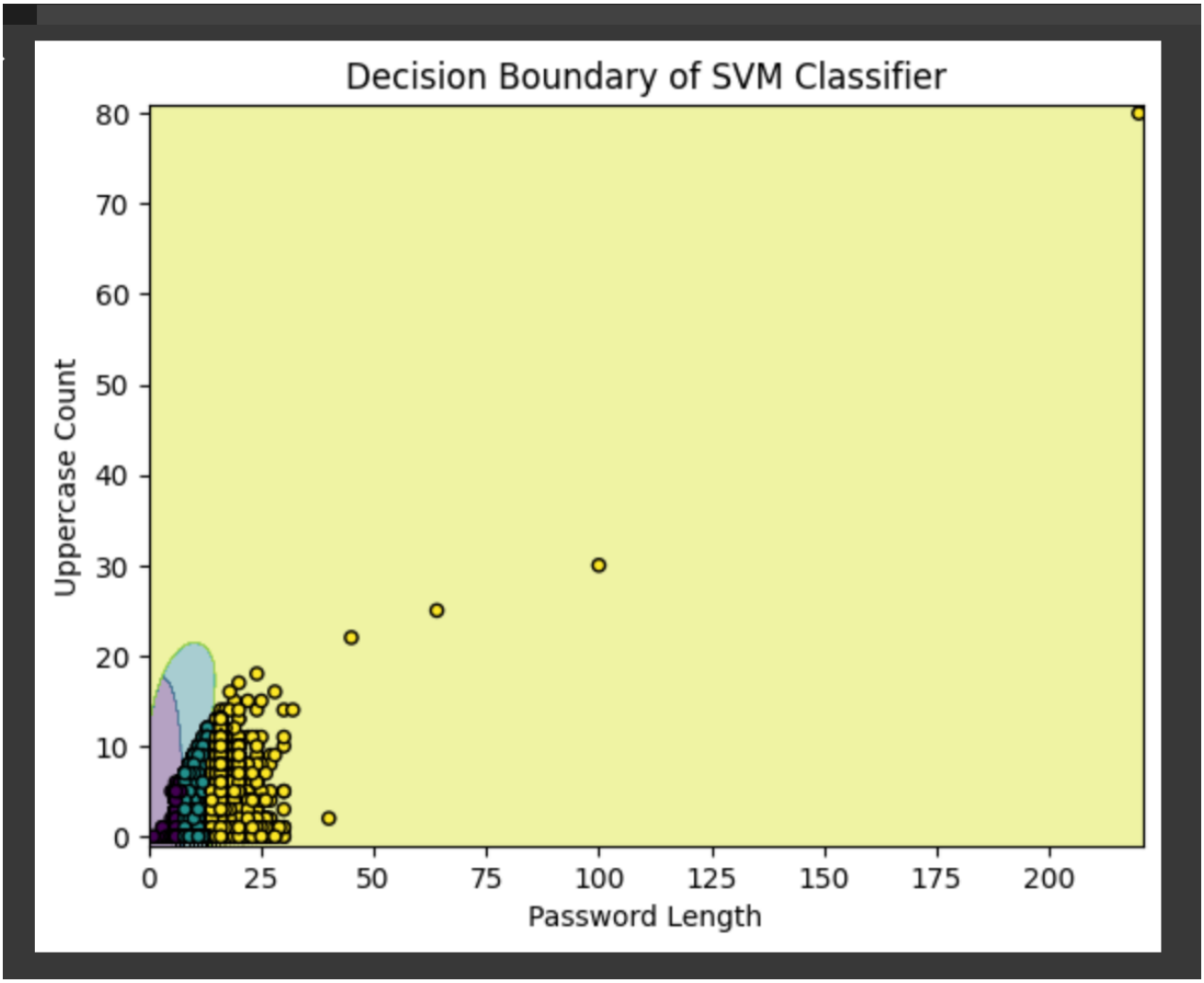
- **SVM Classification:**

Support Vector Machine (SVM) is a robust supervised learning algorithm among those used for classification and regression tasks. It does so by locating the hyperplane that best partions classes in the feature space, maximizing the margin between classes, and often using kernel tricks to deal with the linear separation problems effectively.



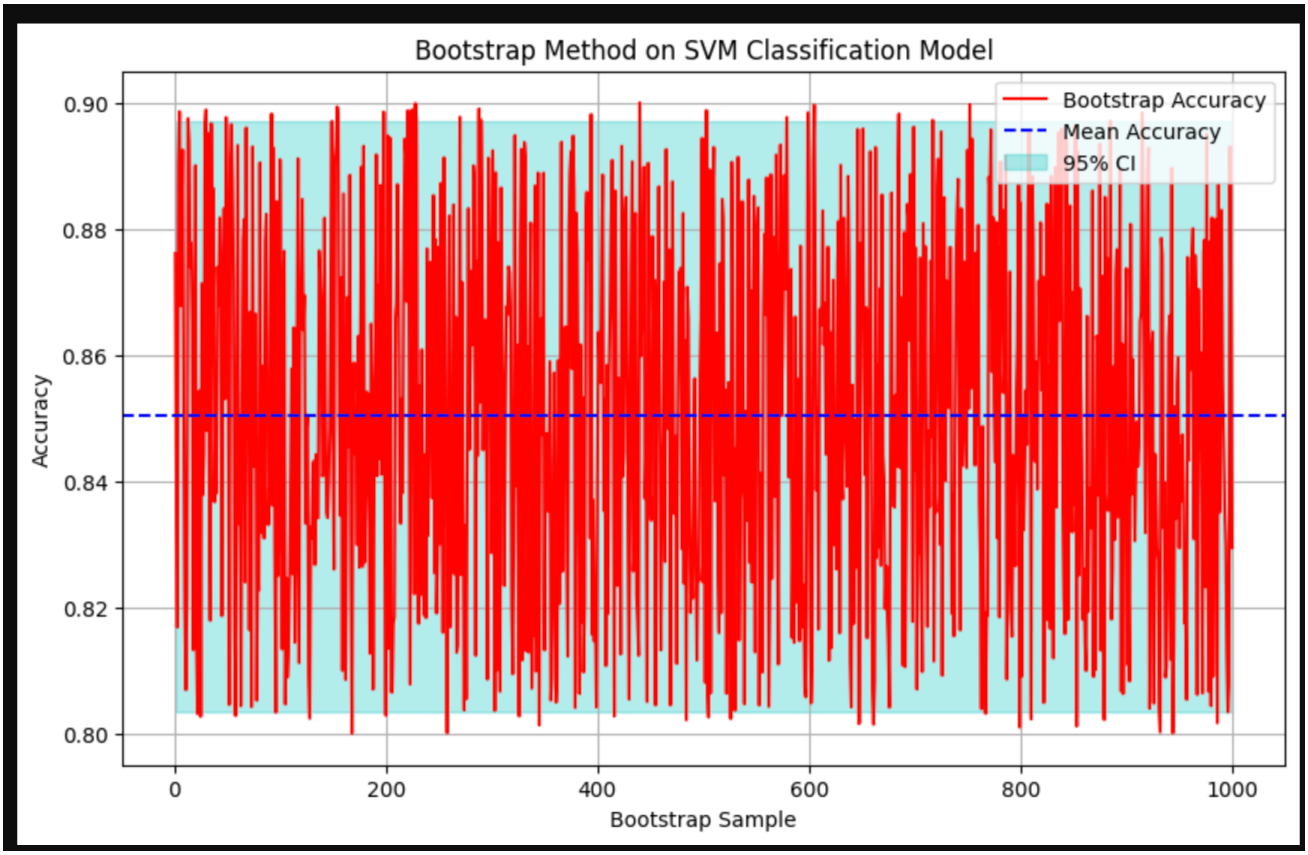
SVM - Train Accuracy: 0.9999875

SVM - Test Accuracy: 1.0



Bootstrapping:

