INTERDISCIPLINARY PROJECT REPORT
at
Sathyabama Institute of Science and Technology
(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering Degree in Computer Science and Engineering

By

LEVAKU VENKATA KOWSHIK REDDY
(Reg No : 40731054)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU

APRIL 2023

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI– 600119

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **LEVAKU VENKATA KOWSHIK REDDY (Reg No : 40731054)** who carried out the project entitled **" Early risk prediction of forest fires based on Machine Learning Algorithm"** under my supervision from Feb 2023 to April 2023.

Internal Guide

Ms. G.Anbu Selvi M.Tech.,(Ph.D)

Head of the Department

Dr. S. Vigneshwari, M.E., Ph.D

Submitted for Viva voce Examination held on____________________

Internal Examiner **External Examiner**

DECLARATION

I,**LEVAKU VENKATA KOWSHIK REDDY(40731054)** hereby declare that the Project Report entitled **" Early risk prediction of forest fires based on Machine Learning Algorithm"** done by me under the guidance of **Ms.G.Anbu Selvi, M.Tech.,(PhD)** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

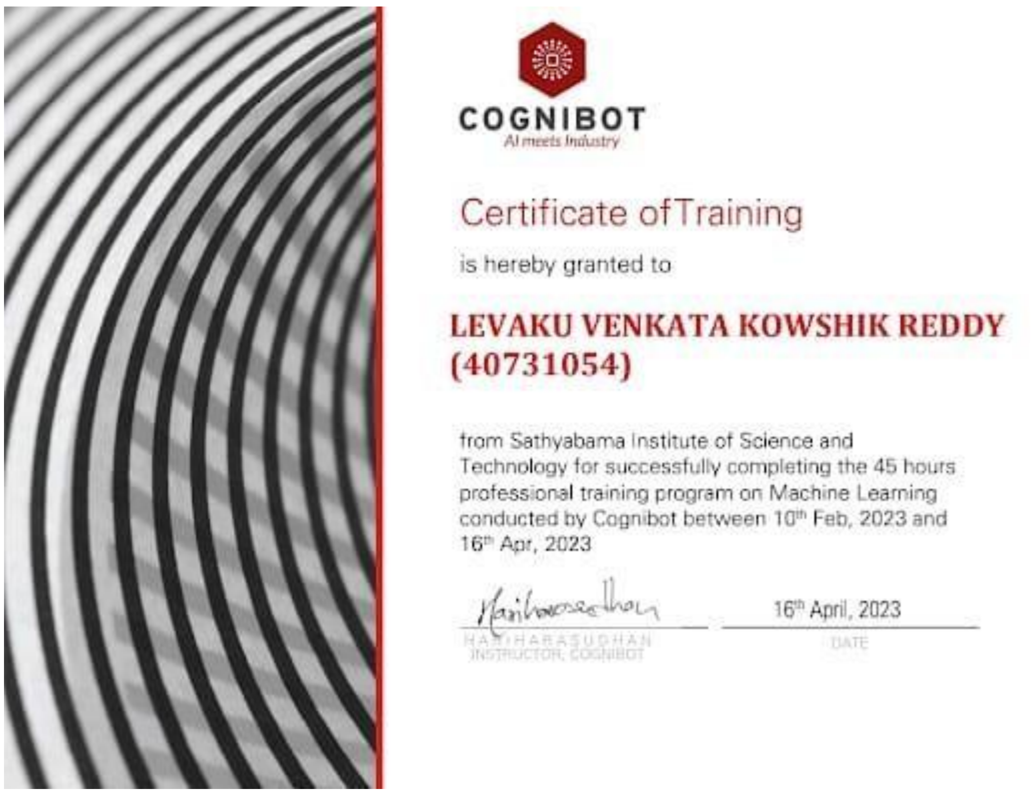
I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr.T.Sasikala M.E.,Ph.D**, **Dean**, School of Computing, **Dr.S.Vigneshwari M.E., Ph.D.,** Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Ms.G.Anbu Selvi, M.Tech.,(Ph.D)**,for her valuable guidance, suggestions andconstant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

TRAINING CERTIFICATE



COGNIBOT
AI meets Industry

Certificate of Training

is hereby granted to

LEVAKU VENKATA KOWSHIK REDDY (40731054)

from Sathyabama Institute of Science and Technology for successfully completing the 45 hours professional training program on Machine Learning conducted by Cognibot between 10th Feb, 2023 and 16th Apr, 2023

[illegible]
HARIHARASUDHAN
INSTRUCTOR, COGNIBOT

16th April, 2023
DATE

ABSTRACT

Algerian forest fires have become a major environmental and socio-economic concern due to their devastating impacts on forests, ecosystems, and human lives. Accurately predicting forest fires is critical for timely detection, management, and prevention of these disasters. Machine learning algorithms have shown promising results in predicting forest fires, and their potential for use in Algeria is currently being investigated. This study aims to explore the use of machine learning algorithms for predicting forest fires in Algeria by analyzing historical data on weather, vegetation, and forest fires. The research will focus on building predictive models using machine learning algorithms such as Random Forest, Support Vector Machines, and Artificial Neural Networks. The models will be trained and tested on a dataset of forest fire incidents in Algeria to evaluate their accuracy and performance. The results will provide insights into the effectiveness of machine learning algorithms in predicting forest fires in Algeria and their potential use in the country's fire management strategies. This study will contribute to the development of early warning systems for forest fires in Algeria, which could help minimize the devastating impacts of these disasters on the environment and society.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER-1
INTRODUCTION

1.1 ESTIMATING FOREST FIRES

Estimating forest fires is a crucial aspect of fire management, as it helps identify the size, intensity, and spread of the fire. One of the primary methods used for estimating forest fires is remote sensing, which involves using satellite data to detect heat signatures and smoke plumes. This data is processed to create maps and images that provide insights into the location and intensity of the fire. Other methods used for estimating forest fires include ground-based observations, aerial surveys, and fire behavior modeling. These methods provide additional information on fire behavior, fuel types, and weather conditions that can help in predicting the fire's future behavior and potential impact.

Accurate estimation of forest fires is essential for timely and effective fire management. Early detection of forest fires and accurate estimation of their size and intensity can help fire managers make informed decisions about resource allocation and prioritization of firefighting efforts. It can also help in the evacuation of affected communities and the implementation of preventative measures to minimize the fire's impact. Therefore, ongoing research on improving estimation methods and developing more advanced technologies, such as machine learning algorithms, is critical for enhancing forest fire management and reducing the negative impacts of these disasters.

1.1.1 TYPES OF FOREST FIRES

There are three basic types of wildfires:

- Crown Fires
- Surface Fires
- Ground Fires

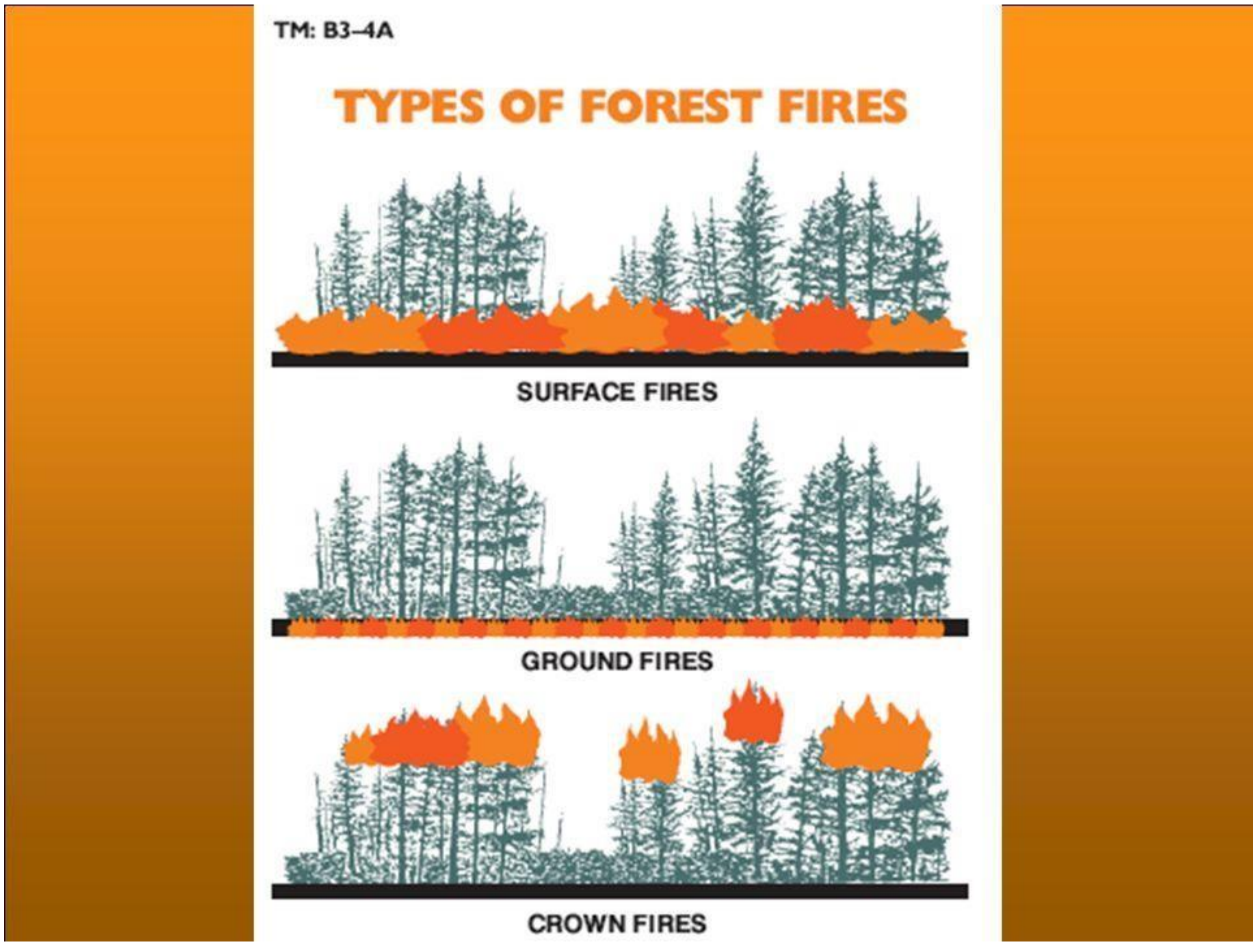
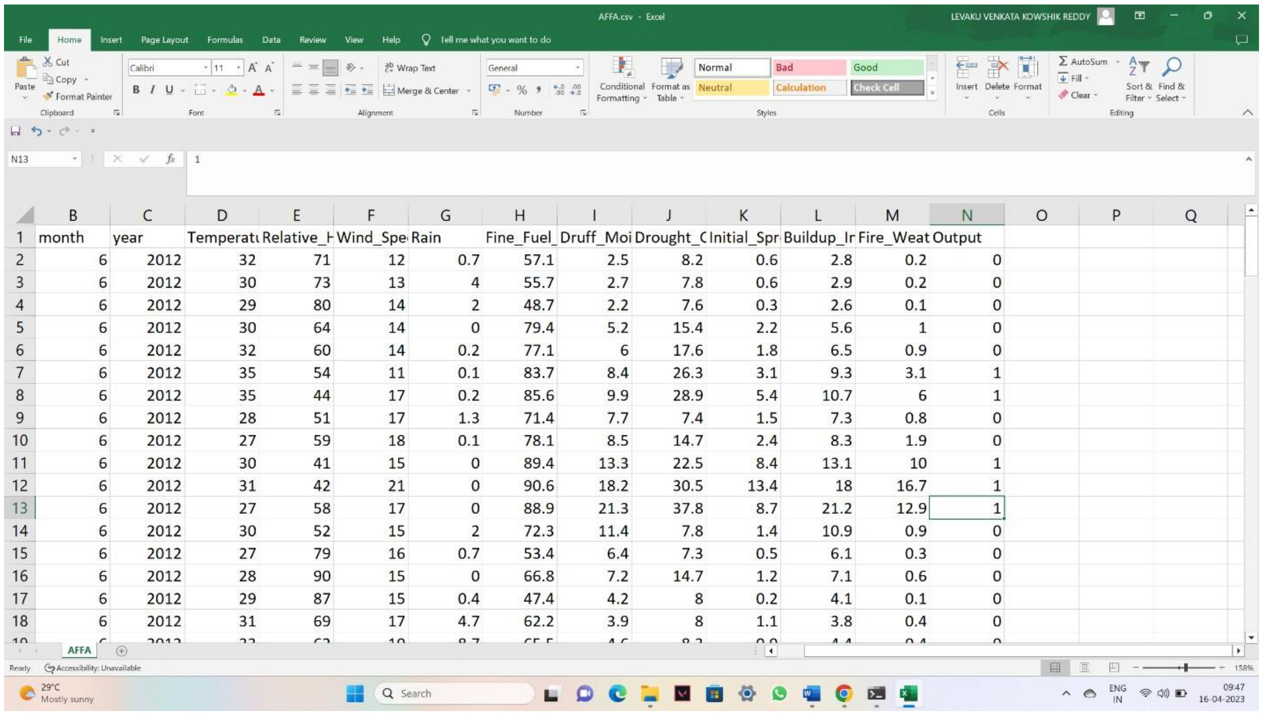


Fig: 1.1.1 Types of Forest Fires

1.2 ESTIMATING FOREST FIRES DATASET INFORMATION

There are 14 attributes in total related to the habits of forest fires and values that are likely to determine forest fires such as day, month, year, temperature, relative humidity,wind speed, rain, fine fuel, druff moisture code, drought code, initial spread index, buildupindex, fire weather index, output



	B	C	D	E	F	G	H	I	J	K	L	M	N
1	month	year	Temperatu	Relative_H	Wind_Spe	Rain	Fine_Fuel_	Druff_Moi	Drought_C	Initial_Spr	Buildup_Ir	Fire_Weat	Output
2	6	2012	32	71	12	0.7	57.1	2.5	8.2	0.6	2.8	0.2	0
3	6	2012	30	73	13	4	55.7	2.7	7.8	0.6	2.9	0.2	0
4	6	2012	29	80	14	2	48.7	2.2	7.6	0.3	2.6	0.1	0
5	6	2012	30	64	14	0	79.4	5.2	15.4	2.2	5.6	1	0
6	6	2012	32	60	14	0.2	77.1	6	17.6	1.8	6.5	0.9	0
7	6	2012	35	54	11	0.1	83.7	8.4	26.3	3.1	9.3	3.1	1
8	6	2012	35	44	17	0.2	85.6	9.9	28.9	5.4	10.7	6	1
9	6	2012	28	51	17	1.3	71.4	7.7	7.4	1.5	7.3	0.8	0
10	6	2012	27	59	18	0.1	78.1	8.5	14.7	2.4	8.3	1.9	0
11	6	2012	30	41	15	0	89.4	13.3	22.5	8.4	13.1	10	1
12	6	2012	31	42	21	0	90.6	18.2	30.5	13.4	18	16.7	1
13	6	2012	27	58	17	0	88.9	21.3	37.8	8.7	21.2	12.9	1
14	6	2012	30	52	15	2	72.3	11.4	7.8	1.4	10.9	0.9	0
15	6	2012	27	79	16	0.7	53.4	6.4	7.3	0.5	6.1	0.3	0
16	6	2012	28	90	15	0	66.8	7.2	14.7	1.2	7.1	0.6	0
17	6	2012	29	87	15	0.4	47.4	4.2	8	0.2	4.1	0.1	0
18	6	2012	31	69	17	4.7	62.2	3.9	8	1.1	3.8	0.4	0

Fig 1.2- Excel sheet of the given estimated forest fires predictions dataset

1.3 COMMON MACHINE LEARNING ALGORITHMS AND GOALS

There are three types of Machine learning algorithms which are been widely used. They are:

- **Supervised Learning**
- **Reinforcement Learning**
- **Unsupervised Learning**
-

Supervised Learning:

Supervised Learning is a machine learning paradigm for problems where the available data consists of labelled examples, meaning that each data point contains features and an associated label. The model is used widely to predict the label of new observations using the features. Depending on the characteristics of the target variable i.e., it can be either **classification**(discrete variable) or **regression**(continuous variable).

Unsupervised Learning:

Unsupervised learning is a Machine learning paradigm for problems where the available data consists of un labelled examples, meaning that each data point contains features only, without an associated label. The goal of unsupervised learning algorithms is learning useful patterns or structural properties of the data. It finds the structures in un labelled data.

Reinforcement Learning:

Reinforcement learning is an area of machine learning concerned with how intelligent agents ought to take actions in an environment in order to maximize the notion of cumulative reward. It works on the action-reward principle. An agent learned to reach the goal by continuously calculating the rewards that it gained from the actions.

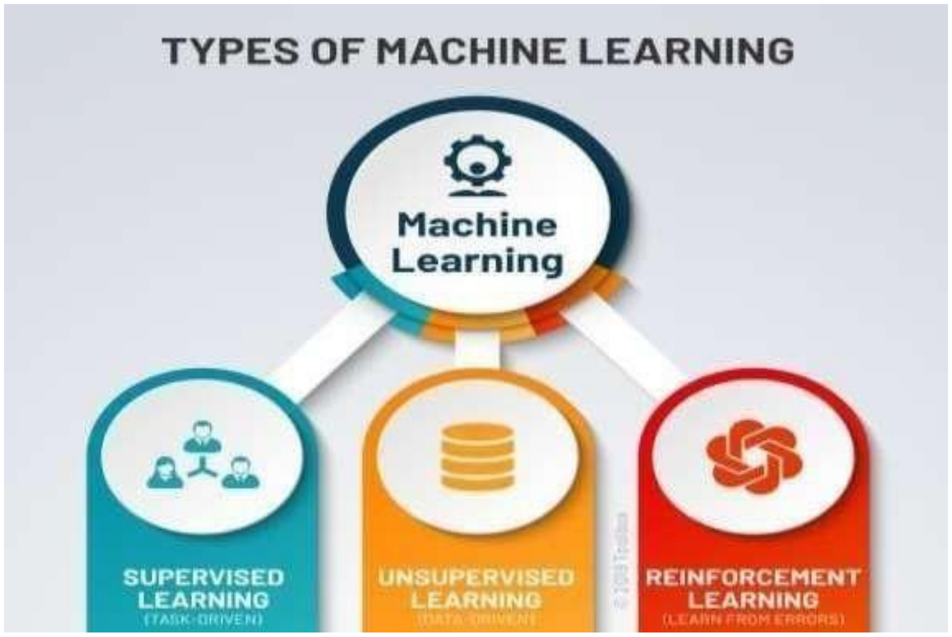


Fig 1.3.1 -Types of machine learning with field of use

ALGORITHMS

1. Linear Regression:

Linear regression is the type of supervised machine learning algorithm where the anticipated output is continuous and features a constant slope. It predicts the values withinendless range, instead of trying to classify them into categories. It is used to predict the worth of a variable supported the worth of another variable. For choosing this algorithm, there needs be a linear relation between independent and target variable. As scatter plot shows the positive correlation between an independent variable(x-axis) and dependent variable (y-axis).

