DEPARTMENT OF COMPUTER & INFORMATION SYSTEMS ENGINEERING
BACHELORS IN COMPUTER SYSTEMS ENGINEERING
Course Code: CS-324
Course Title: Machine Learning
Complex Engineering Problem
TE Batch 2020, Spring Semester 2023
Grading Rubric
TERM PROJECT

Group Members:

Student No.	Name	Roll No.
S1	**Abdul Raffay**	**CS-20094**
S2	**Hunzala Mushtaq**	**CS-20052**
S3	**Hamza Ahad Akhtar**	**CS-200101**

CRITERIA AND SCALES				**Marks Obtained**		
				S1	**S2**	**S3**
Criterion 1: Does the application meet the desired specifications and produce the desired outputs? (CPA-1, CPA-2, CPA-3) [8 marks]						
1	2	3	4			
The application does not meet the desired specifications and is producing incorrect outputs.	The application partially meets the desired specifications and is producing incorrect or partially correct outputs.	The application meets the desired specifications but is producing incorrect or partially correct outputs.	The application meets all the desired specifications and is producing correct outputs.			
Criterion 2: How well is the code organization? **[2 marks]**						
1	2	3	4			
The code is poorly organized and very difficult to read.	The code is readable only to someone who knows what it is supposed to be doing.	Some part of the code is well organized, while some part is difficult to follow.	The code is well organized and very easy to follow.			
Criterion 3: Does the report adhere to the given format and requirements? **[6 marks]**						
1	2	3	4			
The report does not contain the required information and is formatted poorly.	The report contains the required information only partially but is formatted well.	The report contains all the required information but is formatted poorly.	The report contains all the required information and completely adheres to the given format.			
Criterion 4: How does the student performed individually and as a team member? (CPA-1, CPA-2, CPA-3) **[4 marks]**						
1	2	3	4			
The student did not work on the assigned task.	**The student worked on the assigned task, and accomplished goals partially.**	**The student worked on the assigned task, and accomplished goals satisfactorily.**	**The student worked on the assigned task, and accomplished goals beyond expectations.**			

Final Score = (Criterial_1_score x 2) + (Criteria_2_score / 2) + (Criteria_3_score x (3/2)) + (Criteria_4_score)

= _____

PROBLEM STATEMENT:

Complex problem-solving attributes (CPA) covered (as per PEC - OBA manual – 2019)

- **CPA-1 Depth of analysis required:** Have no obvious solution and require abstract thinking, originality in analysis to formulate suitable models.
- **CPA-2 Level of interaction:** Require resolution of significant problems arising from interactions between wide ranging or conflicting technical, engineering or other issues.
- **CPA-3 Familiarity:** Can extend beyond previous experiences by applying principles-based approaches.

Description

You are required to develop a machine learning system to classify the output as indicated below for any one of the following datasets.

Dataset title	Type of data	Label to be predicted	Download link
Weather Dataset	Structured tables	Multiclass classification – Predict the 'Summary' attribute. You can reduce the number of classes to up to 3 for better performance.	https://www.kaggle.com/datasets/muthuj7/weather-dataset
CIFAR-10	Images	Binary classification – Create a new label by relabeling the outputs into two classes, namely 'animal' and 'vehicle'.	https://www.cs.toronto.edu/~kriz/cifar.html

A BRIEF OVERVIEW OF THE PROJECT:

In this project we have done weather forecasting by using various machine learning algorithms, for the training of all the models we have picked up an open source wheather dataset from kaggle.com The w
hole project is being done in two steps:

1. Data Pre-processing and Feature Engineering.
2. Model Training

1. DATA PRE-PROCESSING AND FEATURE ENGINEERING:

In the implementation of any machine learning system, data pre-processing and feature engineering is the first and the most crucial step as the more ou data is cleaned, free from noise and outliers the more accurate the model will be able to generalize the results. We have chosen the open source **weather dataset** which is available on kaggle.

Having the above facts in mind we have cleaned our data and pre processed it in five steps which are:

a. aMissing Values Treatment.
b. Outlier Treatment.
c. Redundant Columns Removal / Handling multicollinearity.
d. Data Encoding and Data Split.
e. Data Scaling (Standardization).

a. Missing Values Treatment:

In this step we have checked for null values in all the 12 columns of the dataset and it came out that in one of the input features columns "Precip Type" there are 517 null values, so have dropped all those rows which have null value in that column. We have not

taken the mean of all the other values in the “Precip Type” as there were total 517 null values which is very less than the total size of the dataset which is 95k.

STEP 1: Missing Values Treatment

```
def check_null_values(df):
    for column in df.columns.to_list():
        count = 0
        for isNullValue in df[column].isna().tolist():
            if isNullValue:
                count += 1
        if (count > 0):
            print(f'There are {count} null values in column {column}')
```

```
df = pd.read_csv('../data/weather.csv')
check_null_values(df)
```

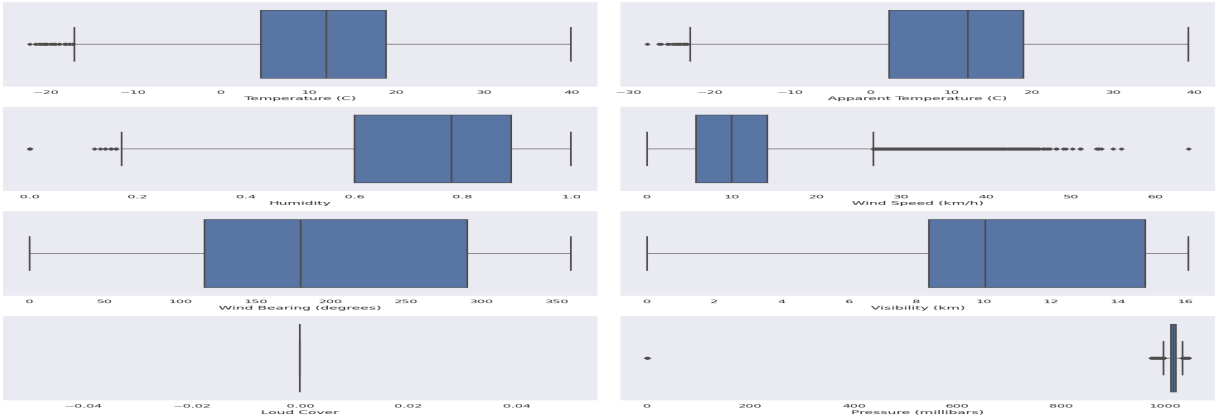
```
There are 517 null values in column Precip Type
```

Hence dropping all those 517 rows which has null values in their **Precip Type** column

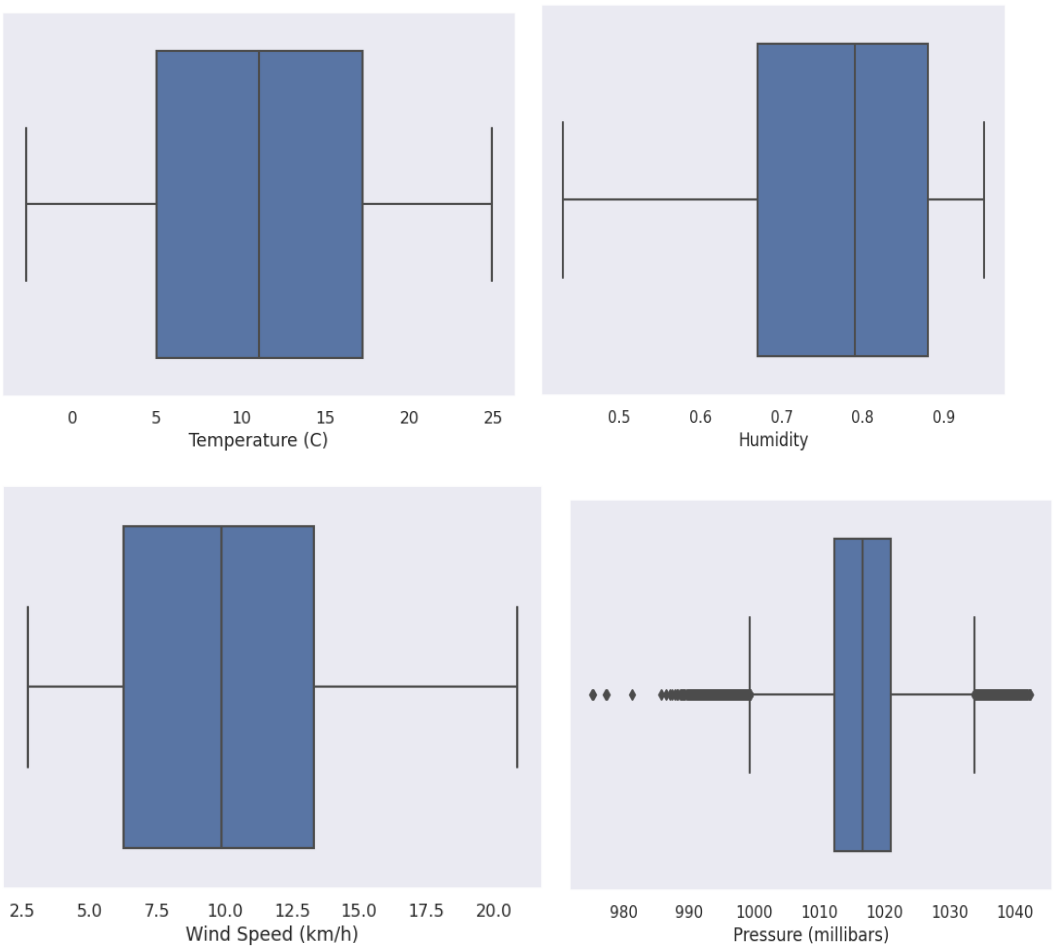
```
df_cleaned = df.dropna(subset=['Precip Type'])
check_null_values(df_cleaned)
```

b. Outlier Treatment:

In this step we have checked if there exists outliers in any of the 12 columns of the dataset by plotting box plot for each column, and as it can be seen from the plot below there are some columns which has outliers so we have removed outliers by using the percentile technique.