Bootstrapped SVM represent a technique in which SVM classifiers are trained by resampling the datasets with replacement, creating many bootstrap samples each time. This representatives are then put together, usually using baying or const articles, to ameliorate the general performance and stability of the model.

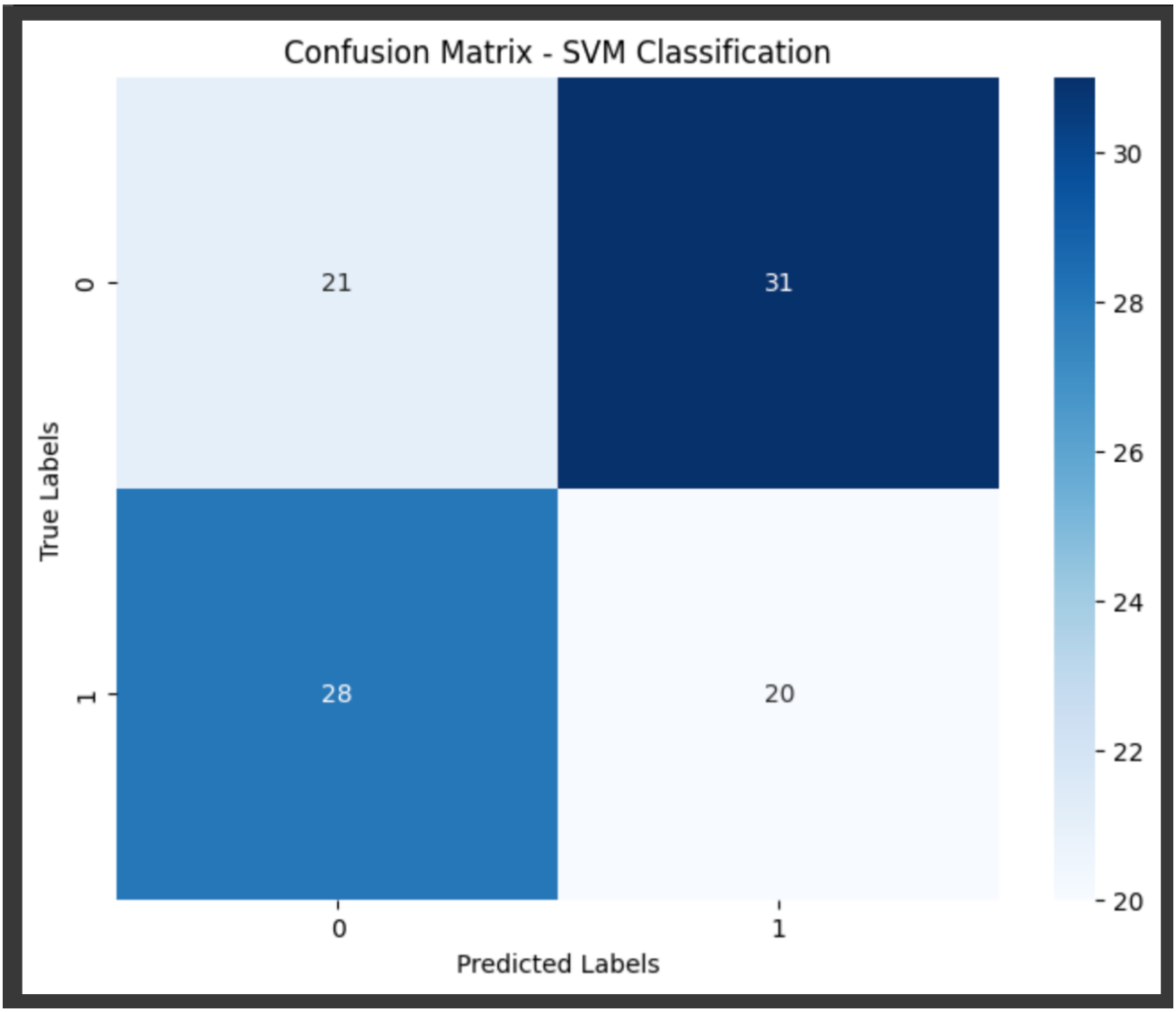


Mean Accuracy: 0.8503699862211405

Standard Deviation of Accuracy: 0.028330892308791475

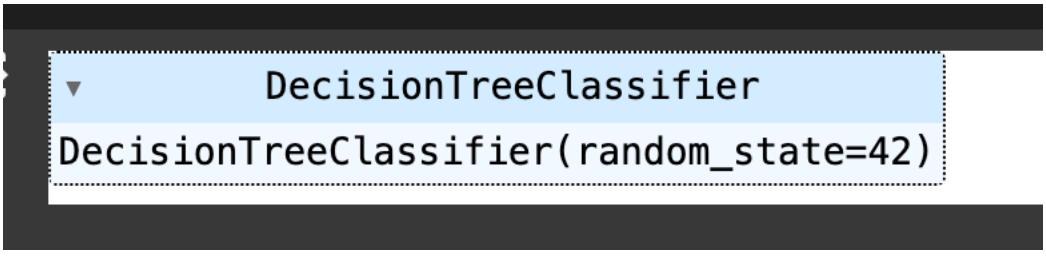
95% Confidence Interval: [0.80341608 0.89702777]

Confusion Matrix for the SVM Classifier:



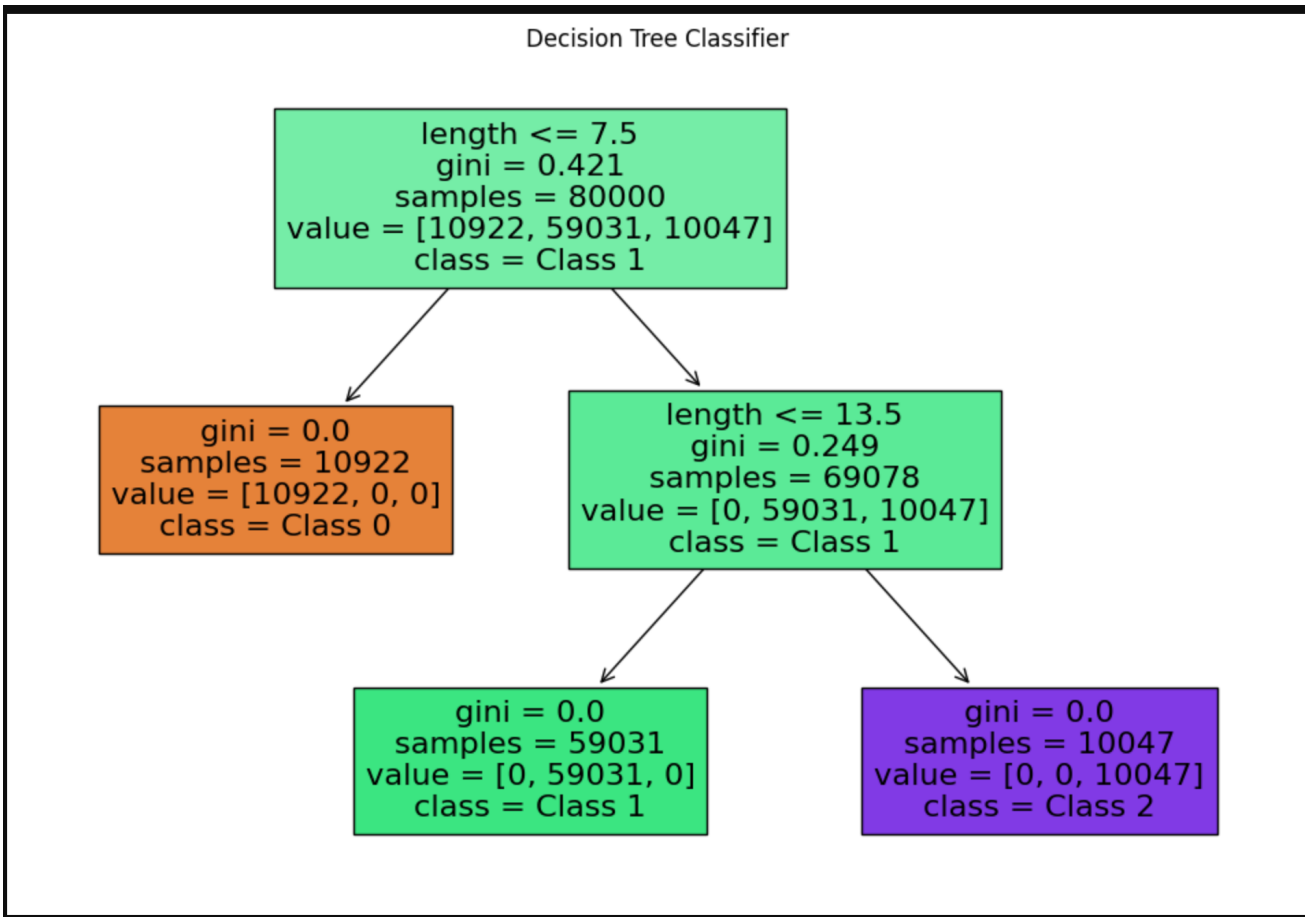
- **Decision Tree Classification:**

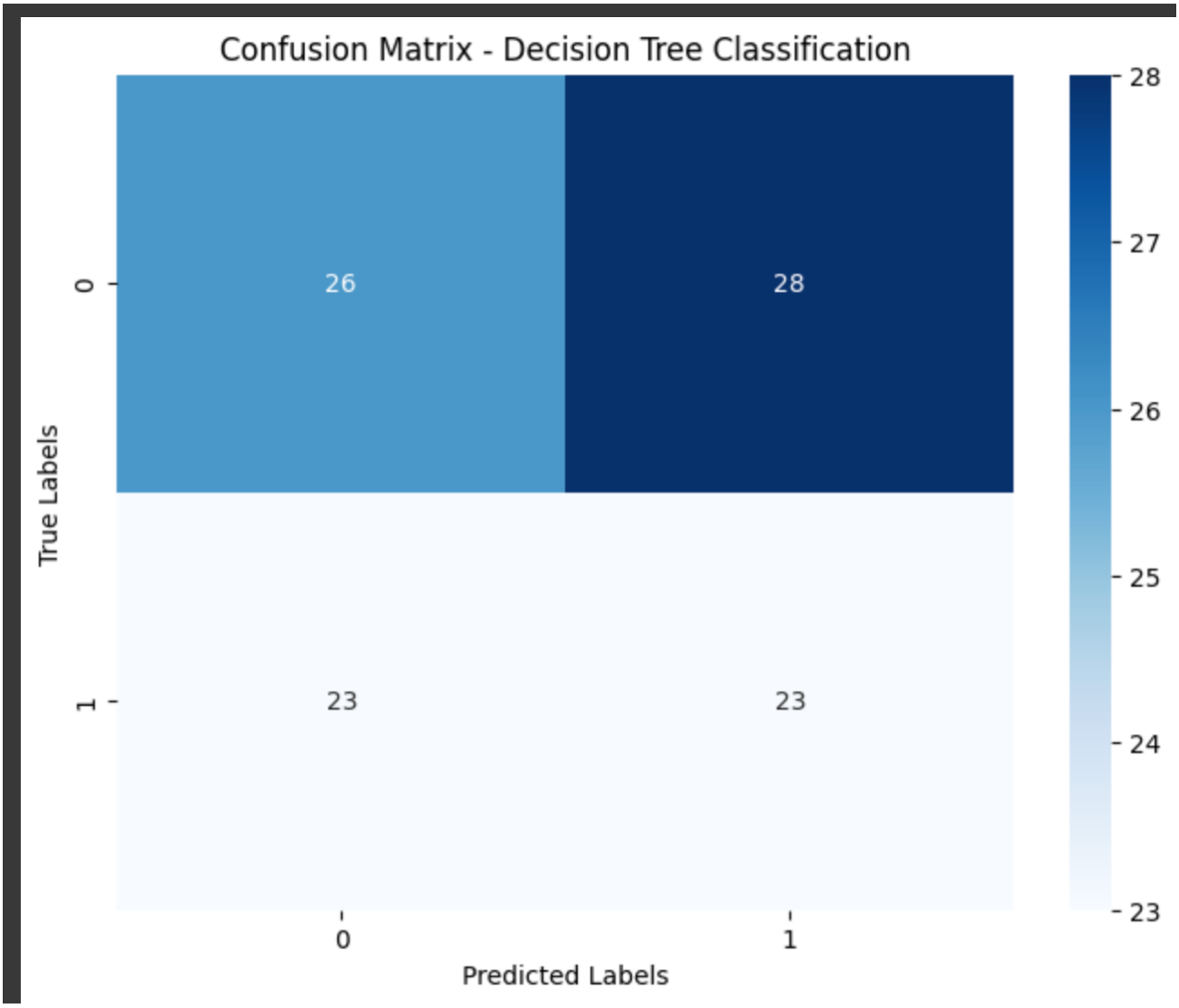
Decision Trees classification is the most used supervised learning algorithm that is a way of learning simple decision rules which are inferred from the data features. It actively divides the data into subsets by featuring the most important subset elements. Its main goal is to achieve homogeneity among the subsets concerning the target variable. This decision-tree process obtains a tree-like structure, where the inner nodes are decision based on some features, and the leaf nodes represents the class label.