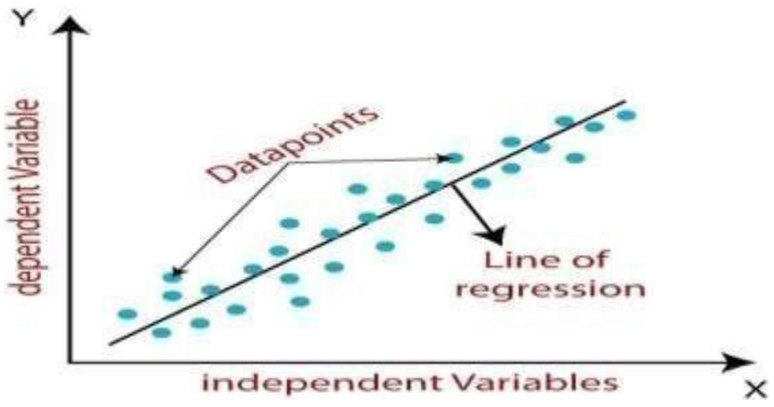


Fig 1.3.2-Linear Regression

2. Naive Bayes classifiers:

Naive Bayes classifiers is a supervised machine learning model for constructing classifiers or models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. It assumes that the factors are independent of each other and there is no correlation between features. As assumptionof features are being uncorrelated, it gets its name as Naive Bayes.

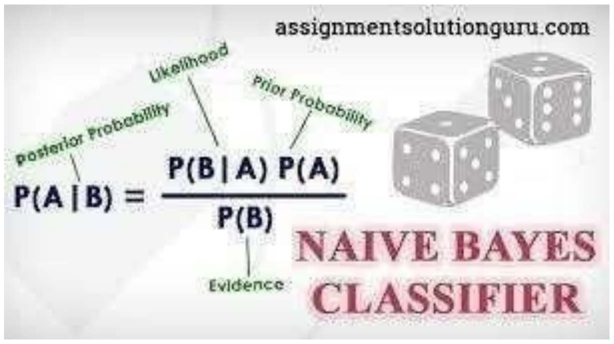


Fig 1.3.3-Naive Bayes Classifier

Where,

p(A|B): Probability of event A given event B has already occurredp(B|A): Probability of event B given event A has already occurredp(A): Probability of event A

p(B): Probability of event B

3. Logistic Regression:

Logistic Regression is a supervised learning algorithm which is mostly used in binary classification problems. Even when regression contradicts with classification, here the spotis for logistic that refers to logistic function which does the classification task. It is simple but effective classification algorithms most commonly used for binary classification problems. Itis also called sigmoid function.

$$\text{Sigmoid Function:}\quad y = \frac{1}{1 + e^{-x}}$$

Logistic regression takes linear equation as input and uses sigmoid function and logs oddsto performs a binary problem. As a result ‘s’ or sigmoid curve will be obtained as the output.

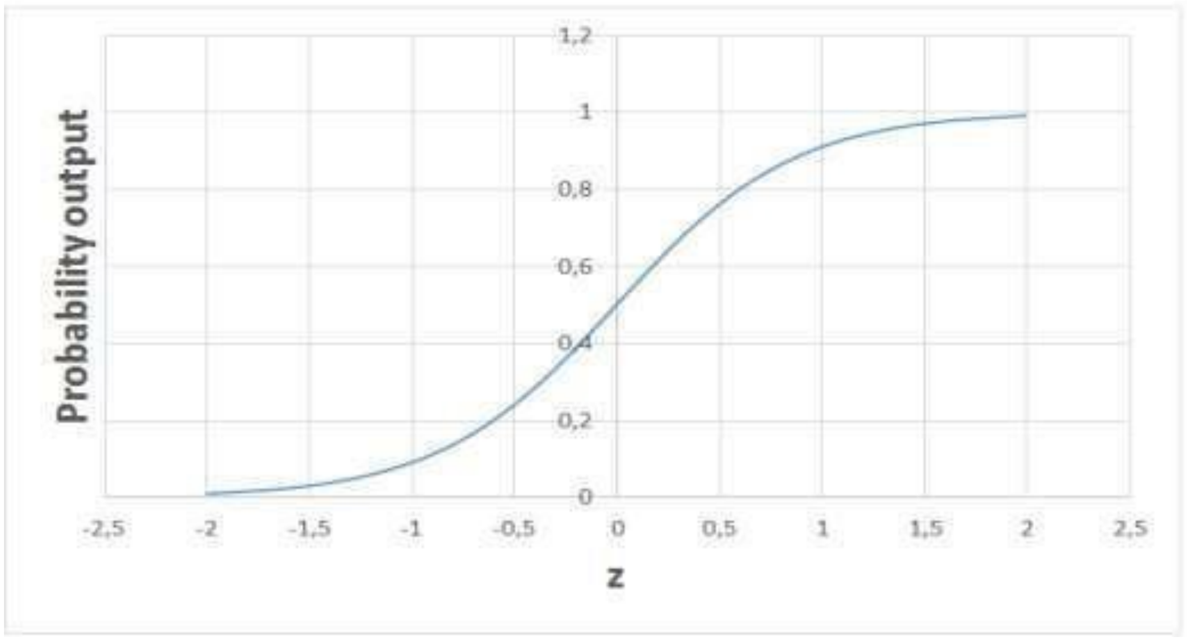


Fig 1.3.4 – Logistic Regression

4. Decision Trees:

Decision trees are a non parametric supervised learning method used for classification andregression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation. Though it achieves high accuracy with training set but poorly on new. The depth of the tree is controlled by max_depth parameter for decision treealgorithm in scikit-learn

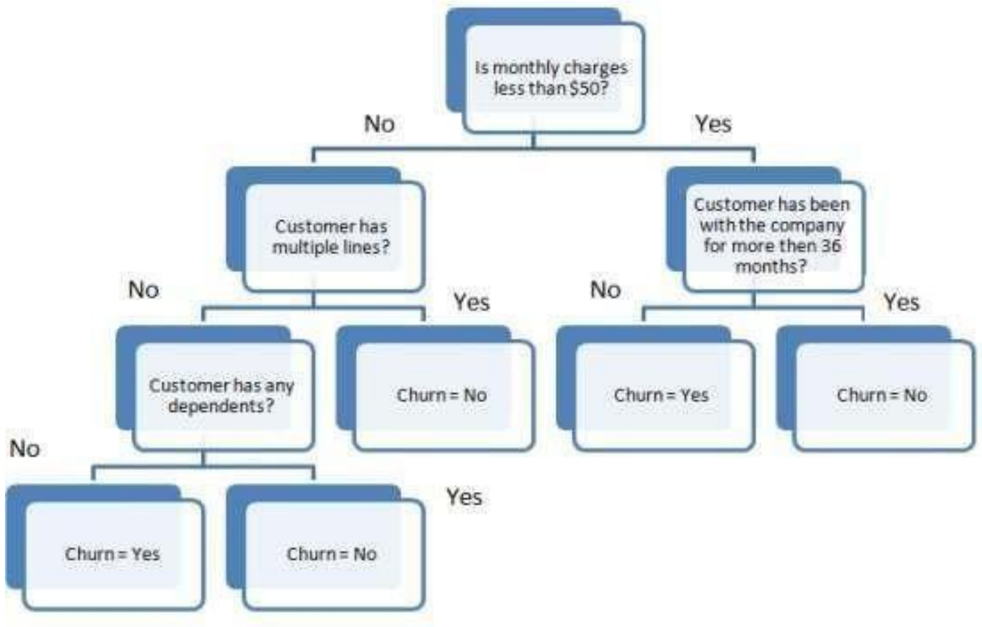

Is monthly charges less than $50?
No
Yes
Customer has multiple lines?
Customer has been with the company for more then 36 months?
No
Yes
No
Yes
Customer has any dependents?
Churn = No
Churn = Yes
Churn = No
No
Churn = Yes
Churn = No
Yes

Fig 1.3.5-Decision Tree

5. Random Forest:

Random forests or Random decision forests is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decisiontrees at training time. For classification tasks, the output of the random forests is the class selected by the most trees.

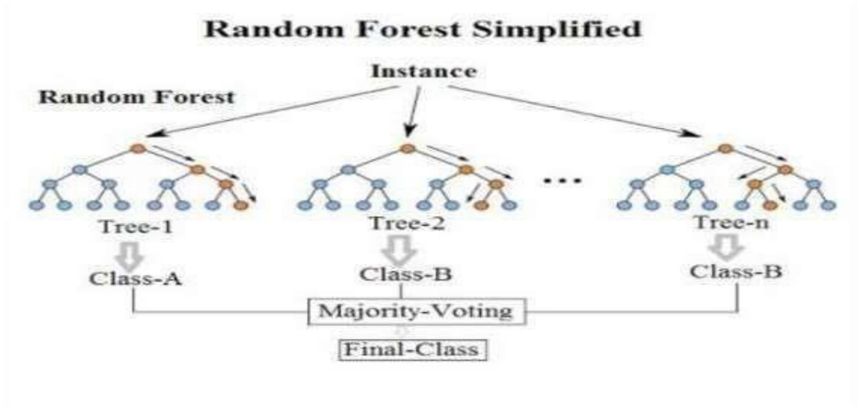


Fig1.3.6-Random Forest

6. K Means Clustering:

K-means clustering is a method of vector quantization, originally from signal processing, that aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. It was a way to group ofset of data points are together. Thus, they took look for dissimilarities or similarities among data points. It is an unsupervised learning so there is no label associated with data points. They try to find the underlying structures of the data. Clustering is not classification.