And after removing all the outliers we have again plotted the box plot to verify that the outliers are removed from the dataset:

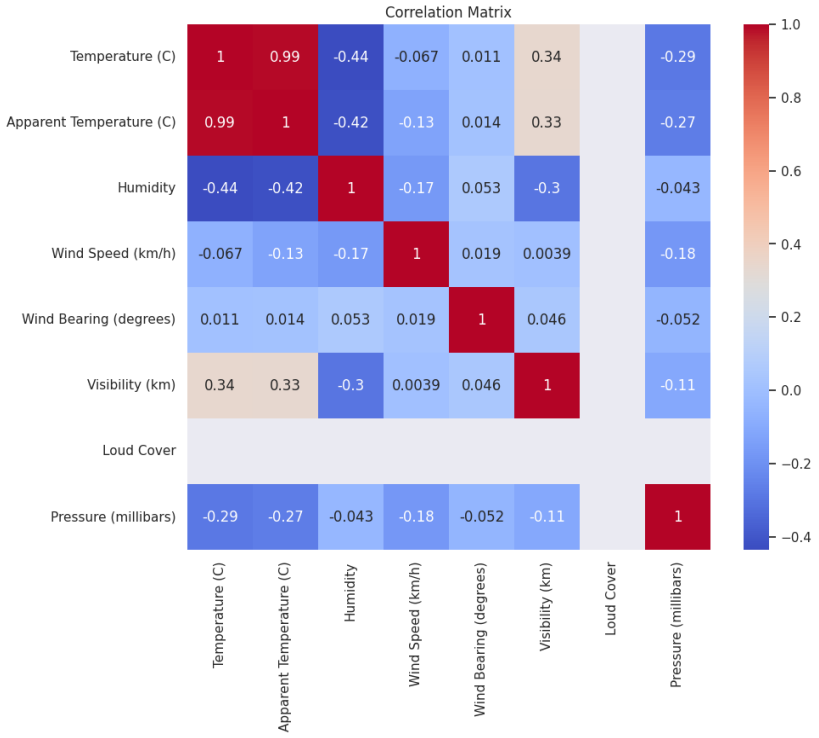


c. Redundant Columns Removal / Handling Multicollinearity:

In this step we have plot a correlation matrix to check which columns are highly correlated with each other, and from the correlation matrix it is cleared that there are two columns which are highly correlated with each other which are these:

1. Temperature (C)
2. Apparent Temperature (C)

So it shows that out of these two columns one column is redundant so in order to remove this redundancy we have removed the column Apparent Temperature (C) so that by removing the redundant columns we can reduce the features and our model can trained more quickly.



d. Data Encoding and Data Split:

In this step we first have picked up three classes which are **"Overcast", "Clear" and "Foggy"** out of total 11 classes from the output **"Summary"** column so that out model can be trained quickly.

We have also break down one column which is **"Formatted Date"** into four parts i.e **"Hour", "Day", "Year" and "Week".** The **"Precip Type"** column is being encoded by using **"binary or label encoder"**

e. Data Scaling (Standardization):

In this step we have applied standardization, to the input features so that all of the input features column values lies in the same range.

2. MODEL BUILDING:

We have trained mainly three types of models which includes parametric models, non-parametric and a neural network architecture. In the parametric models we have trained **Logistic Regression, Gaussian Naive Bayes Classifier, Stochastic Gradient Descent.** In non-parametric models we have trained **Decision Tree, K Nearest Neighbours Classifier, Random Forest Classifier.** In neural network architecture we have implemented three variants of it namely: **NEURAL**

NETWORK 1: 16-256 ANN with 'tanh', NEURAL NETWORK 2: 32-64-512 ANN with 'relu', NEURAL NETWORK 1: 32-1024 ANN with 'sigmoid'

a . Logistic Regression:

Logistic regression is a statistical model used for binary classification, predicting a categorical outcome variable with two possible values. It employs a logistic function to model the relationship between independent variables and the probability of belonging to a particular class. The process involves selecting relevant features, constructing the model by estimating coefficients, applying the logistic function to convert logits into probabilities, training the model through parameter estimation, evaluating its performance, and making predictions on new data. Logistic regression is widely used when the relationship between features and the outcome follows a linear pattern and finds applications in diverse fields like medicine, finance, marketing, and social sciences.

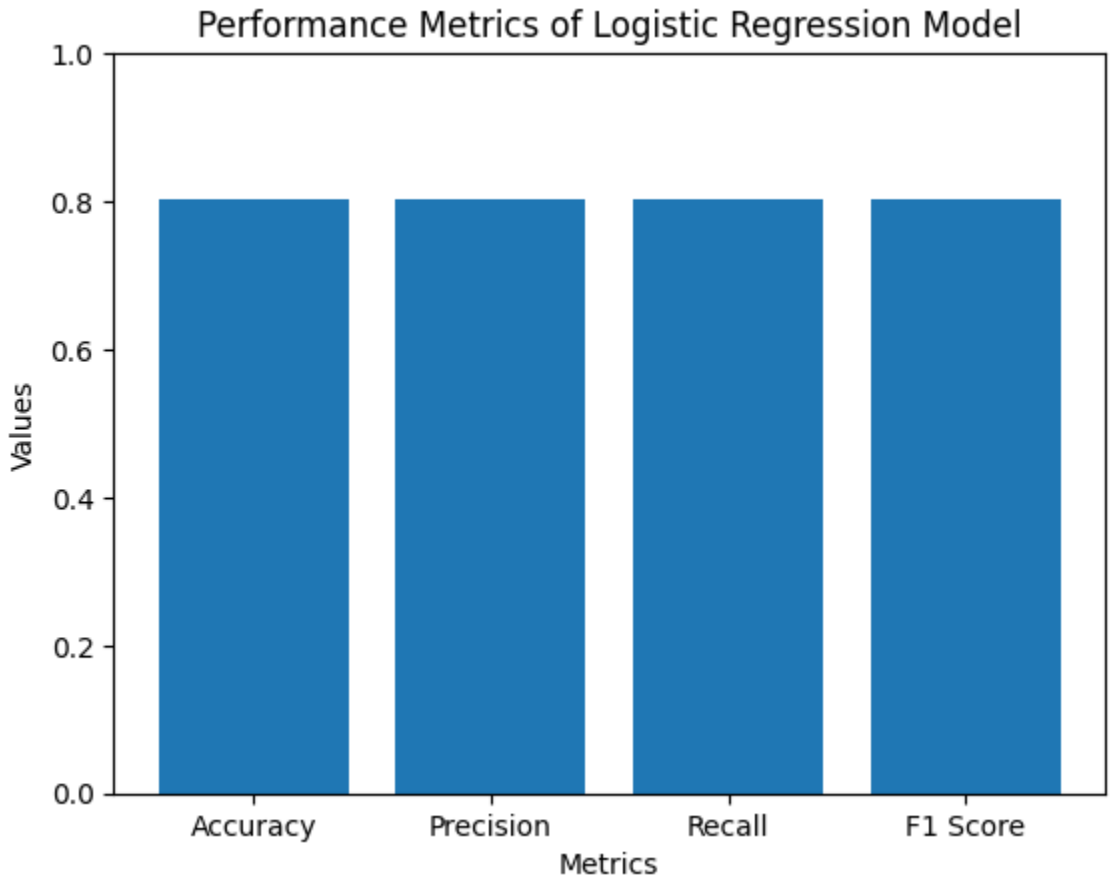
Here are the stats of the performance metrics of logistic regression:

Accuracy: 0.804

Precision: 0.802

Recall: 0.804

F1 Score: 0.803



b . Gaussian Naive Bayes Classifier:

The Gaussian Naive Bayes Classifier is a probabilistic machine learning model used for classification tasks. It assumes that the features are independent of each other and follows a Gaussian (normal) distribution. The classifier calculates the probability of an instance belonging to a particular class by applying Bayes' theorem. The working of Gaussian Naive Bayes involves estimating the mean and standard deviation of each feature for each class, based on the training data. When predicting the class of a new instance, it calculates the probability of the instance belonging to each class using the Gaussian probability density function. The class with the highest probability is then assigned as the predicted class. Gaussian Naive Bayes is computationally efficient and performs well when the independence assumption holds. It is commonly used in text classification, spam filtering, and sentiment analysis tasks.