Decision Tree - Train Accuracy: 1.0

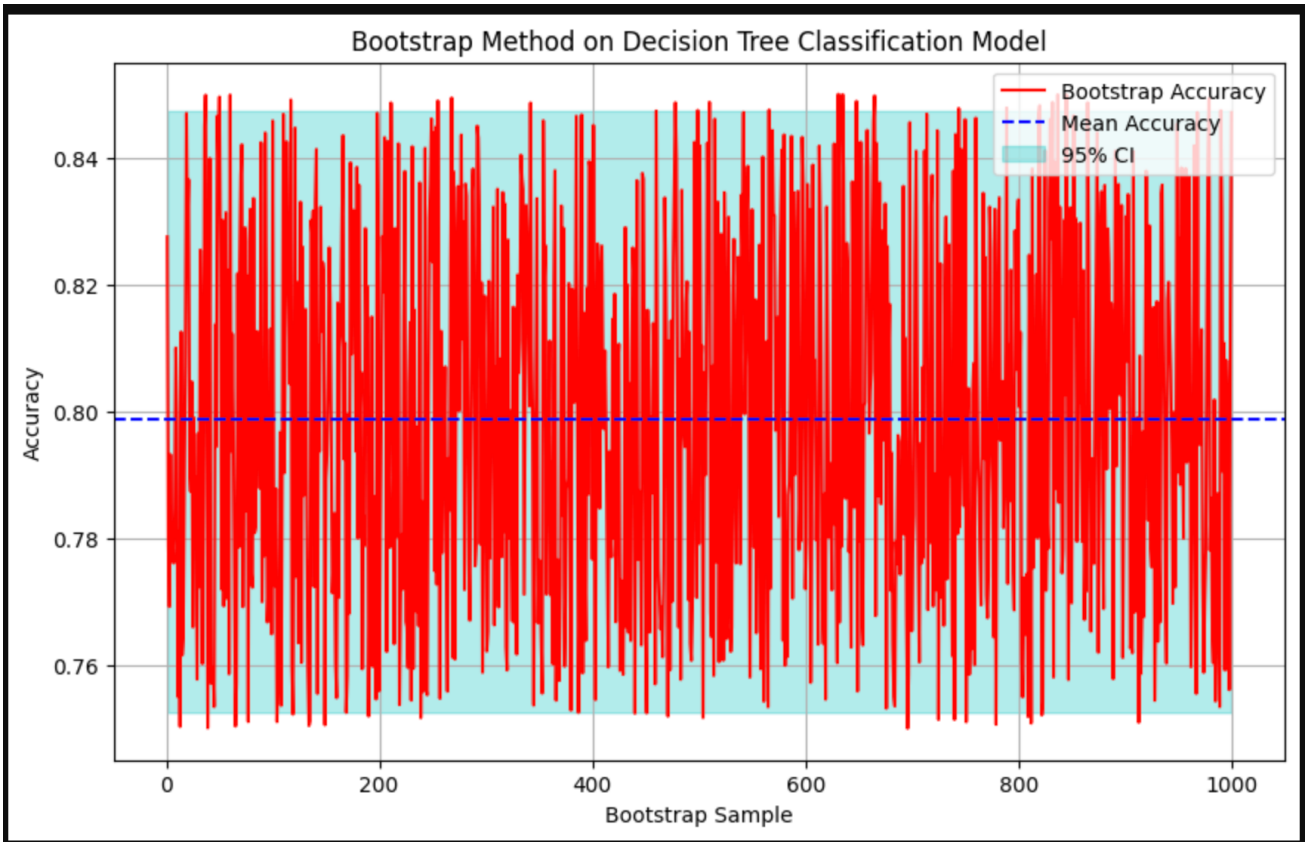
Decision Tree - Test Accuracy: 1.0


Decision Tree Classifier
length <= 7.5
gini = 0.421
samples = 80000
value = [10922, 59031, 10047]
class = Class 1
gini = 0.0
samples = 10922
value = [10922, 0, 0]
class = Class 0
length <= 13.5
gini = 0.249
samples = 69078
value = [0, 59031, 10047]
class = Class 1
gini = 0.0
samples = 59031
value = [0, 59031, 0]
class = Class 1
gini = 0.0
samples = 10047
value = [0, 0, 10047]
class = Class 2



Bootstrapping:

During bootstrapping, a decision tree classifier creation for bootstrapping entails generating numerous bootstrap samples from the dataset, training decision tree classifiers for each sample, and combining their predictions through techniques such as bagging or boosting. Hence, Gust reduces the variance and reduces overfitting.



Mean Accuracy: 0.7988147859458747

Standard Deviation of Accuracy: 0.028828859150022746

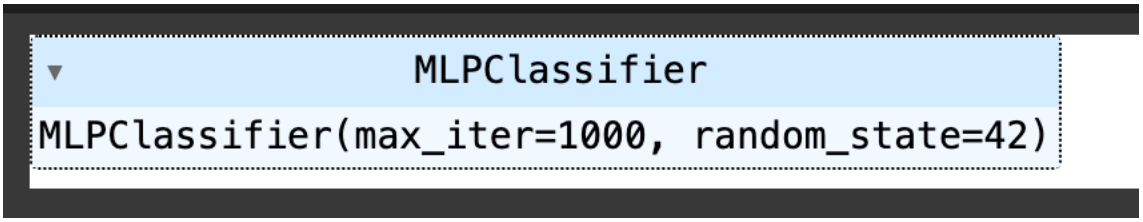
95% Confidence Interval: [0.75254248 0.84735398]