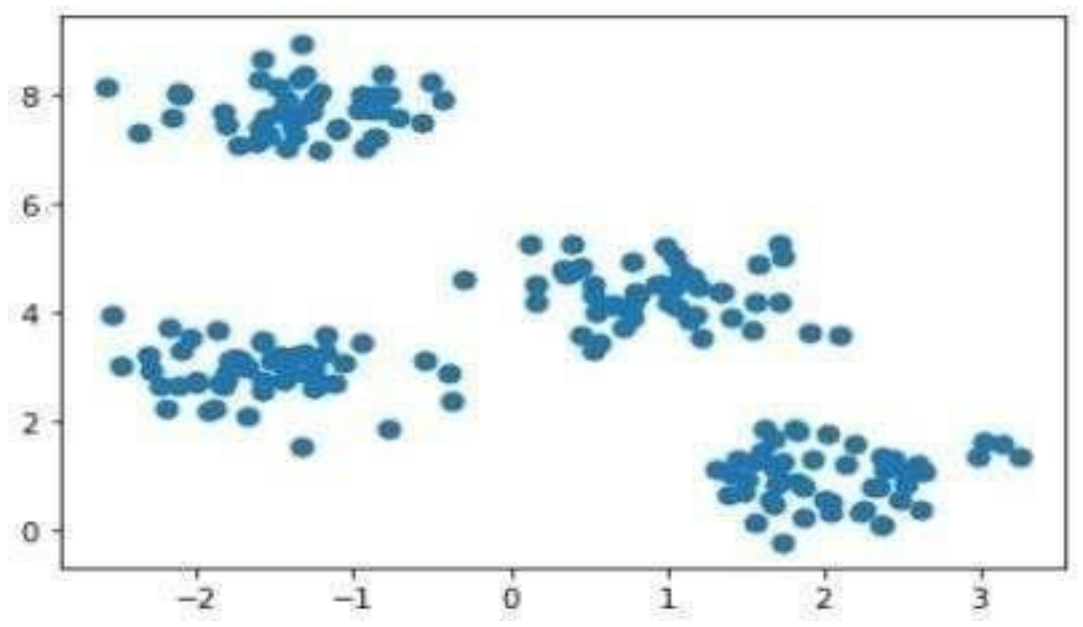


Fig 1.3.7-K Means Clustering

CHAPTER-2

AIM AND SCOPE OF PRESENT INVESTIGATION

2.1 AIM:

To predict the forest fires from the given forest fires dataset.

2.2 SCOPE:

There is a significant scope for predicting forest fires using advanced technologies such as satellite imagery, artificial intelligence (AI), and machine learning (ML) algorithms. These tools can help monitor and analyze various environmental factors that contribute to the risk of forest fires, such as temperature, humidity, wind speed and direction, vegetation moisture content, and topography.Satellite imagery, in particular, can provide valuable information about the extent and severity of forest fires, as well as the location and distribution of smoke and other airborne pollutants. AI and ML algorithms can also be used to analyze historical data and real-time environmental data to identify patterns and predict the likelihood of a forest fire in a particular region.

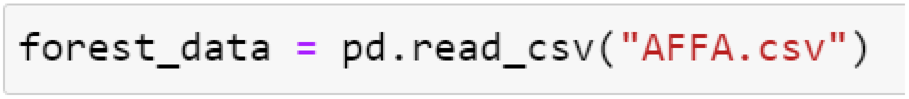
In the given dataset,

- Data set characteristics are Multivariate
- Attribute characteristics are categorical,integer.
- Associated task – Classification
- no.of instances – 122
- no.of attributes – 14

2.3 DATA PREPARATION:

In this project, Python serves as the key tool which carries out important machine learningalgorithms. With the help of Anaconda Navigator, a desktop Graphical User Interface(GUI), a web based interactive application called Jupyter Notebook that allows editing and running notebook documents via web browser. It is an incredibly powerful tool for interactively developing and presenting Machine learning and Data Science projects.After importing the required libraries, the dataset will be read in the notebook with the help of data frame(two-dimensional labeled data structure with columns of potentially different types) and read_csv(desired file type).

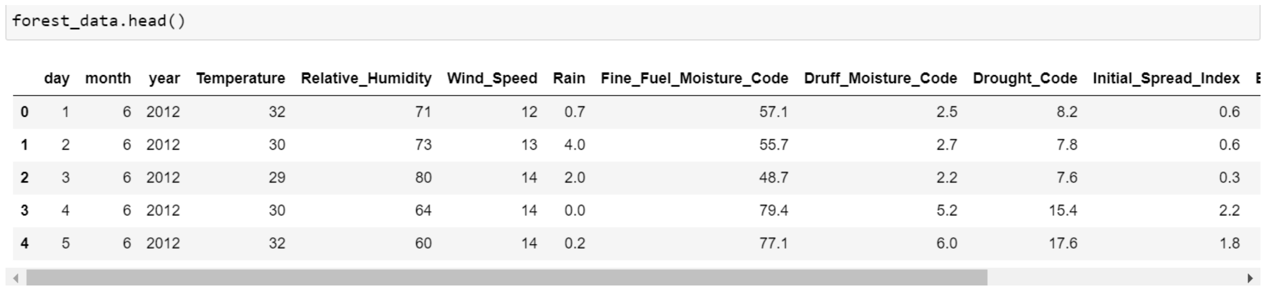
2.4 READING THE DATASET:



```
forest_data = pd.read_csv("AFFA.csv")
```

Fig 2.4.1 -syntax for reading the dataset

The data set provided cannot always be fully valued set, in that case we need to prepare the data in such way the machine understands what is the value that has been entered.

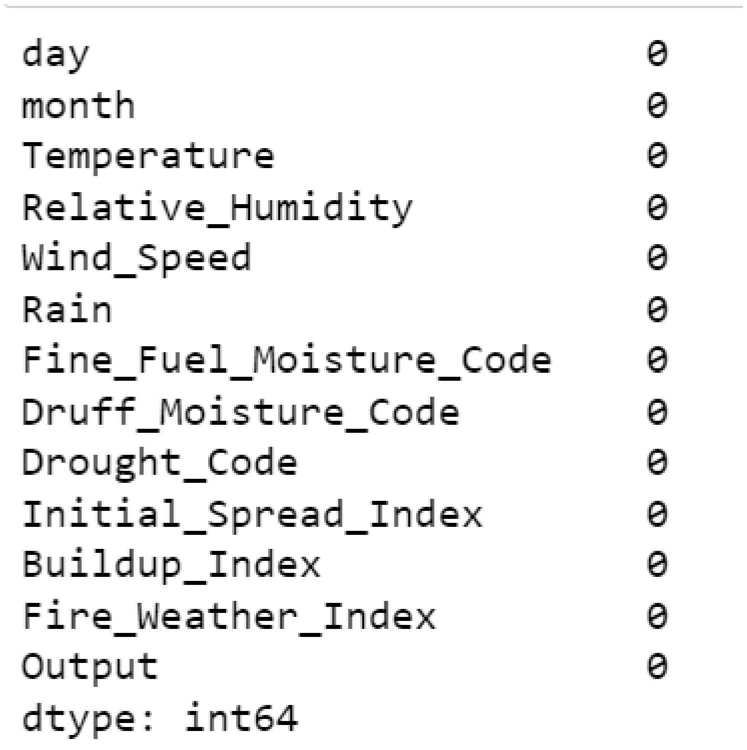


```
forest_data.head()
```

	day	month	year	Temperature	Relative_Humidity	Wind_Speed	Rain	Fine_Fuel_Moisture_Code	Druff_Moisture_Code	Drought_Code	Initial_Spread_Index
0	1	6	2012	32	71	12	0.7	57.1	2.5	8.2	0.6
1	2	6	2012	30	73	13	4.0	55.7	2.7	7.8	0.6
2	3	6	2012	29	80	14	2.0	48.7	2.2	7.6	0.3
3	4	6	2012	30	64	14	0.0	79.4	5.2	15.4	2.2
4	5	6	2012	32	60	14	0.2	77.1	6.0	17.6	1.8

Fig 2.4.2- imported estimated forest fires dataset in jupyter notebook

Check the values for null using isnull() function



```
day                          0
month                        0
Temperature                  0
Relative_Humidity            0
Wind_Speed                   0
Rain                         0
Fine_Fuel_Moisture_Code      0
Druff_Moisture_Code          0
Drought_Code                 0
Initial_Spread_Index         0
Buildup_Index                0
Fire_Weather_Index           0
Output                       0
dtype: int64
```

Fig 2.4.3 -using isnull()function checking for null values

CHAPTER 3
EXPERIMENTAL OR MATERIAL AND METHODS ALGORITHMS USED

The given data set is in the form of classification algorithm. So we used classification types to predict the accuracy

3.1 TYPES OF CLASSIFICATIONS ALGORITHMS USED:

1. Decision Tree Algorithm
2. Random Forests Algorithm

3.2 DECISION TREE ALGORITHM:

Decision Tree is a decision support tool that uses a tree like model of decisions and theirpossible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

The goal is to create a model that predicts the value of a target variable by learning simpledecision rules inferred from the data features. A tree can be seen as a piecewise constantapproximation.

For instance, decision trees learn from data to approximate a sine curve with a set of if- then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model.

Some of the advantages of decision trees are being simple to understand and to interpretand it requires very little data preparation whereas other methods require data normalization, dummy variables etc.

Decision Tree classifier is a class capable of performing multi-class classification on a dataset. As with other classifiers, Decision Tree Classifier takes as input two arrays: an array X, sparse or dense, of shape holding the training samples, and an array Y of integervalues, shape holding the class labels for the training samples.

Why use Decision Trees?

- Decision Trees usually mimic human thinking ability while making a decision, so itis easy to understand.
- The logic behind the decision tree can be easily understood because it shows atree-like structure.