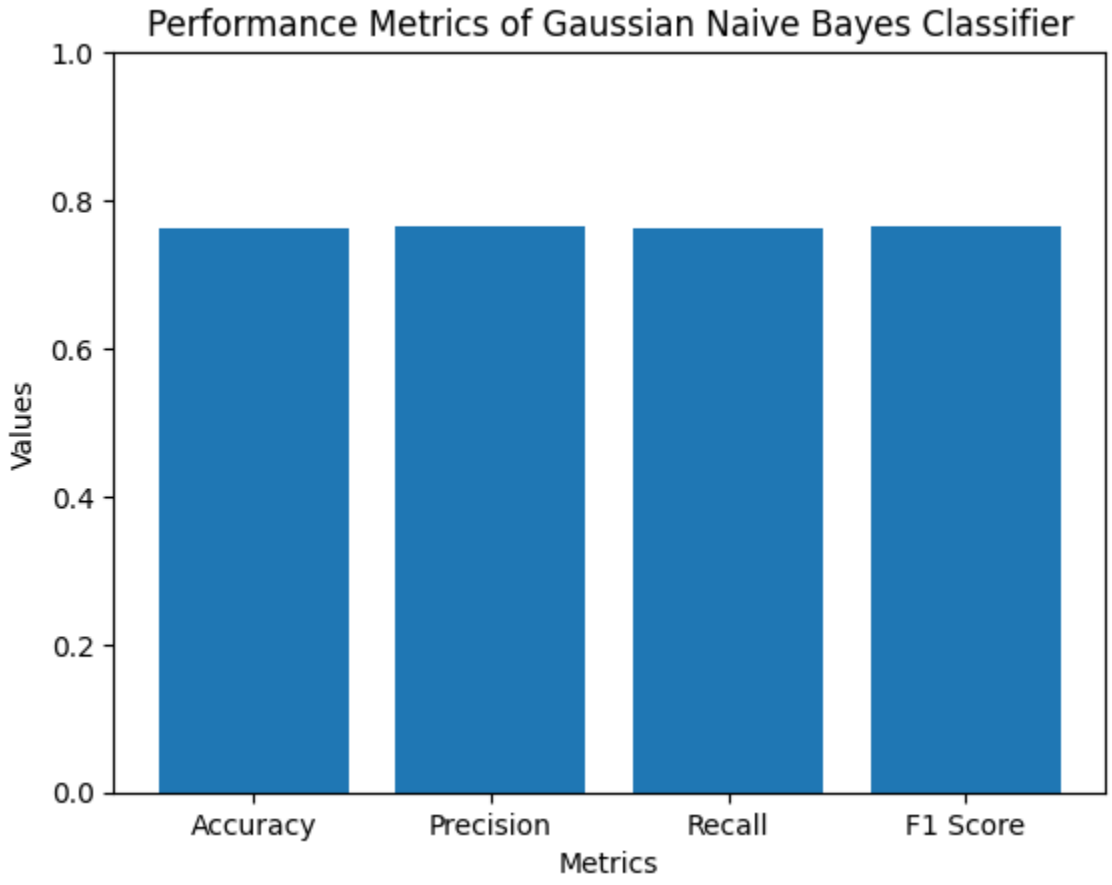
Here are the stats of the performance metrics of Gaussian Naive Bayes Classifier:

Accuracy: 0.763

Precision: 0.765

Recall: 0.763

F1 Score: 0.765



c . Stochastic Gradient Descent:

Stochastic Gradient Descent (SGD) is an iterative optimization algorithm used for training machine learning models, particularly in large-scale and online learning scenarios. It is a variant of the Gradient Descent algorithm that computes the updates for model parameters based on small random subsets, called mini-batches, of the training data instead of the entire dataset. In each iteration, SGD randomly selects a mini-batch, calculates the gradient of the loss function with respect to the mini-batch, and updates the model parameters in the direction of the negative gradient scaled by a learning rate. This process is repeated until convergence or a predetermined number of iterations. By using mini-batches, SGD reduces the computational requirements and accelerates convergence, making it suitable for handling large datasets and updating models in real-time. However, the stochastic nature of the algorithm introduces some noise in the updates, which can lead to oscillations during training but also enables it to escape local minima and explore the parameter space more broadly.

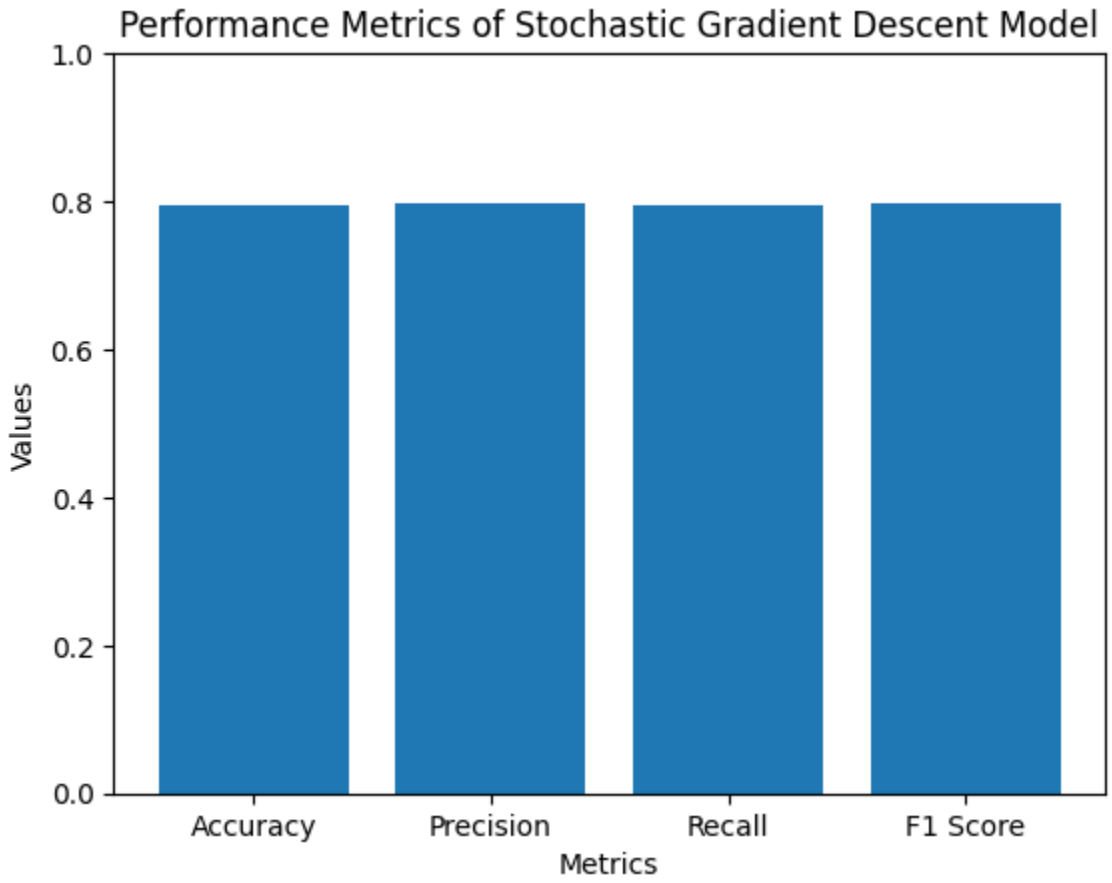
Here are the stats of the performance metrics of Stochastic Gradient Descent:

Accuracy: 0.7960

Precision: 0.798

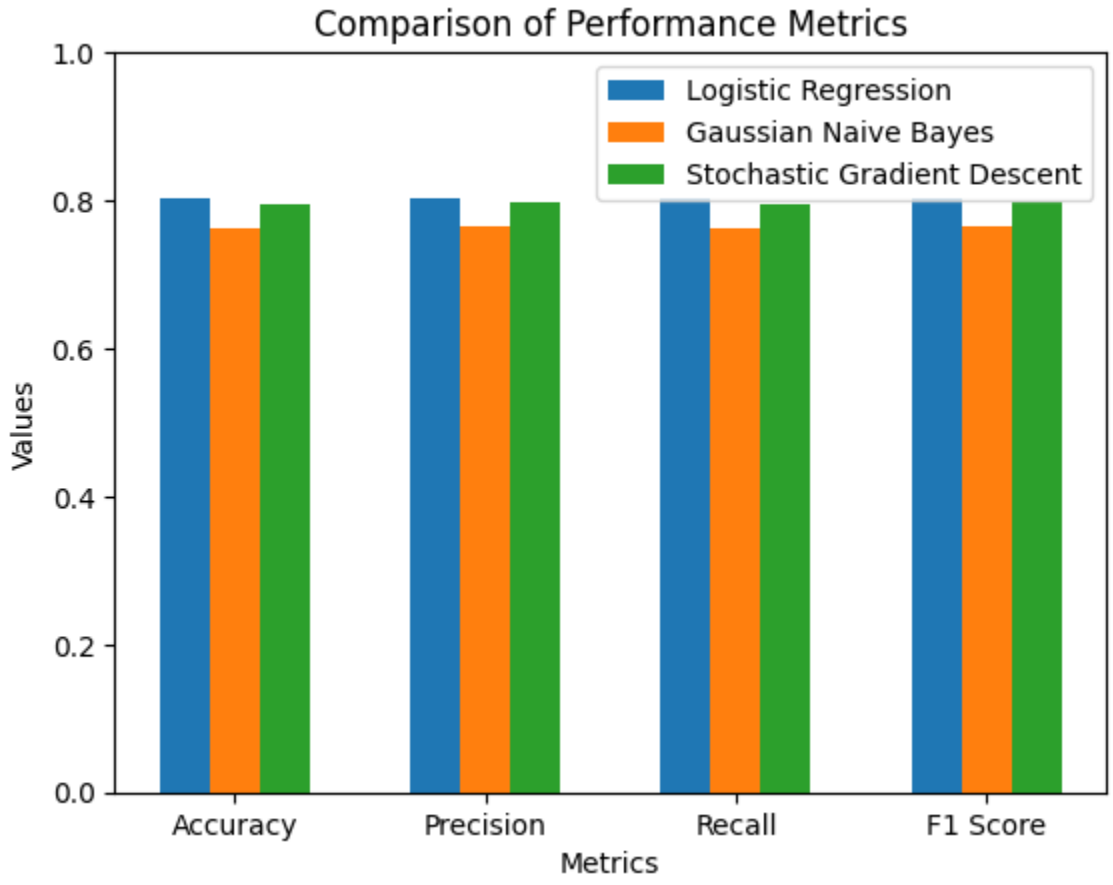
Recall: 0.796

F1 Score: 0.797



*** Graphical and Tabular Comparision of Parametric Algorithms:**

	Accuracy	Precision	Recall	F1 Score
Logisitic Regression	**0.804**	**0.802**	**0.804**	**0.803**
Guassian Naive Bayes	**0.763**	**0.765**	**0.763**	**0.763**
Stochastic Gradient Descent	**0.796**	**0.798**	**0.796**	**0.797**



Hence, the best model among the parametric models is **Logisitic Regression.**

d . Decision Tree:

A decision tree is a predictive modeling tool that utilizes a hierarchical structure consisting of nodes and branches to represent possible decisions and their outcomes. It works by recursively partitioning the data based on input features, constructing a tree where each internal node represents a feature and each leaf node represents a decision or outcome. At each internal node, the tree splits the data based on a specific feature value, aiming to maximize information gain or minimize impurity measures. This process continues until a stopping criterion is met, resulting in a tree that can be used to make predictions by traversing the branches based on the input features until reaching a leaf node, which provides the decision or outcome.

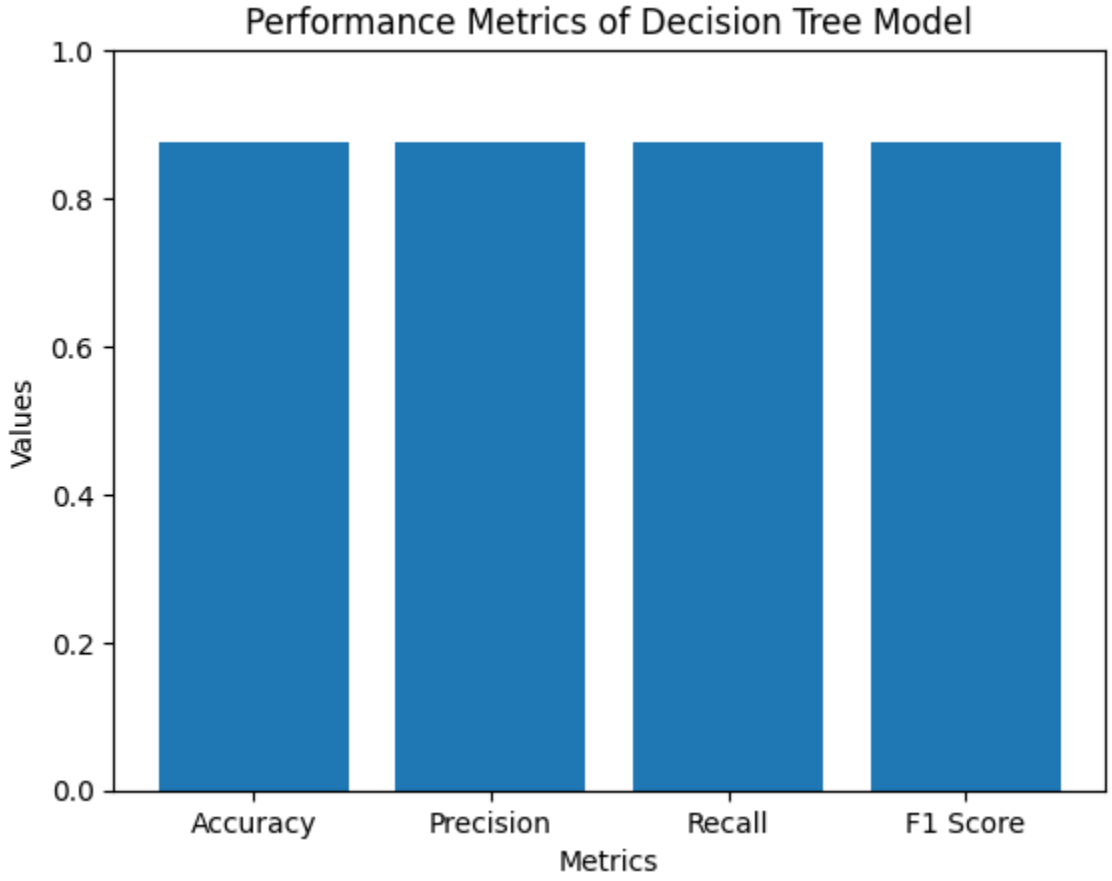
Here are the stats of the performance metrics of decision tree:

Accuracy: 0.875

Precision: 0.877

Recall: 0.875

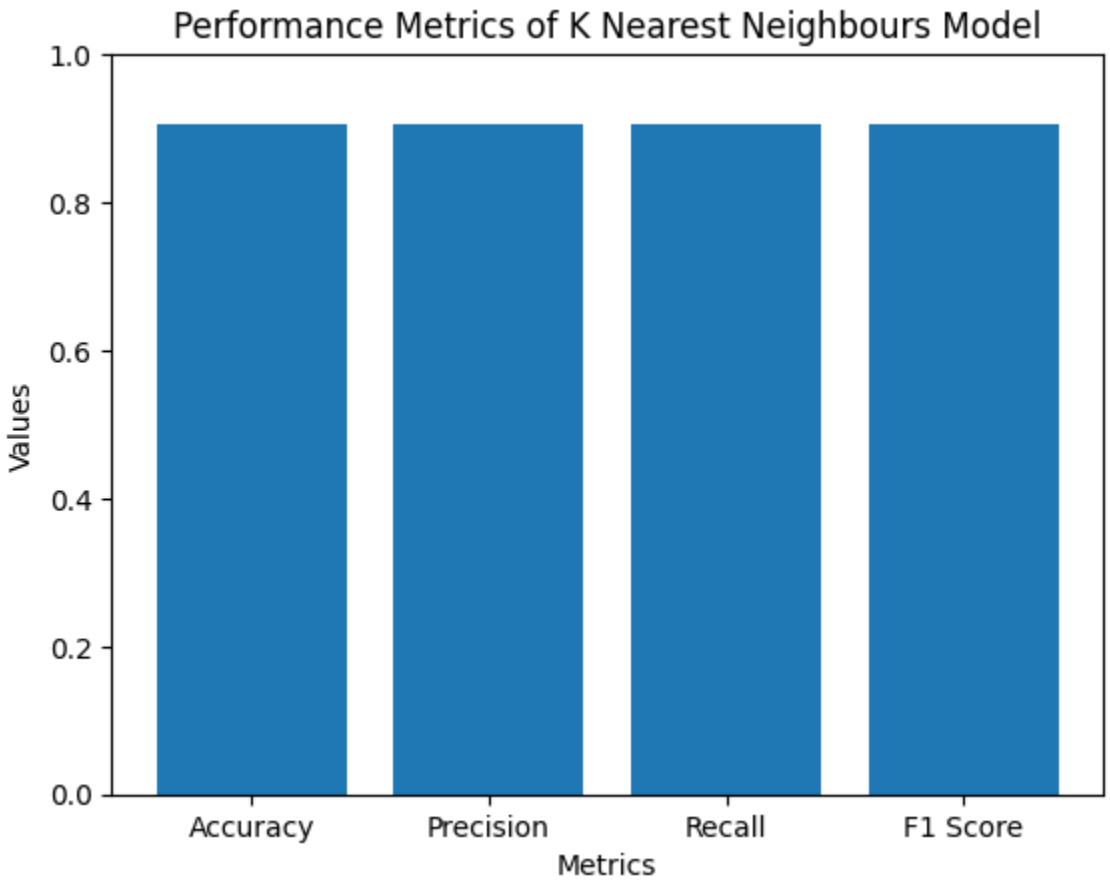
F1 Score: 0.876



e . K nearest neighbors:

K-nearest neighbors (KNN) is a non-parametric algorithm used for both classification and regression tasks in machine learning. It operates based on the principle that similar instances are likely to have similar outputs. KNN works by storing a training dataset consisting of labeled

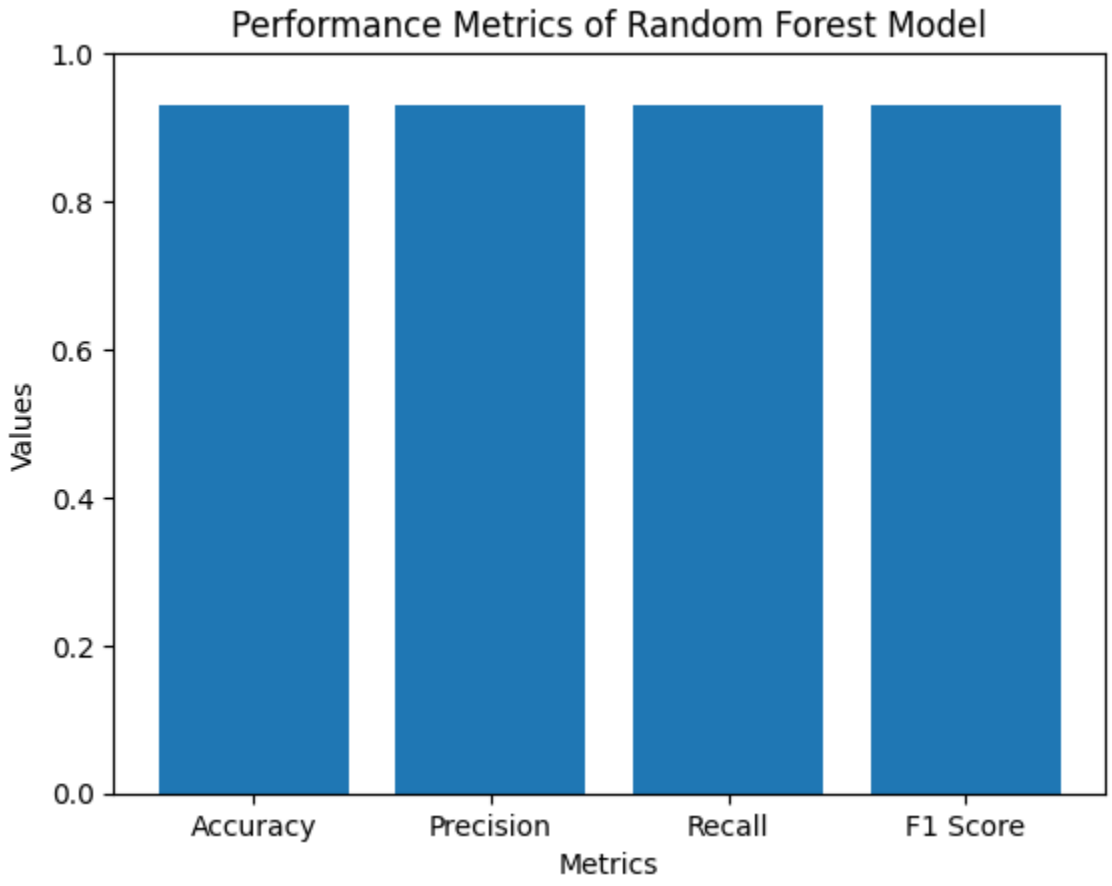
instances and, given a new input, it identifies the K nearest neighbors in the training data based on a distance metric (e.g., Euclidean distance). In the classification task, the class label of the new instance is determined by a majority vote among the K neighbors. In the regression task, the predicted value is computed as the average of the target values of the K neighbors. KNN does not make assumptions about the underlying data distribution and can handle complex relationships, but it may be sensitive to the choice of K and the distance metric used.
Here are the stats of the performance metrics of K nearest neighbor:
Accuracy: 0.906
Precision: 0.906
Recall: 0.906
F1 Score: 0.906



f . Random Forest:

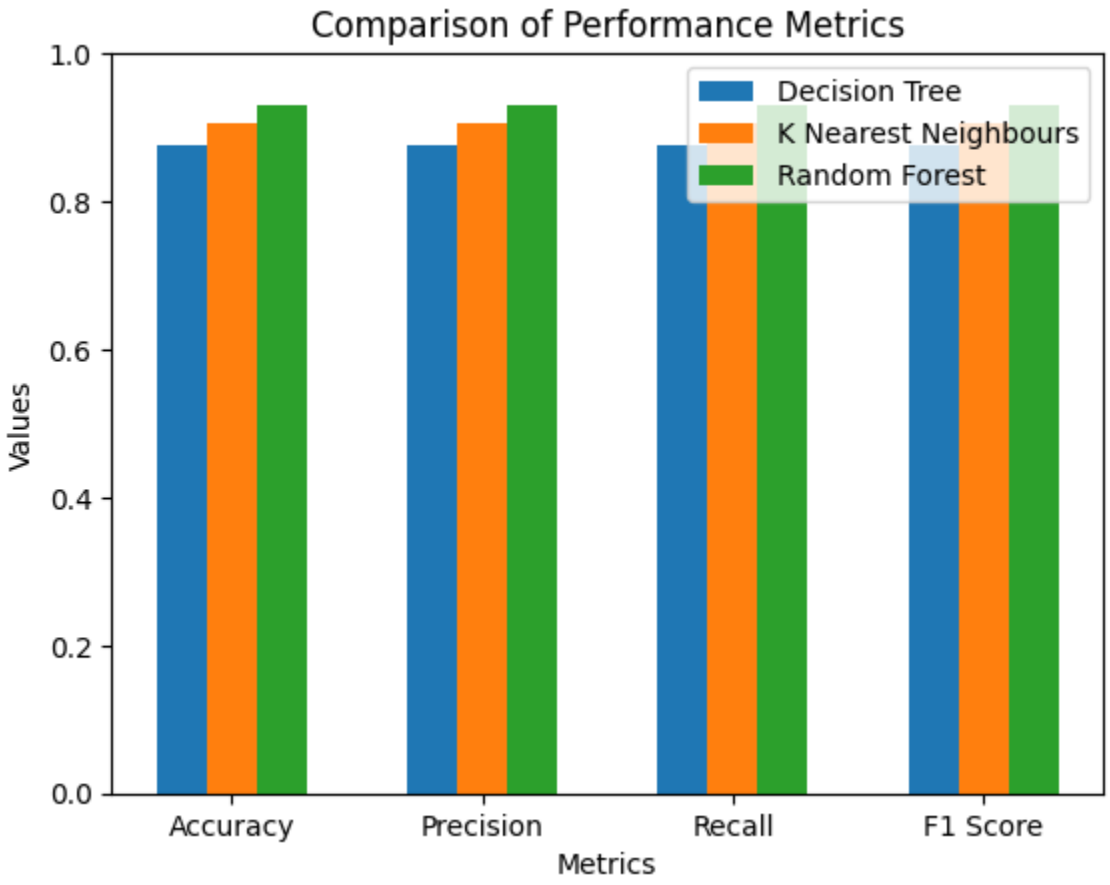
Random Forest is an ensemble learning method that combines multiple decision trees to create a robust and accurate predictive model. It works by constructing a multitude of decision trees using a random subset of features and random subsets of the training data. Each decision tree in the forest independently makes predictions, and the final prediction is determined by aggregating the

results from all the trees. During training, the decision trees are built by recursively partitioning the data based on input features, similar to the individual decision tree algorithm. However, the randomness introduced in Random Forest, such as feature subsampling and bootstrap aggregating (i.e., sampling with replacement), helps to reduce overfitting and increase generalization performance. The final prediction in Random Forest is typically determined by majority voting in classification tasks or averaging in regression tasks, where each tree's prediction carries equal weight. The strength of Random Forest lies in its ability to handle high-dimensional data, identify important features, and provide robust predictions while minimizing overfitting.
Here are the stats of the performance metrics of Random Forest:
Accuracy: 0.930
Precision: 0.930
Recall: 0.930
F1 Score: 0.930