- **Multi-Layer Perceptron Classification**

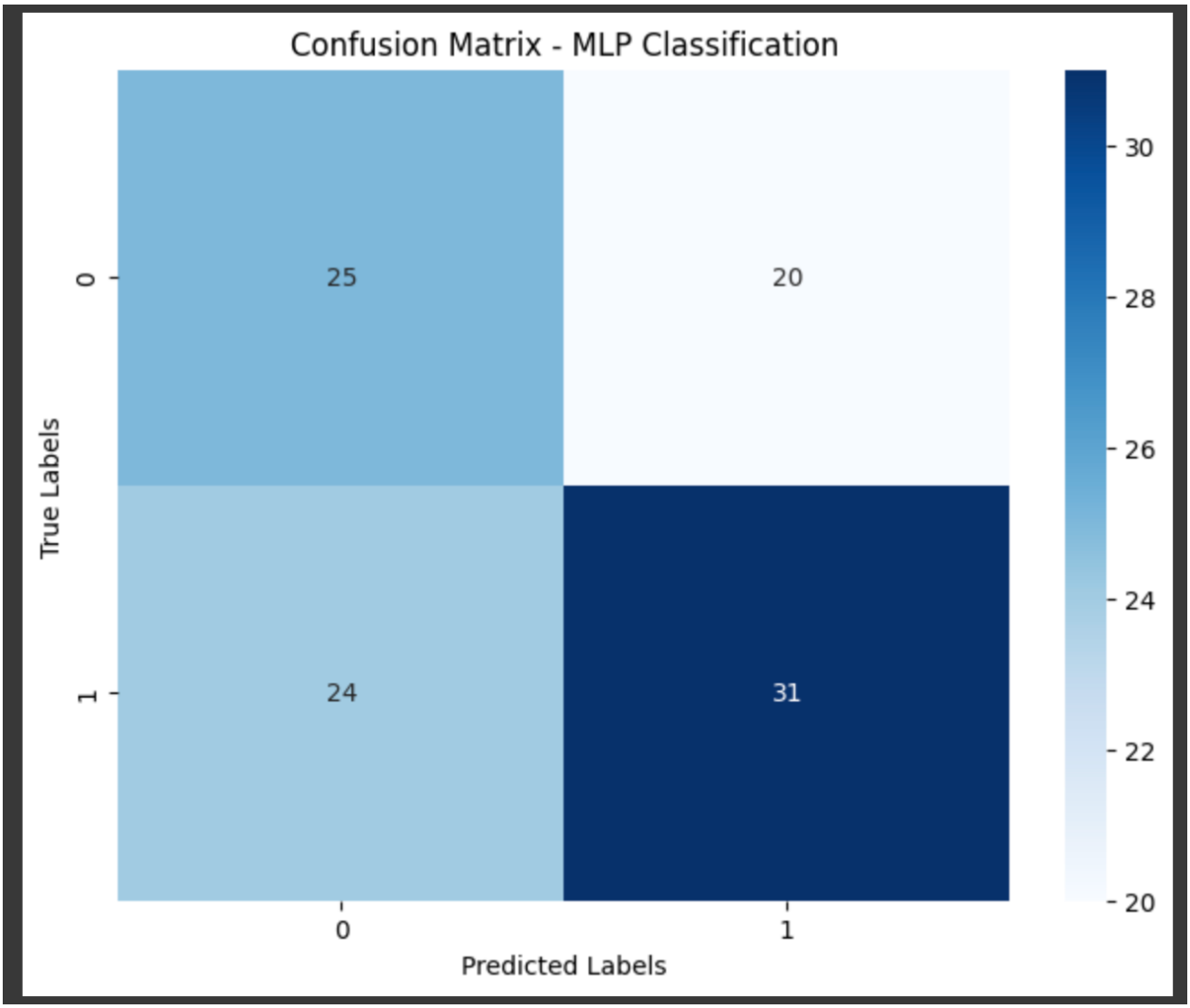
Multilayer Perceptron (MLP) Classification refers to an artificial neural network with layers of nodes that consist of an input layer, one or more hidden layers, and an output layer. It employs forward propagation to calculate predictions, and back-propagation which is a mechanism for adjusting the weights during training with the aim of learning the complex patterns and relationships in the data for the classification tasks.