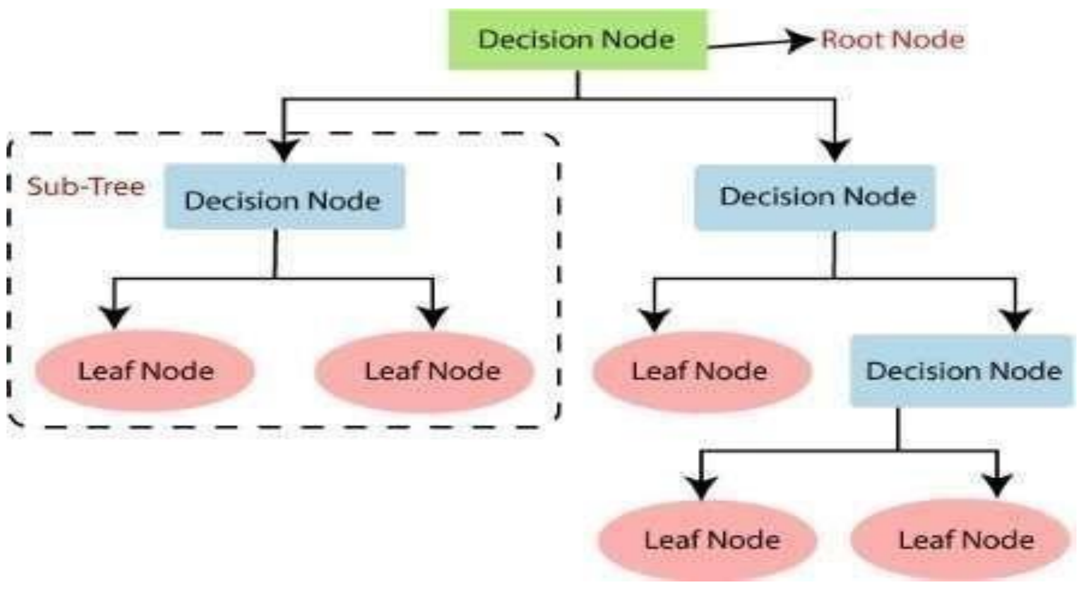


Fig 3.2.1-Decision Tree Algorithm

Decision Tree Terminologies

Root Node: Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.

Leaf Node: Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.

Splitting: Splitting is the process of dividing the decision node/root node into subnodes according to the given conditions

Parent/Child node: The root node of the tree is called the parent node, and other nodesare called the child nodes.

How does the Decision Tree algorithm Work?

In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node. For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reachesthe leaf node of the tree. The complete process can be better understood using the below algorithm:

- **Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.
- **Step-2:** Find the best attribute in the dataset using Attribute Selection Measure (ASM).
- **Step-3:** Divide the S into subsets that contains possible values for the best attributes.
- **Step-4:** Generate the decision tree node, which contains the best attribute.
- **Step-5**: Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannotfurther classify the nodes and called the final node as a leaf node.

Example: Suppose there is a candidate who has a job offer and wants to decide whetherhe should accept the offer or not. So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM). The root node splits further into the next decision node (distance from the office) and one leaf node based on the corresponding labels. The next decision node further gets split into one decision node (Cab facility) and one leaf node.

Finally, the decision node splits into two leaf nodes (Accepted offers and Declined offer).

3.3 RANDOM FORESTS:

Random forests or Random decision forests is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. It is a supervised learning machine learning algorithm made up of decision trees which are used in classification and regression. It plays a majorrole in our model.It is called a “forest” because it grows a forest of decision trees. The data from these trees are then merged together to ensure the most accurate predictions.While a solo decision tree has one outcome and a narrow range of groups, the forest assures a more accurate result with a bigger number of groups anddecisions. It has theadded benefit of adding randomness to the model by finding the best feature among a random subset of features. Overall, these benefits create a modelthat has wide diversitythat many data scientists favor.

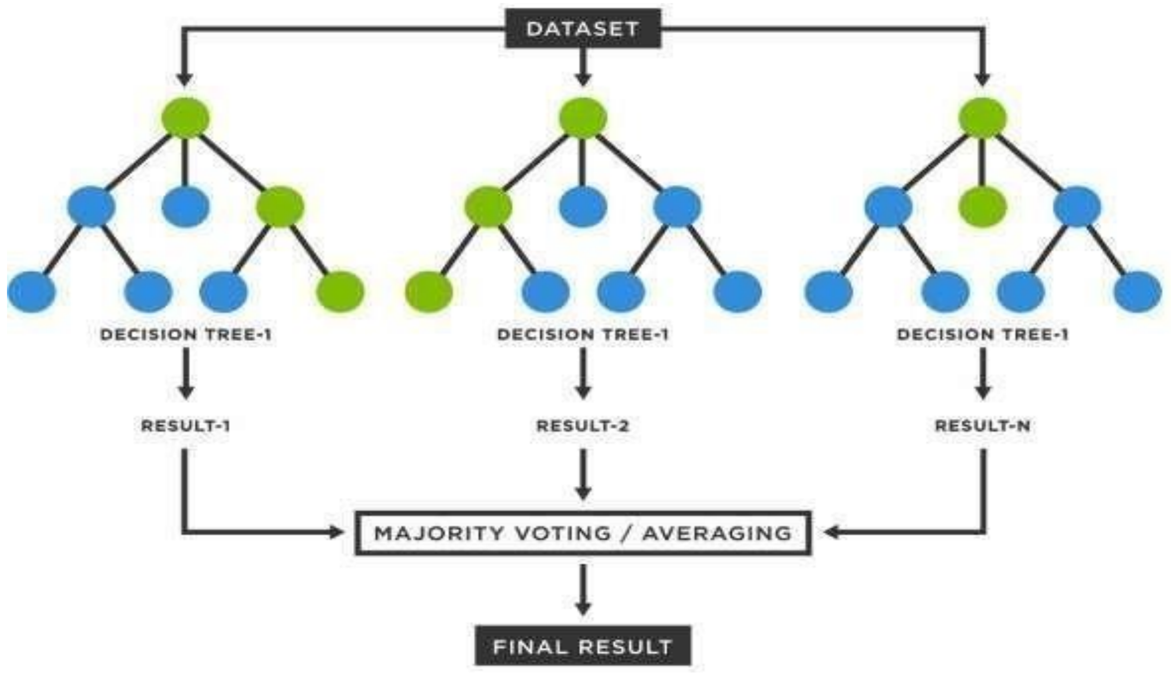


Fig 3.3.1 -Random Forest Algorithm

Assumptions for Random Forest

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random Forest classifier:

There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.

The predictions from each tree must have very low correlations.

Why use Random Forest?

- Below are some points that explain why we should use the Random Forestalgorithm. It takes less training time as compared to other algorithms.
- It predicts output with high accuracy, even for the large dataset it runs efficiently.
- It can also maintain accuracy when a large proportion of data is missing.

How does Random Forest algorithm work?

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase. The Working process can be explained in the below steps and diagram:

Step-1: Select random K data points from the training set.

Step-2: Build the decision trees associated with the selected data points 13

Step-3: Choose the number N for decision trees that you want to build.

Step-4: Repeat Step 1 & 2.

Step-5: For new data points, find the predictions of each decision tree, and assign

the new data points to the category that wins the majority votes.

Applications of Random Forest:

There are mainly four sectors where Random Forest mostly used:

1. Banking: Banking sector mostly uses this algorithm for the identification of loan risk.

2. Medicine: With the help of this algorithm, disease trends and risks of the disease canbe identified.

3. Land Use: We can identify the areas of similar land use by this algorithm.

4. Marketing: Marketing trends can be identified using this algorithm.

Advantages of Random Forest:

- Random Forest is capable of performing both Classification and Regression tasks.
- It is capable of handling large datasets with high dimensionality.
- It enhances the accuracy of the model and prevents the overfitting issue.

Disadvantages of Random Forest:

Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.

3.4 MACHINE LEARNING LIBRARIES:

Libraries are collections of prewritten code that users can use to optimize tasks. In projetas python is used for implementation tool, it has the most libraries as compared to otherprogramming languages. More than of 60% machine learning developers use and goes for python as it is easy to learn. As python has comparatively large collection of librarieslets look at the libraries that came handy for our dataset.

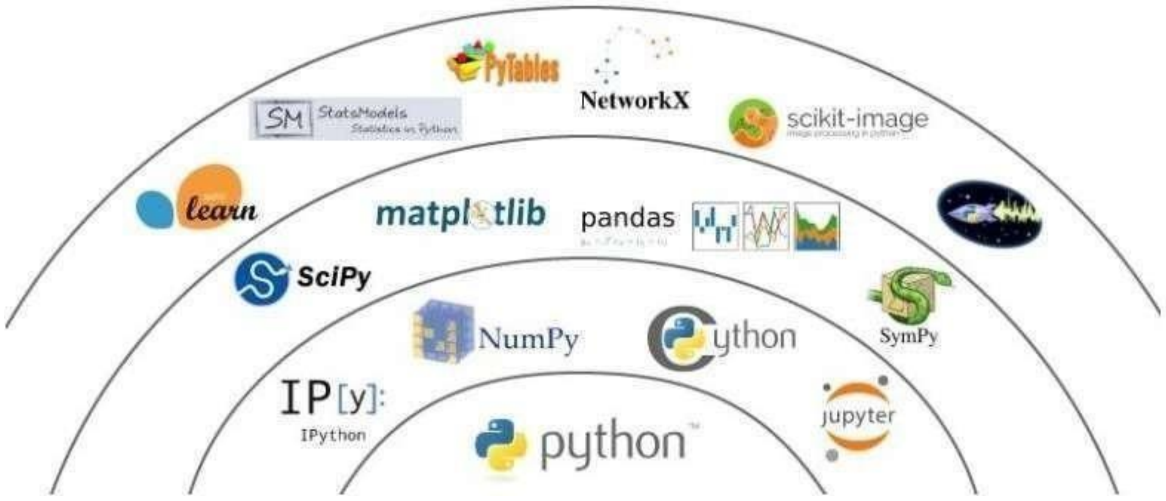


Fig 3.4.1-Various python libraries for machine learning

1. Sklearn:

Sklearn stands for Scikit-learning, a machine learning library. It is imported for various classification, regression and clustering algorithms including k-means,random forest, support vector machines, gradient boosting and DBSCAN. It is designed using libraries Numpy and Scipy. From the sklearn library and from the tree inside the library DecisionTreeClassifier.It is a class capable of performing multi-class classifier on a dataset. When compared with other classifiers, DecisionTreeClassifier takes input as two arrays:an array X, aparse or dense,of shape(n_samples,n_features) holding training samples and an array Y of integer values, shape holding class labels for training sample.From sklearn another one called model_selection for training and testing the model imports train_test_split. It is a method setting a blueprint to analyze data and using it to measure new data. Selecting a proper model allows to generate accurate results while making prediction. For proceeding, we need to train the model by using a specific datasetand test the model by using a specific dataset and test the model against another dataset.By default,sklearntrain_test_split will make random partitions for two subsets. We can also specify a random state for the operation. First, we need to split the dataset and thenallocate the size for train and test.