*** Graphical and Tabular Comparision of Non-Parametric Algorithms:**

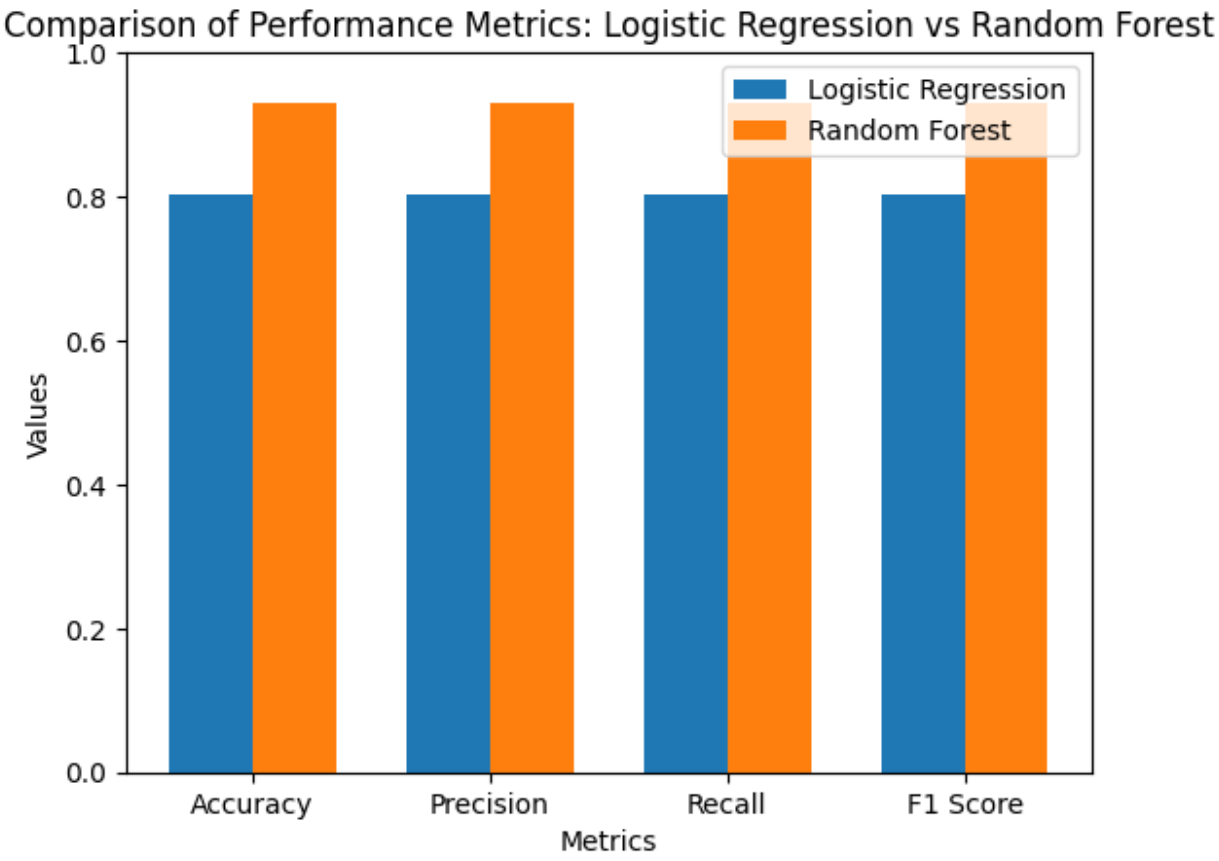
	Accuracy	Precision	Recall	F1 Score
Decision Tree	**0.875**	**0.877**	**0.875**	**0.876**
KNN	**0.906**	**0.906**	**0.906**	**0.906**
Random Forest	**0.930**	**0.930**	**0.930**	**0.930**



Hence, the best model among the parametric models is **Random Forest.**

* Comparision between the best parametric and non-parametric algorithms:

Now lets see the graphical comparison between the best parametric and non-parametric algorithms i.e logistic regression and random forest.



So the best model out of all the 6 models is **Random Forest.**

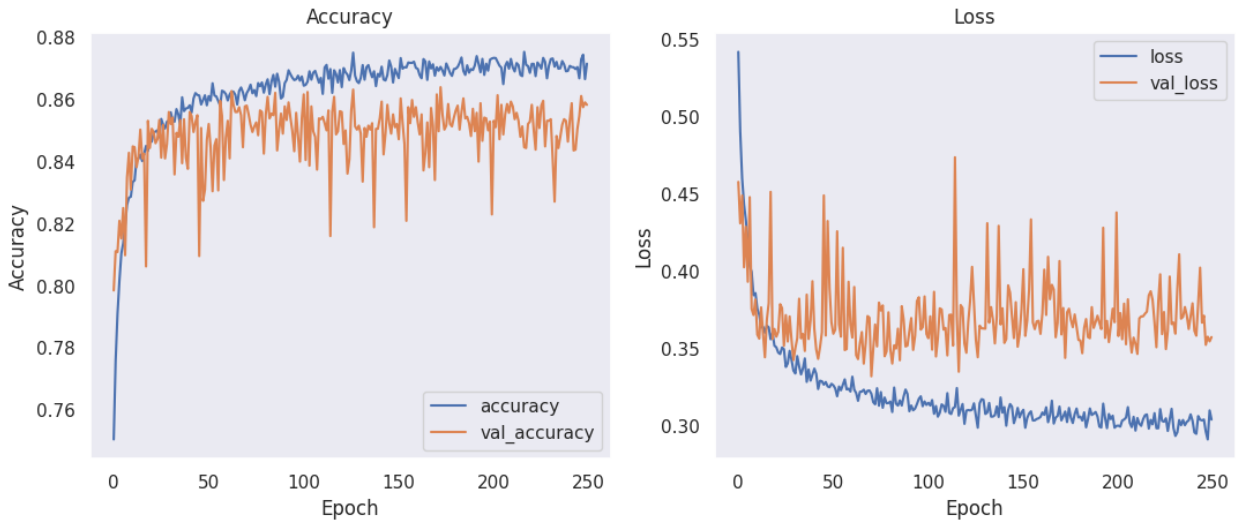
g . Neural Network 1 (16-256 ANN with ‘tanh’):

A 16-256 Artificial Neural Network (ANN) with 'tanh' activation function is a feedforward neural network architecture that consists of an input layer with 16 nodes, a single hidden layer with 256 nodes, and an output layer. The activation function used in the hidden layer and the output layer is the hyperbolic tangent (tanh) function. The input layer receives the input data, which is then propagated through the hidden layer, where each node applies the 'tanh' activation function to the weighted sum of its inputs. The activation function introduces non-linearity and allows the network to capture complex relationships in the data. The outputs of the hidden layer serve as inputs to the output layer, where another set of weights and biases are applied, and the 'tanh' activation function is again used. The final outputs from the output layer represent the predictions or classifications made by the network. During training, the network adjusts the weights and biases using techniques like backpropagation and gradient descent to minimize the difference between the predicted outputs and the expected outputs, ultimately optimizing the network's performance.

Here are the stats of the performance metrics of the Neural Network 1:

Test Loss: 0.357

Test Accuracy: 0.858



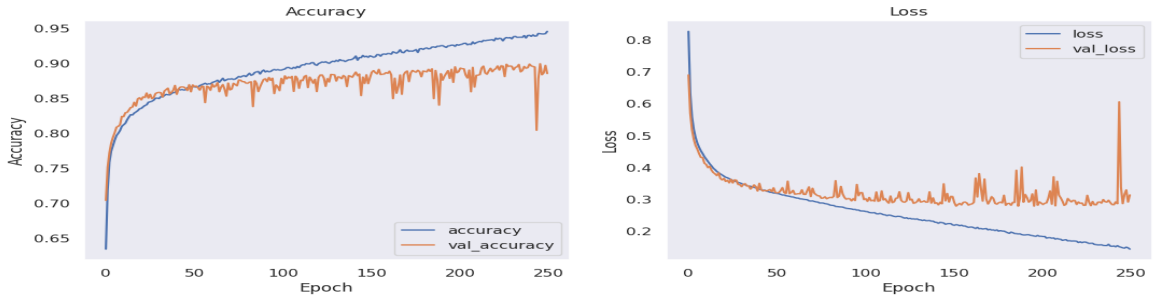
h . Neural Network 2 (32-64-512 ANN with ‘relu’):

A 32-64-512 Artificial Neural Network (ANN) with 'relu' activation function is a feedforward neural network architecture consisting of an input layer with 32 nodes, a hidden layer with 64 nodes, and another hidden layer with 512 nodes. The 'relu' (Rectified Linear Unit) activation function is used in both the hidden layers. The input layer receives the input data, which is then propagated through the first hidden layer. Each node in the hidden layer applies the 'relu' activation function to the weighted sum of its inputs, resulting in an output that is zero for negative inputs and equal to the input for positive inputs. This non-linear activation function allows the network to model complex relationships in the data. The outputs of the first hidden layer serve as inputs to the second hidden layer, which follows the same process of applying 'relu' activation to the weighted sums. Finally, the outputs from the second hidden layer are fed into the output layer, where the network makes predictions or classifications. During training, the network adjusts the weights and biases using techniques like backpropagation and gradient descent to minimize the discrepancy between the predicted outputs and the expected outputs, thereby optimizing the network's performance. The 'relu' activation function is known for its ability to alleviate the vanishing gradient problem, allowing for efficient training and better representation learning in deep neural networks.