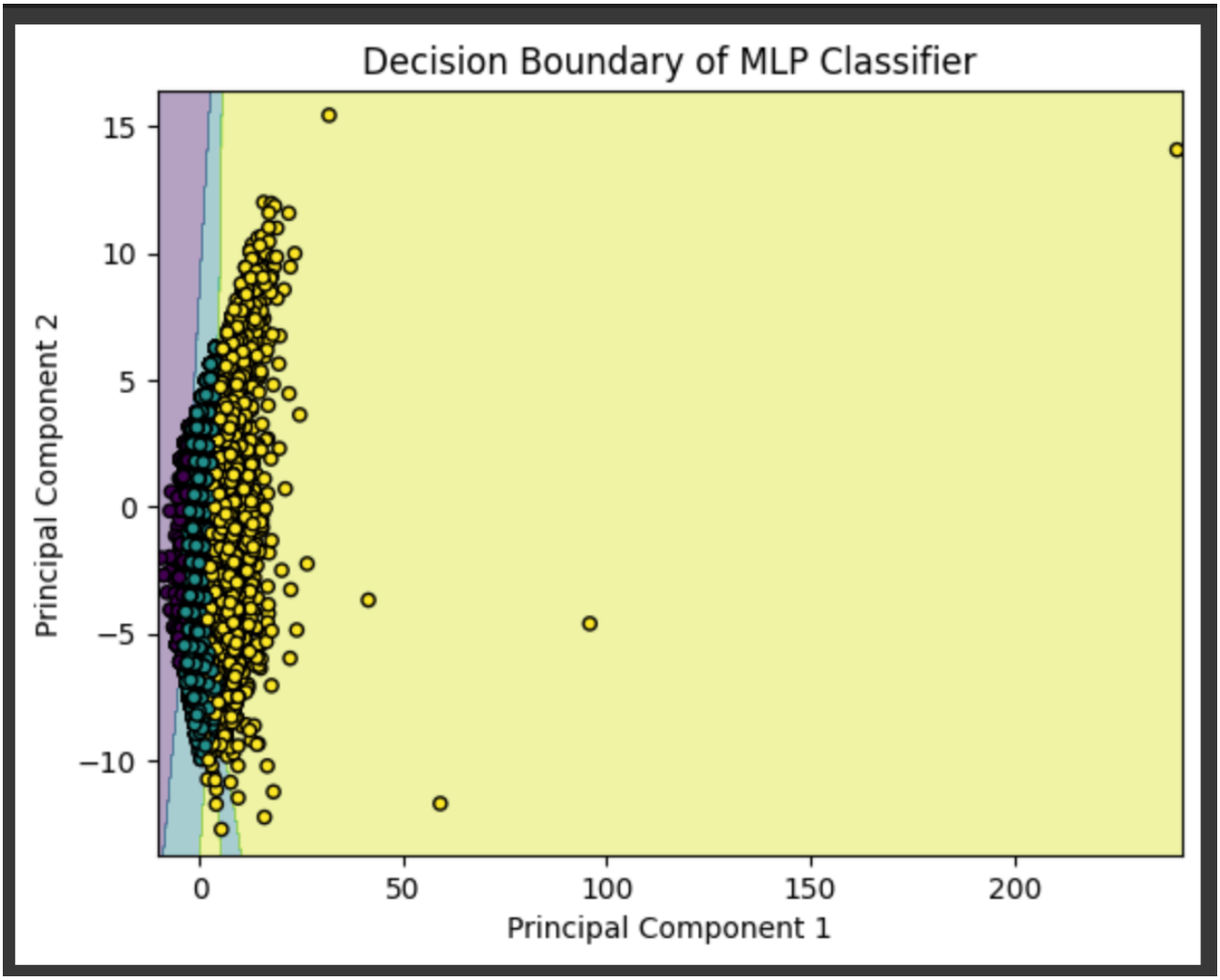
MLP - Train Accuracy: 0.9999875

MLP - Test Accuracy: 1.0



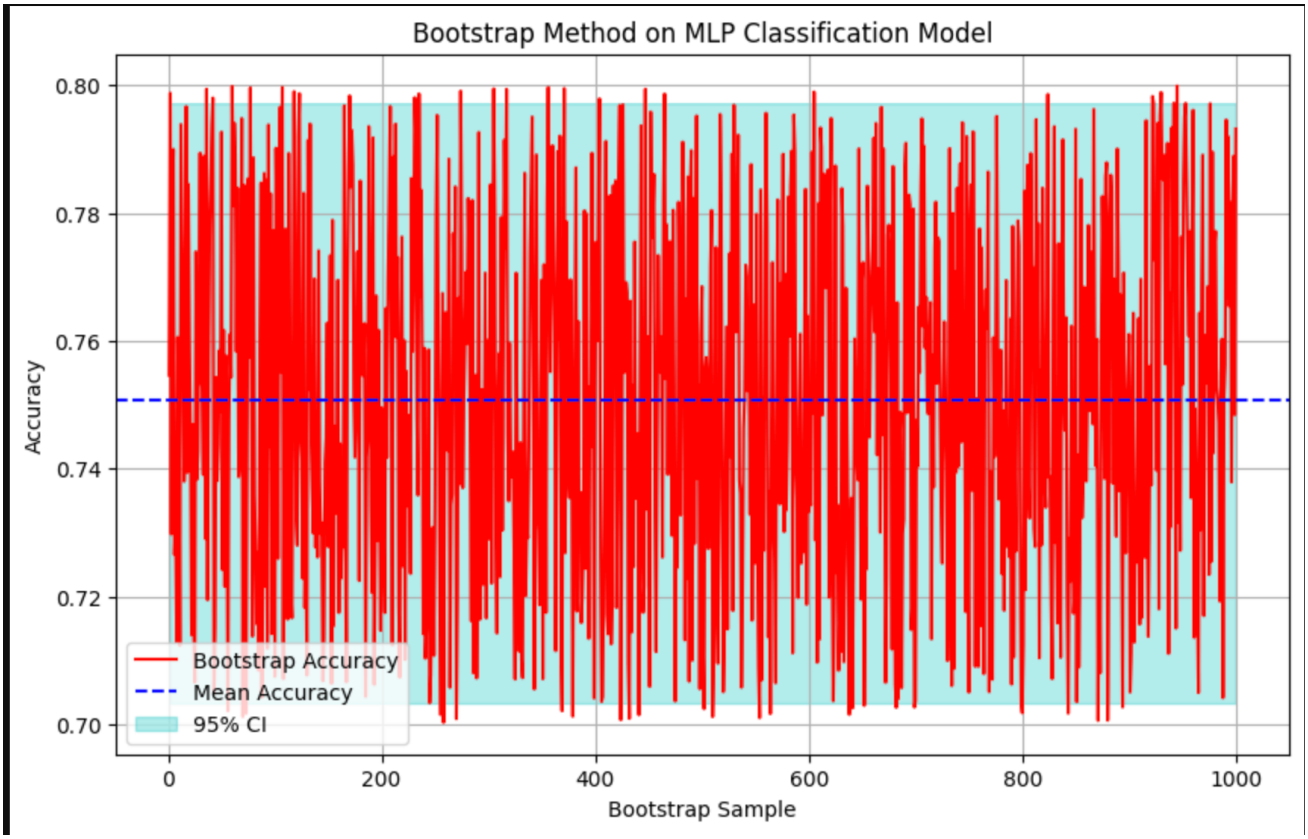
Confusion Matrix for Multilayer Perceptron:




Decision Boundary of MLP Classifier
Principal Component 2
Principal Component 1
15
10
5
0
−5
−10
0
50
100
150
200

Bootstrapping:

A bootstrapping MLP that uses classification involves feeling several bootstrap samples from a dataset, training each MLP classifier individually on each sample. The classifiers are then assembled using techniques such as bagging and boosting to combine them, thereby curbing overfitting and the extra variation that the model might face.



Mean Accuracy: 0.7507810908756811

Standard Deviation of Accuracy: 0.028223496630109677

95% Confidence Interval: [0.70329431 0.79720171]

- **CNN Classification:**

CNN classification, which is a deep learning architecture, often referred to as Convolutional Neural Network has been generally employed to solve image classification problems. It combines convolutional layers to extract features in the hierarchical mode, with the subsequent pooling layers used for dimensionality reduction, and the end with the fully connected layers which classify. Such architectures excel in detecting patterns from images, hence forming a basis for most computer-vision tasks.

Epoch 1/20

2500/2500 [==============================] - 8s 3ms/step - loss: 0.5964 - accuracy: 0.7874 - val_loss: 0.4306 - val_accuracy: 0.8636

Epoch 2/20

2500/2500 [==============================] - 7s 3ms/step - loss: 0.4121 - accuracy: 0.8606 - val_loss: 0.3908 - val_accuracy: 0.8647

Epoch 3/20

2500/2500 [==============================] - 7s 3ms/step - loss: 0.3915 - accuracy: 0.8616 - val_loss: 0.3818 - val_accuracy: 0.8648

Epoch 4/20

2500/2500 [==============================] - 6s 2ms/step - loss: 0.3855 - accuracy: 0.8623 - val_loss: 0.3789 - val_accuracy: 0.8649

Epoch 5/20

2500/2500 [==========] - 6s 3ms/step - loss: 0.3823 - accuracy: 0.8626 - val_loss: 0.3794 - val_accuracy: 0.8640

Epoch 6/20

2500/2500 [==========] - 6s 2ms/step - loss: 0.2382 - accuracy: 0.9182 - val_loss: 0.0875 - val_accuracy: 0.9989

Epoch 7/20

2500/2500 [==========] - 6s 3ms/step - loss: 0.0603 - accuracy: 0.9991 - val_loss: 0.0390 - val_accuracy: 0.9999

Epoch 8/20

2500/2500 [==========] - 6s 2ms/step - loss: 0.0285 - accuracy: 0.9998 - val_loss: 0.0192 - val_accuracy: 0.9999

Epoch 9/20

2500/2500 [==========] - 7s 3ms/step - loss: 0.0147 - accuracy: 0.9998 - val_loss: 0.0199 - val_accuracy: 0.9938

Epoch 10/20

2500/2500 [==========] - 5s 2ms/step - loss: 0.0075 - accuracy: 0.9999 - val_loss: 0.0050 - val_accuracy: 1.0000

Epoch 11/20

2500/2500 [==========] - 7s 3ms/step - loss: 0.0052 - accuracy: 0.9995 - val_loss: 0.0031 - val_accuracy: 0.9999

Epoch 12/20

2500/2500 [==========] - 6s 2ms/step - loss: 0.0031 - accuracy: 0.9998 - val_loss: 0.0020 - val_accuracy: 0.9999

Epoch 13/20

2500/2500 [==========] - 7s 3ms/step - loss: 0.0023 - accuracy: 0.9997 - val_loss: 0.0019 - val_accuracy: 0.9998