2. Math:

Math is a built-in module that you can use for mathematical tasks. It has set of methods and constants. It is a standard module in python and is always available

3. Seaborn:

Seaborn is a library built on top of matplotlib. It is used for data visualization and exploratory data analysis. They work easily with dataframes and pandas library. The graphs created can also be customized easily. It provide default styles and color palettesto make statistical plots more attractive. Also closely integrated to the data structures from pandas

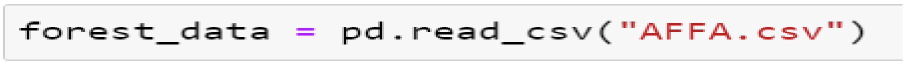
4. Matplotlib.pyplot:

Matplotlib.pyplot is a state-based interface to matplotlib. It provides a MATLAB-like way of plotting. It make changes to figures.

3.5 IMPORTED LIBRARIES

1.Pandas:

Pandas is a widely used data analysis and manipulation library for python. It provides a lot of functions and methods that expedite the data analysis and preprocessing steps. It also provides fast, flexible and expressive data structures working with relational or labeled or both easy and intuitive. Considered as fundamental high-level building block in performing practical, real world data analysis in python. It has Data Frame and series for analyzing.

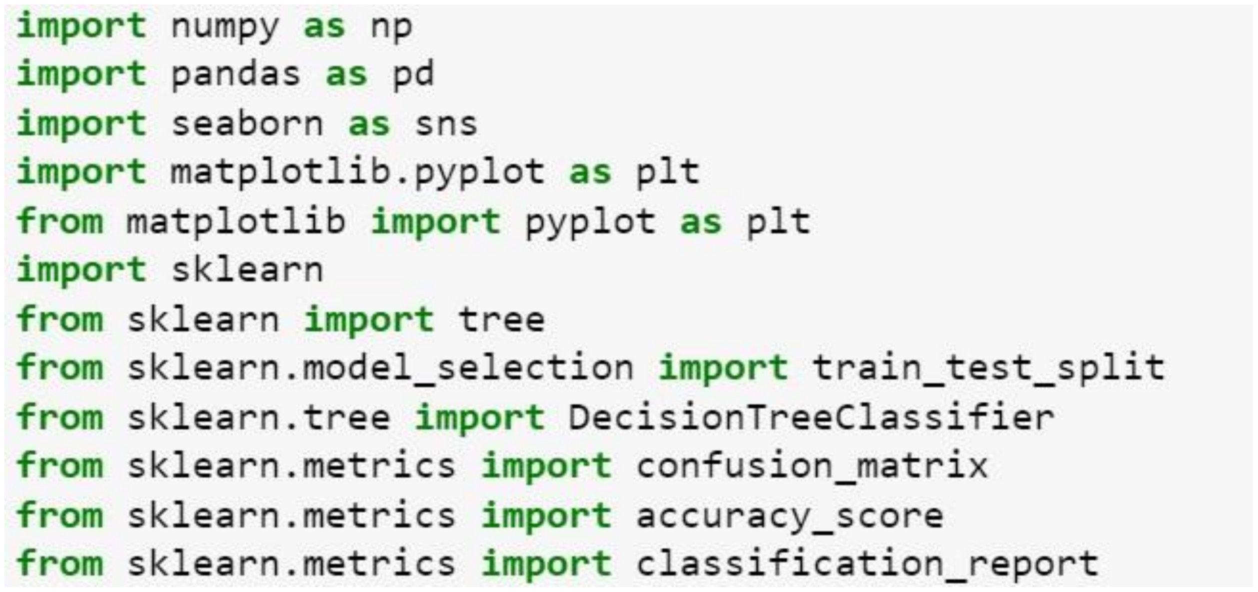


```
forest_data = pd.read_csv("AFFA.csv")
```

Fig 3.5.1- Pandas library is used to read the dataset

2.Numpy:

Numpy stands for Numerical python, is a library consisting of multidimensional array objects and a collection of countless of routines for processing those arrays. Using this mathematical and logical operations on arrays can be performed. The difference in Numpy from Pandas is, it works on numerical data whereas pandas on tabular data.



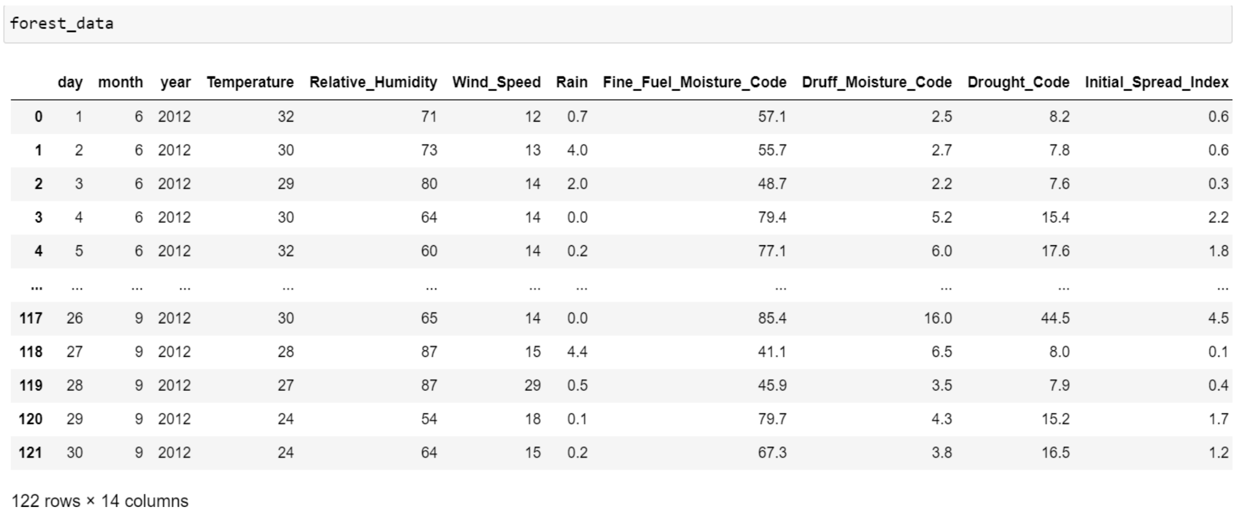
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib import pyplot as plt
import sklearn
from sklearn import tree
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
```

Fig 3.5.2 – Imported libraries in jupyter notebook

3.6 IMPLEMENTATION OF DECISION TREE ALGORITHM

Decision trees are a non parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen asa piecewise constant approximation. Though it achieves high accuracy with training set but poorly on new. The depth of the tree is controlled by max_depth parameter for decision tree algorithm in scikit-learn.For instance, decision trees learn from data to approximate a sine curve with a set of if-then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model. Some of the advantages ofdecision trees are being simple to understand and to interpret and it requires very little data preparation whereas other methods require data normalization, dummy variables etc.

1. Import the packages and classes you need.
2. Provide data to work with and eventually do appropriate transformations.
3. Create a classification model and fit it with existing data.
4. Check the results of model fitting to know whether the model is satisfactory.
5. Apply the model of predictions.

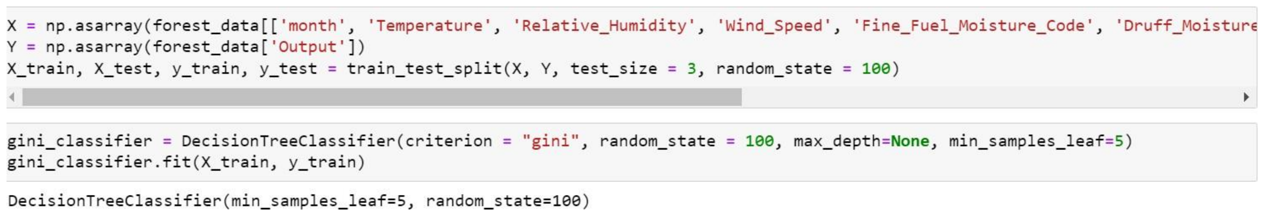


```
forest_data
```

	day	month	year	Temperature	Relative_Humidity	Wind_Speed	Rain	Fine_Fuel_Moisture_Code	Druff_Moisture_Code	Drought_Code	Initial_Spread_Index
0	1	6	2012	32	71	12	0.7	57.1	2.5	8.2	0.6
1	2	6	2012	30	73	13	4.0	55.7	2.7	7.8	0.6
2	3	6	2012	29	80	14	2.0	48.7	2.2	7.6	0.3
3	4	6	2012	30	64	14	0.0	79.4	5.2	15.4	2.2
4	5	6	2012	32	60	14	0.2	77.1	6.0	17.6	1.8
...	...	...	...	...	...	...	...	...	...	...	...
117	26	9	2012	30	65	14	0.0	85.4	16.0	44.5	4.5
118	27	9	2012	28	87	15	4.4	41.1	6.5	8.0	0.1
119	28	9	2012	27	87	29	0.5	45.9	3.5	7.9	0.4
120	29	9	2012	24	54	18	0.1	79.7	4.3	15.2	1.7
121	30	9	2012	24	64	15	0.2	67.3	3.8	16.5	1.2

122 rows × 14 columns

Fig 3.6.1-Given dataset