Here are the stats of the performance metrics of the Neural Network 2:

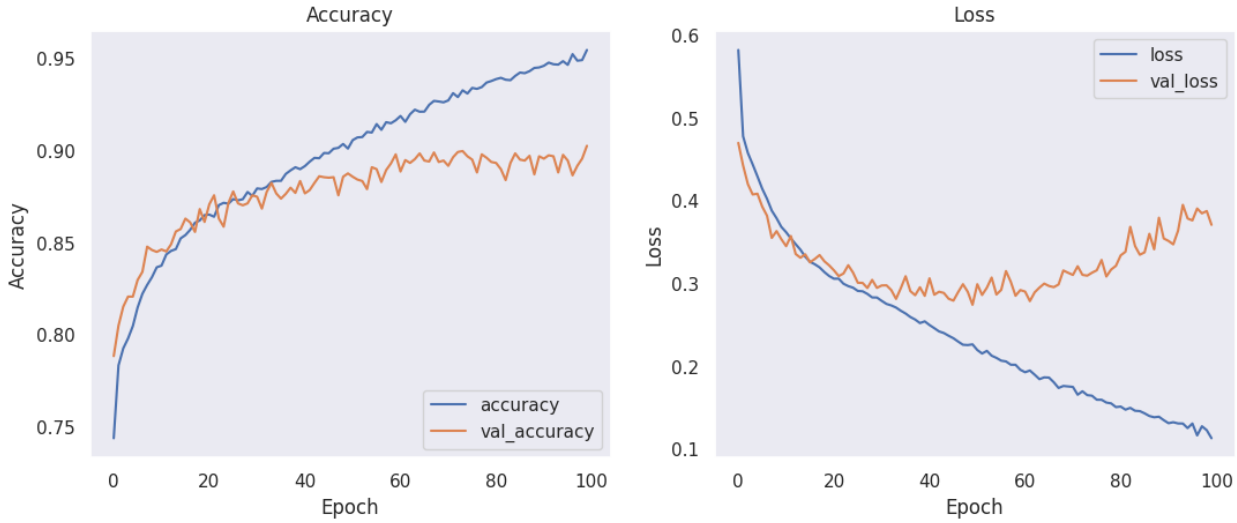
Test Loss: 0.313

Test Accuracy: 0.884



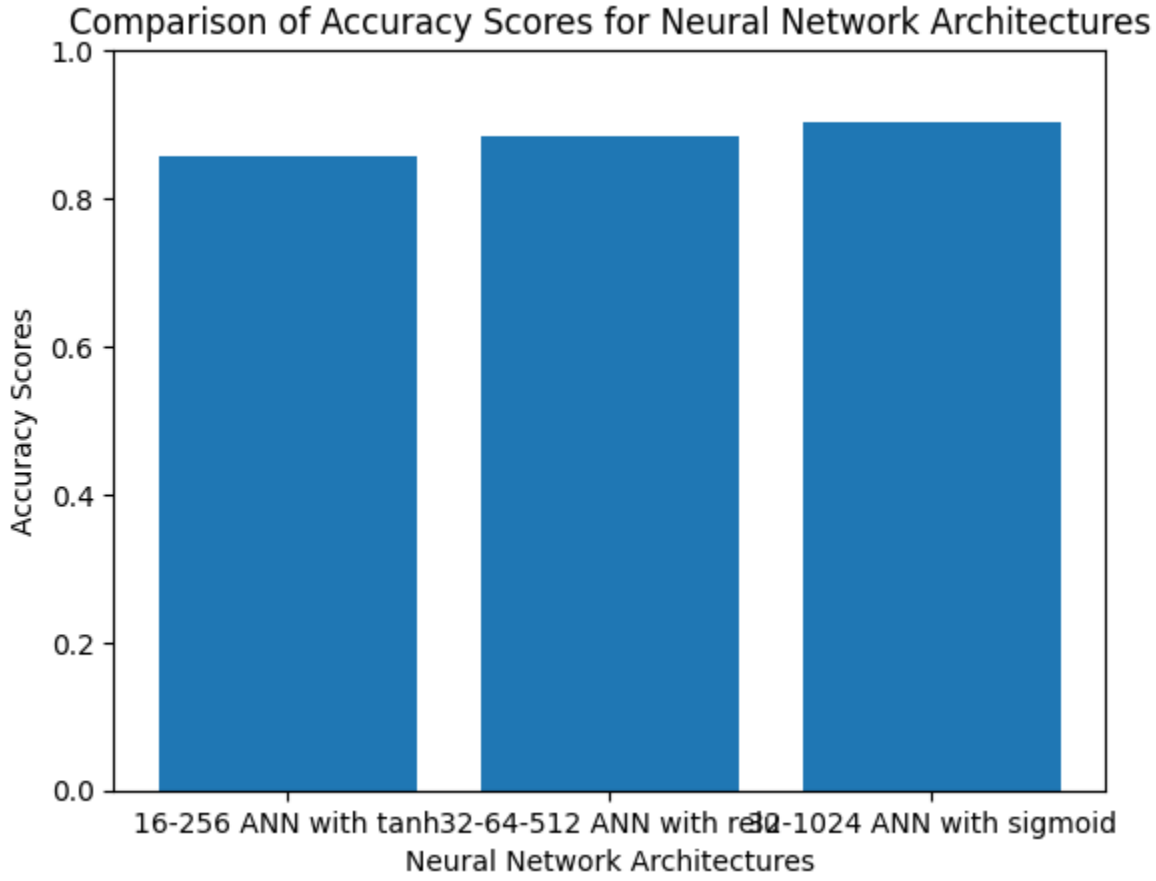
i . Neural Network 3 (32-1024 ANN with ‘sigmoid’):

A 32-1024 Artificial Neural Network (ANN) with 'sigmoid' activation function is a feedforward neural network architecture comprising an input layer with 32 nodes, a single hidden layer with 1024 nodes, and an output layer. The 'sigmoid' activation function, also known as the logistic function, is used in both the hidden and output layers. The input layer receives the input data, which is then propagated through the hidden layer. Each node in the hidden layer applies the 'sigmoid' activation function to the weighted sum of its inputs, introducing non-linearity and enabling the network to capture complex relationships in the data. The outputs of the hidden layer serve as inputs to the output layer, where another set of weights and biases are applied, and the 'sigmoid' activation function is used again. The final outputs from the output layer represent the network's predictions or classifications. During training, the network adjusts the weights and biases using techniques like backpropagation and gradient descent to minimize the difference between the predicted outputs and the expected outputs, ultimately optimizing the network's performance. The 'sigmoid' activation function maps the network's outputs to a range between 0 and 1, making it suitable for binary classification tasks or tasks where probabilities are required.
Here are the stats of the performance metrics of the Neural Network 2:
Test Loss: 0.313
Test Accuracy: 0.884