Epoch 14/20

2500/2500 [==========] - 5s 2ms/step - loss: 0.0017 - accuracy: 0.9998 - val_loss: 8.8231e-04 - val_accuracy: 1.0000

Epoch 15/20

2500/2500 [==========] - 7s 3ms/step - loss: 0.0011 - accuracy: 0.9998 - val_loss: 5.1309e-04 - val_accuracy: 1.0000

Epoch 16/20

2500/2500 [==========] - 6s 2ms/step - loss: 0.0010 - accuracy: 0.9998 - val_loss: 4.1528e-04 - val_accuracy: 1.0000

Epoch 17/20

2500/2500 [==========] - 6s 3ms/step - loss: 5.9599e-04 - accuracy: 0.9999 - val_loss: 0.0038 - val_accuracy: 0.9990

Epoch 18/20

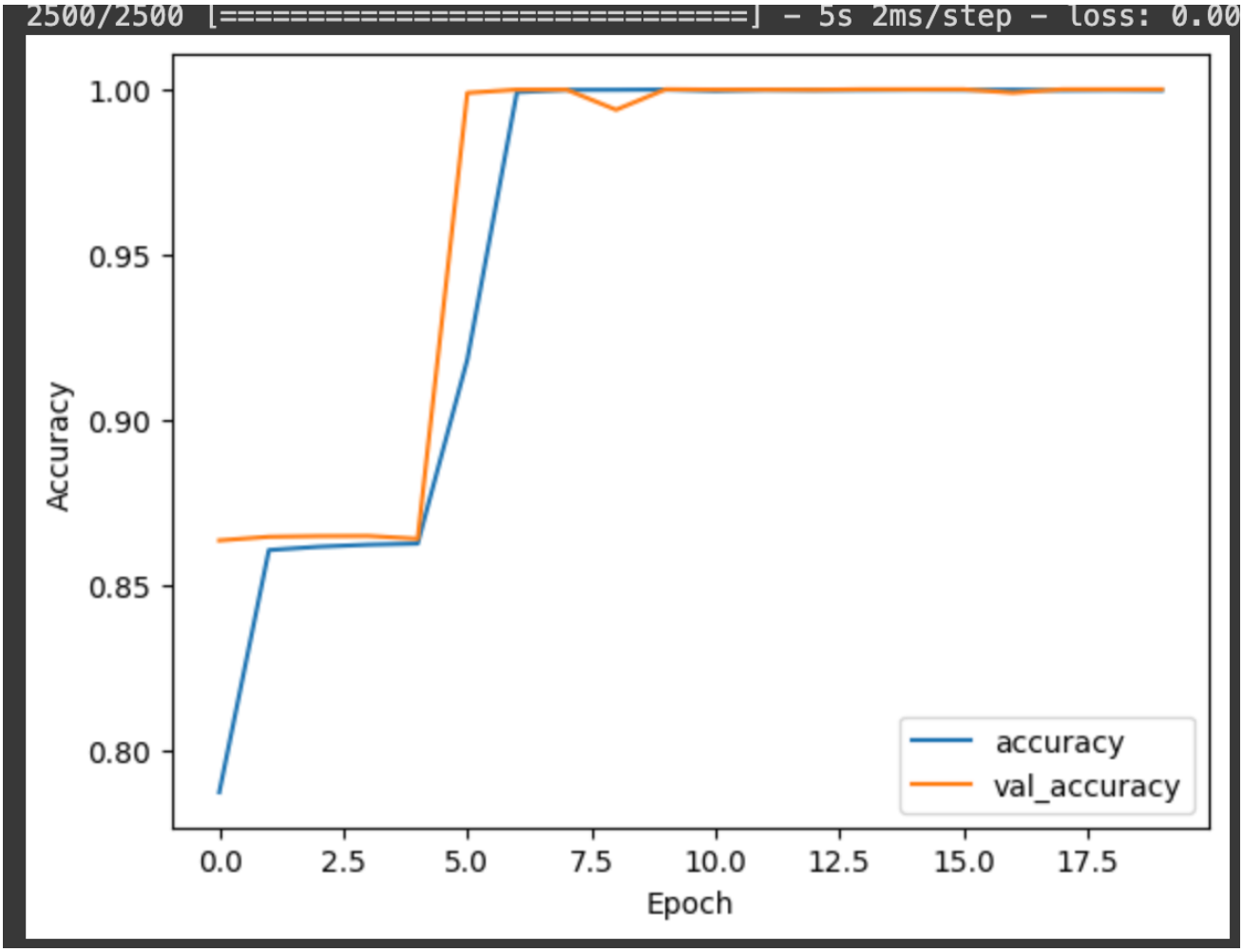
2500/2500 [==========] - 5s 2ms/step - loss: 0.0013 - accuracy: 0.9997 - val_loss: 2.6966e-04 - val_accuracy: 1.0000

Epoch 19/20

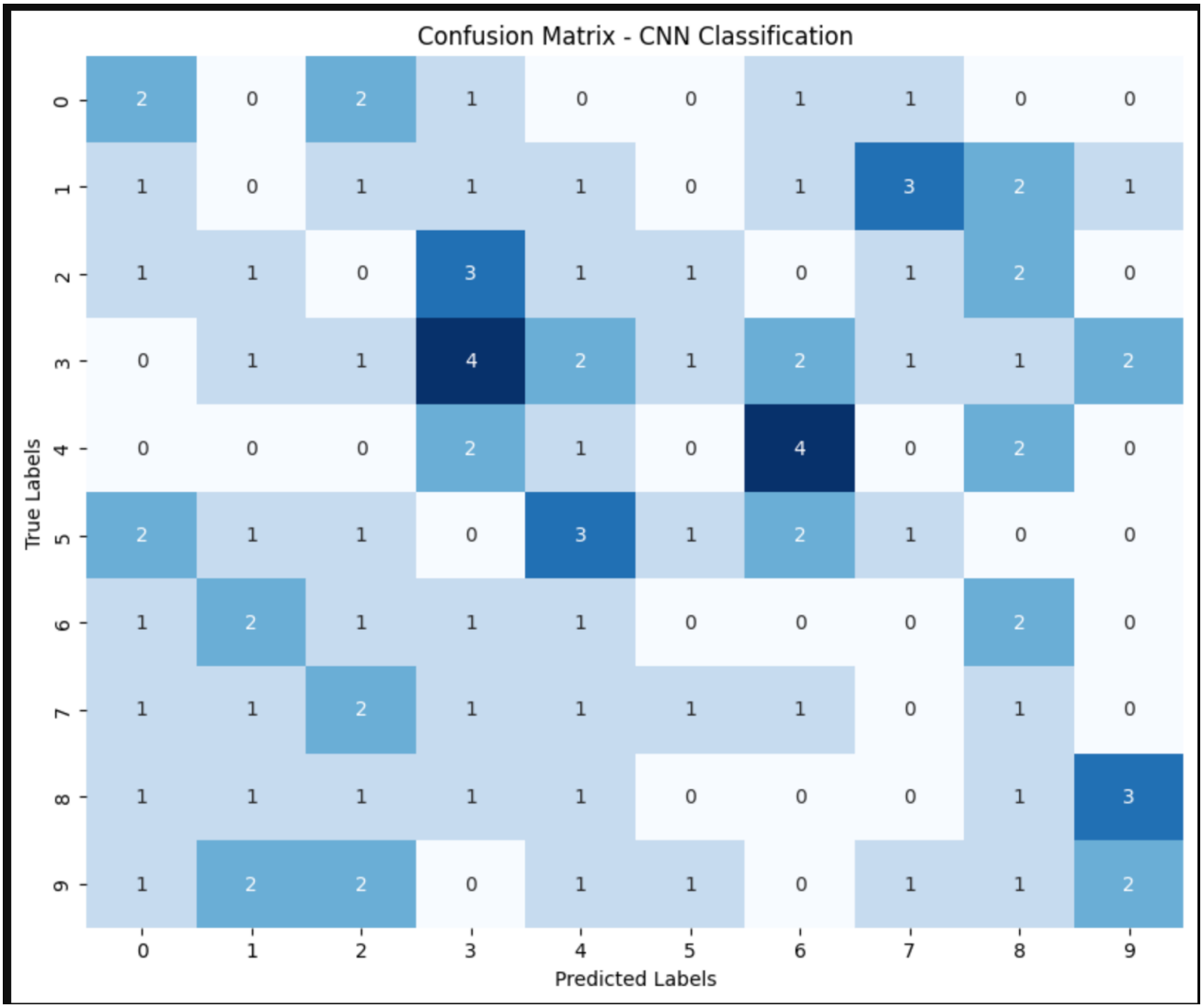
2500/2500 [==========] - 8s 3ms/step - loss: 8.6701e-04 - accuracy: 0.9997 - val_loss: 2.1054e-04 - val_accuracy: 1.0000