```
X = np.asarray(forest_data[['month', 'Temperature', 'Relative_Humidity', 'Wind_Speed', 'Fine_Fuel_Moisture_Code', 'Druff_Moisture
Y = np.asarray(forest_data['Output'])
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 3, random_state = 100)
```

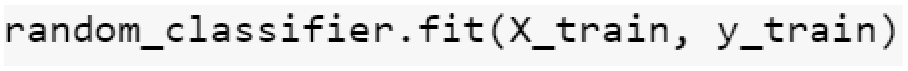
```
gini_classifier = DecisionTreeClassifier(criterion = "gini", random_state = 100, max_depth=None, min_samples_leaf=5)
gini_classifier.fit(X_train, y_train)
```

```
DecisionTreeClassifier(min_samples_leaf=5, random_state=100)
```

Fig 3.6.2-Applying decision tree algorithm

3.7 IMPLEMENTATION OF RANDOM FORESTS ALGORITHM:

Random forests or Random decision forests is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest isthe class selected by most trees. It is a supervised learning machine learning algorithm made up of decision trees which are used in classification and regression. It plays a majorrole in our model. As we know that a forest is made up of trees and more trees means more robust forest. Similarly, random forest algorithm creates decision trees on data samples and then gets the prediction from each of them and finally selects the best solution by means of voting. It is an ensemble method which is better than a single decision tree because it reduces the over-fitting by averaging the result.



```
random_classifier.fit(X_train, y_train)
```

Fig 3.7.1 Applying random forest algorithm

CHAPTER 4

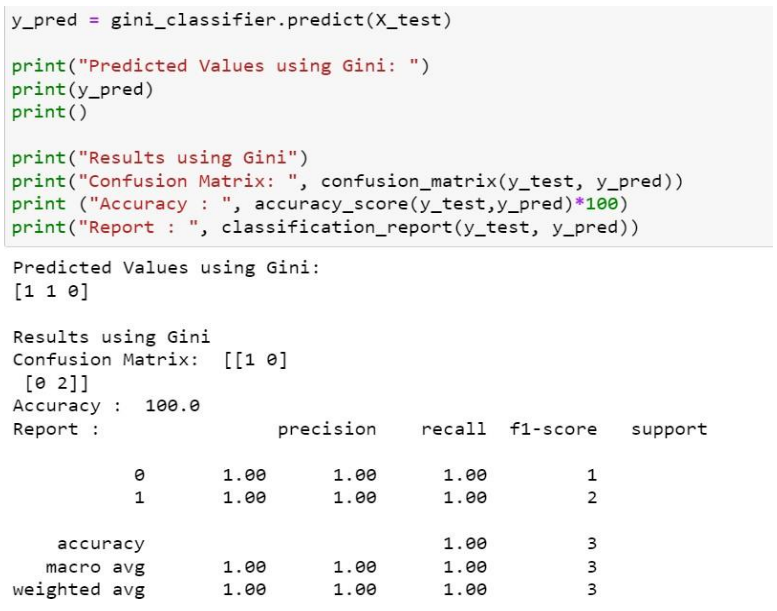
RESULTS AND DISCUSSION,PERFORMANCE ANALYSIS

4.1 MODEL ANALYSIS

The above algorithms are written in python with the help of numpy and executed using jupyter notebook. The accuracy rate of decision tree algorithm when executed for the given dataset i.e Forest Fires prediction Dataset is 100.00 and for random forest is 100.00

The dataset contains 122 training samples with 14 features (day, month, year, temperature, relative humidity, wind speed, rain, fine fuel, druff moisture code, drought code, initial spread index, buildup index, fire weather index, output) our task is to determine the accuracy of working of decision tree and random forest algorithm when trained with the provided dataset

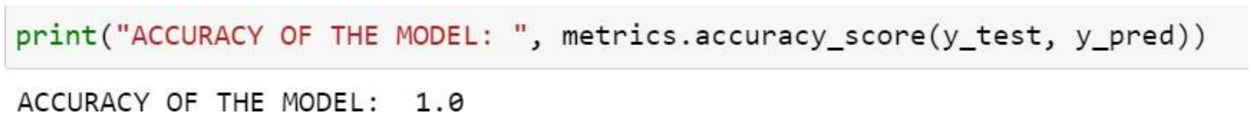
ACCURACY, PRECISION, RECALL, F1-SCORE



```
y_pred = gini_classifier.predict(X_test)

print("Predicted Values using Gini: ")
print(y_pred)
print()

print("Results using Gini")
print("Confusion Matrix: ", confusion_matrix(y_test, y_pred))
print ("Accuracy : ", accuracy_score(y_test,y_pred)*100)
print("Report : ", classification_report(y_test, y_pred))
```

```
Predicted Values using Gini:
[1 1 0]

Results using Gini
Confusion Matrix:  [[1 0]
 [0 2]]
Accuracy :  100.0
Report :               precision    recall  f1-score   support

           0       1.00      1.00      1.00         1
           1       1.00      1.00      1.00         2

    accuracy                           1.00         3
   macro avg       1.00      1.00      1.00         3
weighted avg       1.00      1.00      1.00         3
```

Fig 4.1.1 Accuracy,precision,recall,f1-score for decision tree algorithm



```
print("ACCURACY OF THE MODEL: ", metrics.accuracy_score(y_test, y_pred))