*** Graphical and Tabular Comparision of Neural Network Architecture Algorithms:**

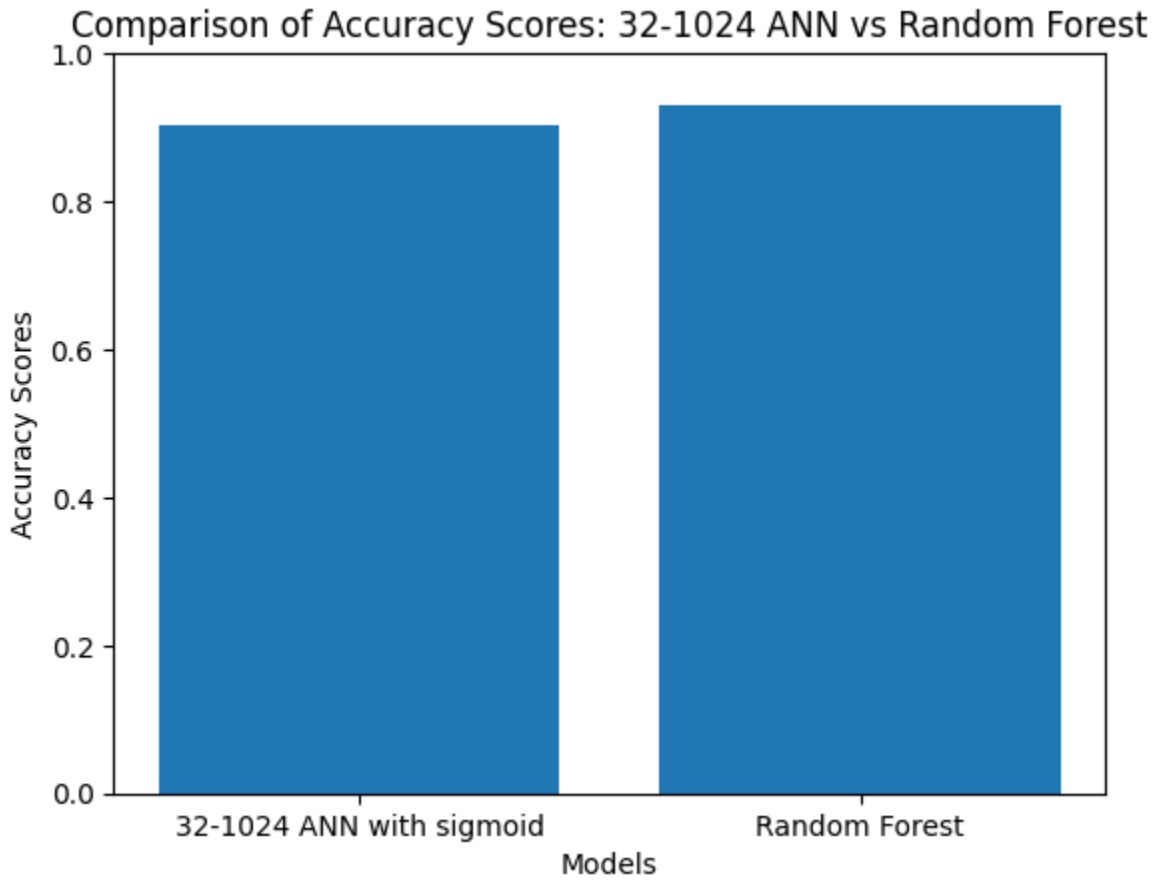
	Activation Functions	Optimizers	Accuracy	Loss
16-256 ANN with 'tanh'	**tanh**	**Adam**	**0.858**	**0.357**
32-64-512 ANN with 'relu'	**relu**	**SGD**	**0.885**	**0.313**
32-1024 ANN with 'sigmoid'	**sigmoid**	**Nadam**	**0.902**	**0.371**



So the best algorithm among the Neural Network Architectures is **32-1024 ANN with "sigmoid"**

* Comparision for the best model:

So the best model among parametric and non-parametric algorithms was **Random Forest** and the best model among Neural Network Architectures is **32-1024 ANN with "sigmoid".** So comparing both the best models so far:



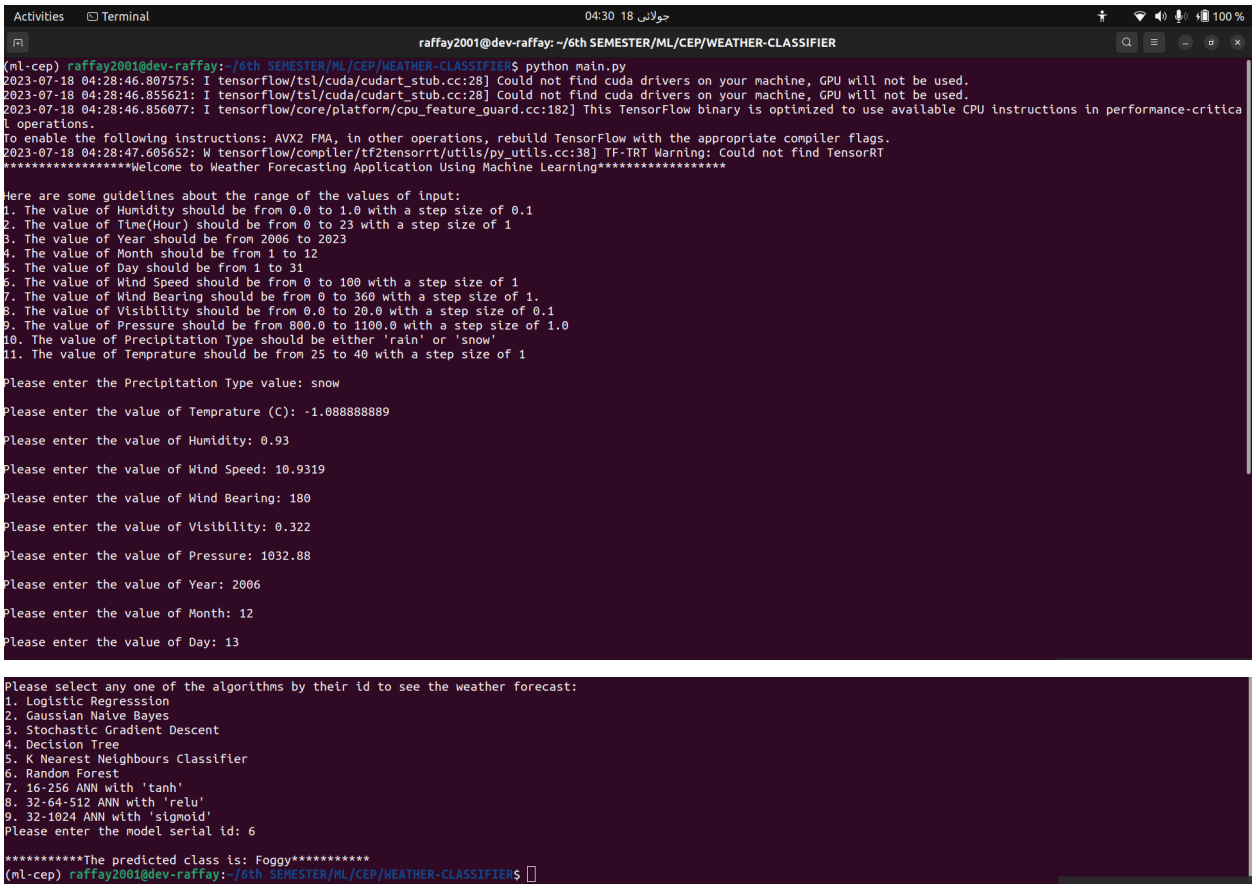
Hence, the best model is **Random Forest** which has the highest accuracy of **93 %.**

Checking for overfitting or underfitting:

During the training of all the 9 models we have checked that none of the models were came out to be overfitted or underfitted.

User Interface:

We also have built an intuitive user interface by which the user can input the features and can select any of the trained 9 models from which he choses to see the output. Here are some screen shots of our CLI application of the system.


Activities
Terminal
04:30 18 جولائی
100 %
raffay2001@dev-raffay: ~/6th SEMESTER/ML/CEP/WEATHER-CLASSIFIER
(ml-cep) raffay2001@dev-raffay:~/6th SEMESTER/ML/CEP/WEATHER-CLASSIFIER$ python main.py
2023-07-18 04:28:46.807575: I tensorflow/tsl/cuda/cudart_stub.cc:28] Could not find cuda drivers on your machine, GPU will not be used.
2023-07-18 04:28:46.855621: I tensorflow/tsl/cuda/cudart_stub.cc:28] Could not find cuda drivers on your machine, GPU will not be used.
2023-07-18 04:28:46.856077: I tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow binary is optimized to use available CPU instructions in performance-critica
l operations.
To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
2023-07-18 04:28:47.605652: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT
**********Welcome to Weather Forecasting Application Using Machine Learning**********
Here are some guidelines about the range of the values of input:
1. The value of Humidity should be from 0.0 to 1.0 with a step size of 0.1
2. The value of Time(Hour) should be from 0 to 23 with a step size of 1
3. The value of Year should be from 2006 to 2023
4. The value of Month should be from 1 to 12
5. The value of Day should be from 1 to 31
6. The value of Wind Speed should be from 0 to 100 with a step size of 1
7. The value of Wind Bearing should be from 0 to 360 with a step size of 1.
8. The value of Visibility should be from 0.0 to 20.0 with a step size of 0.1
9. The value of Pressure should be from 800.0 to 1100.0 with a step size of 1.0
10. The value of Precipitation Type should be either 'rain' or 'snow'
11. The value of Temprature should be from 25 to 40 with a step size of 1
Please enter the Precipitation Type value: snow
Please enter the value of Temprature (C): -1.088888889
Please enter the value of Humidity: 0.93
Please enter the value of Wind Speed: 10.9319
Please enter the value of Wind Bearing: 180
Please enter the value of Visibility: 0.322
Please enter the value of Pressure: 1032.88
Please enter the value of Year: 2006
Please enter the value of Month: 12
Please enter the value of Day: 13

Please select any one of the algorithms by their id to see the weather forecast:
1. Logistic Regresssion
2. Gaussian Naive Bayes
3. Stochastic Gradient Descent
4. Decision Tree
5. K Nearest Neighbours Classifier
6. Random Forest
7. 16-256 ANN with 'tanh'
8. 32-64-512 ANN with 'relu'
9. 32-1024 ANN with 'sigmoid'
Please enter the model serial id: 6
**********The predicted class is: Foggy**********
(ml-cep) raffay2001@dev-raffay:~/6th SEMESTER/ML/CEP/WEATHER-CLASSIFIER$