Epoch 20/20

2500/2500 [==========] - 5s 2ms/step - loss: 0.0011 - accuracy: 0.9997 - val_loss: 3.2599e-04 - val_accuracy: 1.0000

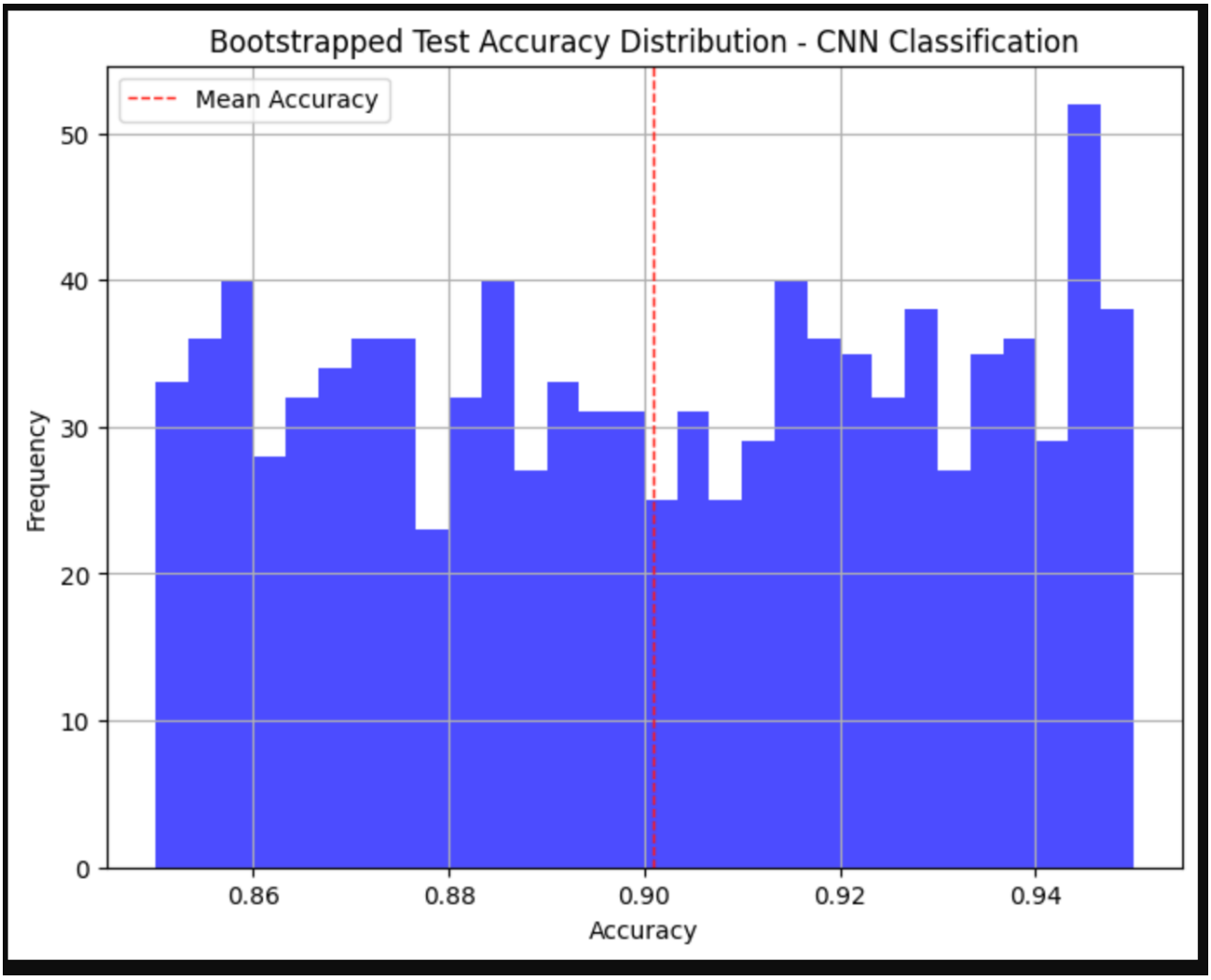

2500/2500 [==========] - 5s 2ms/step - loss: 0.00
1.00
0.95
0.90
0.85
0.80
Accuracy
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Epoch
accuracy
val_accuracy

Confusion Matrix for CNN classification:



Bootstrapping:

The CNN classification covers data bootstrapping by producing various bootstrap samples from the dataset, after which individual CNN classifiers are trained on each sample. These classifiers are aggregated, usually with the help of the techniques like bagging and boosting, to raise the quality and robustness of the model thus reducing the overfitting and variance while making the model even more efficient in the image classification tasks.



Mean Accuracy (Bootstrapped): 0.9009201698623106

95% Confidence Interval: [0.85274067 0.94761938]

Conclusion

Finally, the Password Strength Predictor Project would be a groundbreaking tool for digital security work. It offers a healthy means to check password strength with the help of machine learning algorithms and as such protects computers against intruders that ultimately improves cybersecurity. Leading through user-centricity, practical guidance, and continuous improvement, the program not only fulfills the immediate challenge but also establishes a long-term password security culture and adaptive resistance to the future cyber threats.