ACCURACY OF THE MODEL:  1.0
```

Fig 4.1.2 Accuracy of Random forest algorithm

CHAPTER 5

5.1 SUMMARY AND CONCLUSIONS

Our goal was to predict diabetes risk accurately using Random forests and Decision Tree algorithms.Here, we will train both the models and test the accuracy of them . There was a significant improvement in the accuracy levels upto 100 in decision tree classification and 100 in random forest classifications respectively.we got a significant accuracy while training the model using random forest classifier.While the accuracy is high, the model is not substantially overfitted, as the cross-validation scores for each of the ensemble methods all differed from the model accuracy by less than 1%.Also, we concluded that f1_score,precision,recall by using these random forest and decision tree classifier.

.

REFERENCES

[1] Refferal code for forest fires

https://www.kaggle.com/datasets/elikplim/forest-fires-data-set/code

[2] Dataset

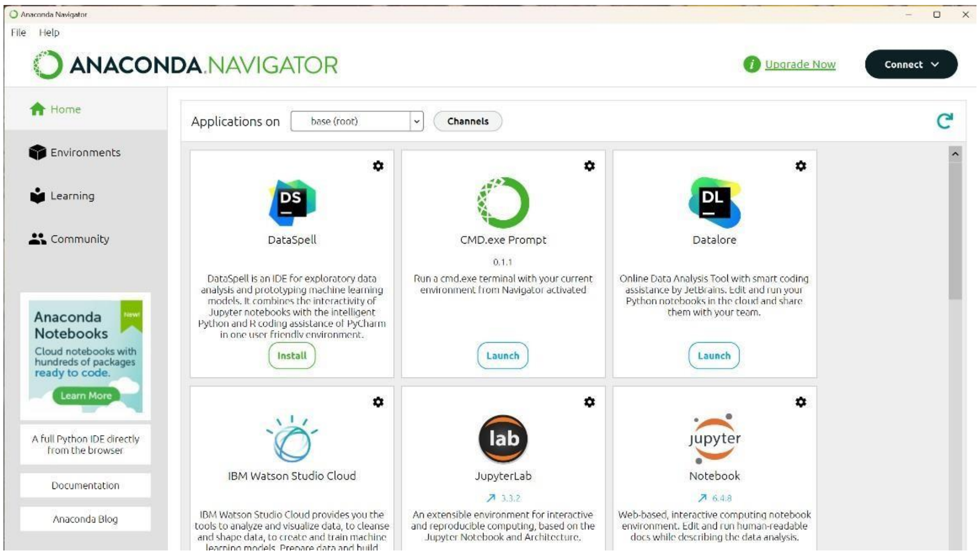
https://drive.google.com/drive/folders/1xf6uwCPMXsHxHhpax8agm4dkJQWNavDE

[3] Dataset Information

https://archive.ics.uci.edu/ml/datasets/Algerian+Forest+Fires+Dataset++

WORKING ENVIRONMENT

ANACONDA NAVIGATOR is desktop GUI used to launch applications and also manage packages in one place.



DataSpell is an IDE for exploratory data analysis and prototyping machine learning models. It combines the interactivity of Jupyter notebooks with the intelligent Python and R coding assistance of PyCharm in one user friendly environment.

Run a cmd.exe terminal with your current environment from Navigator activated

Online Data Analysis Tool with smart coding assistance by JetBrains. Edit and run your Python notebooks in the cloud and share them with your team.

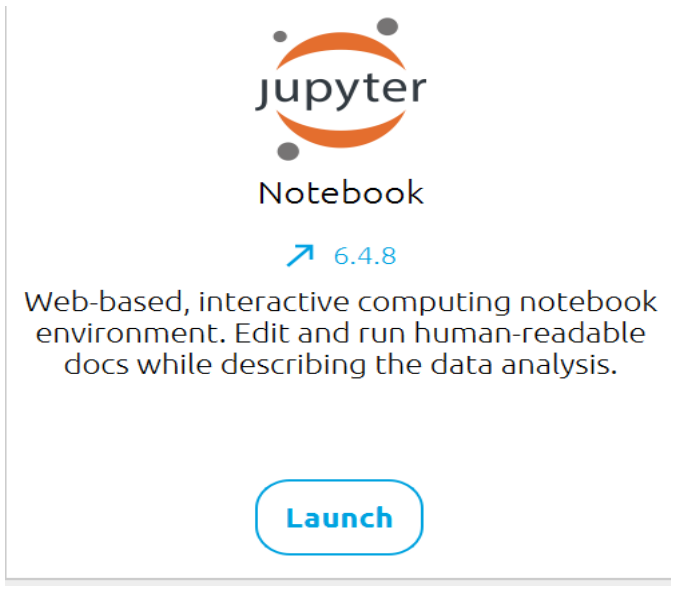
IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

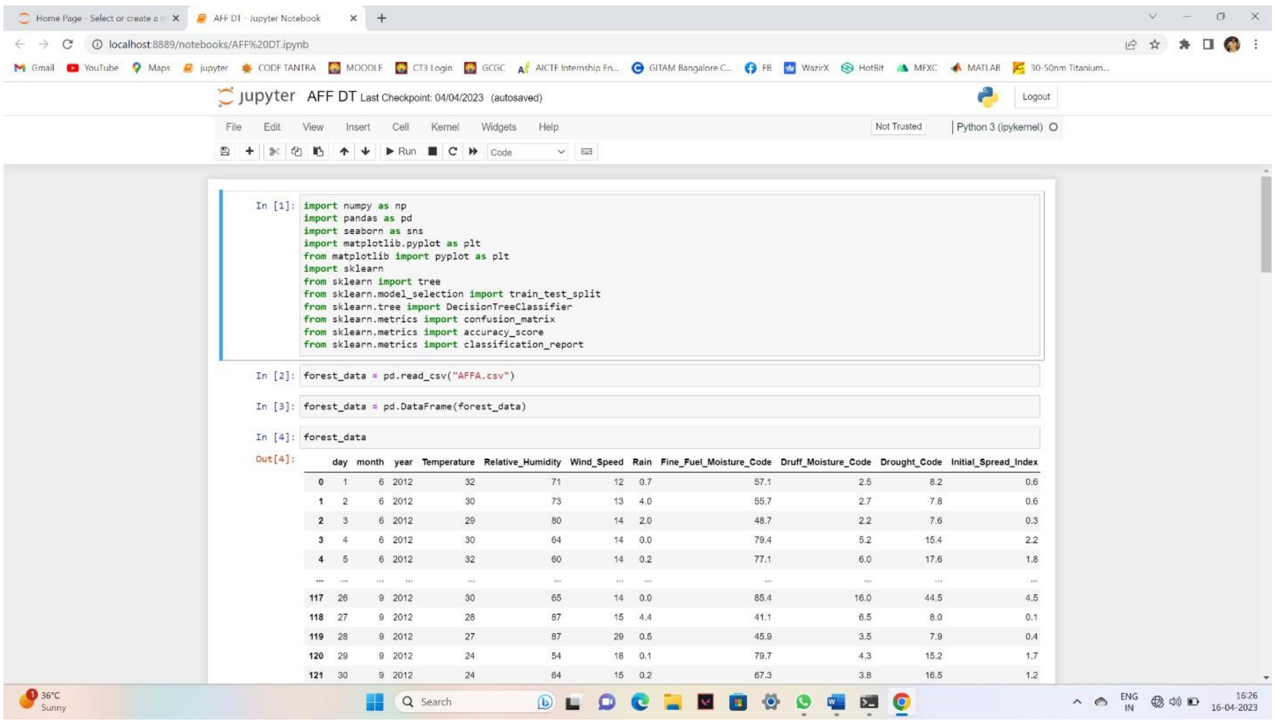
CODING ENVIRONMENT

Jupyter notebook from the anaconda navigator is launched along with all the preinstalled packages for python.

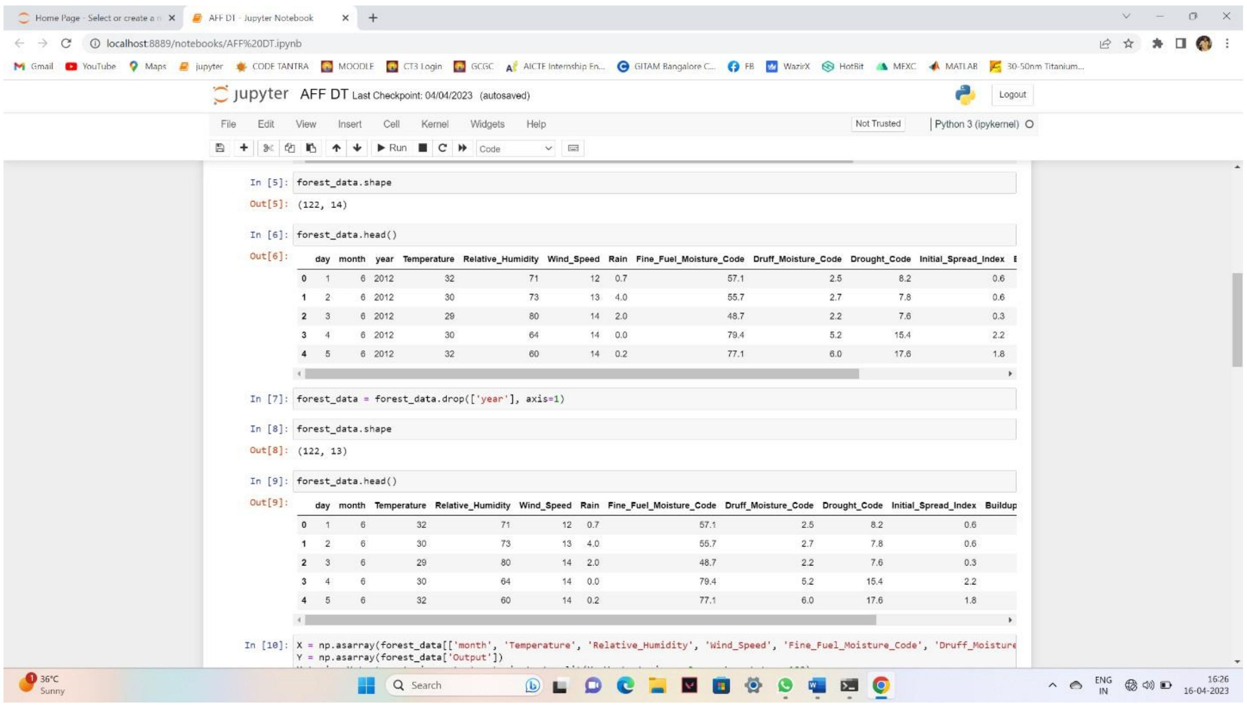


Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

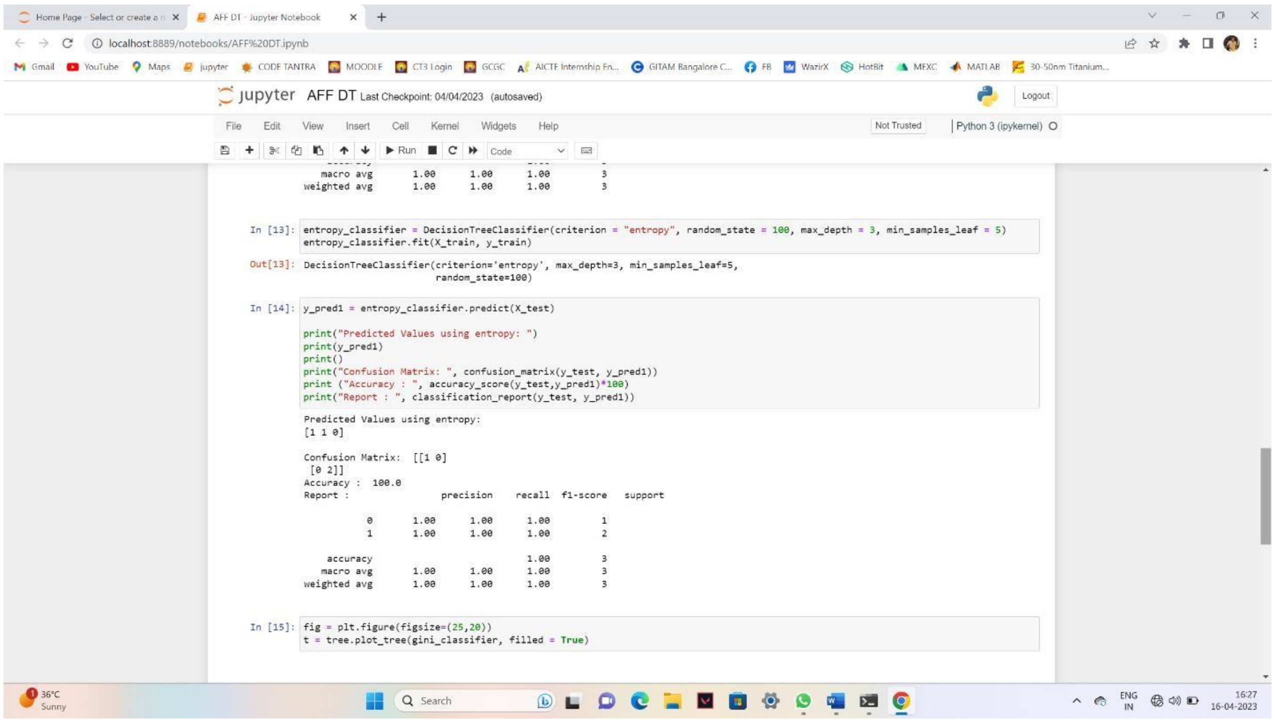
SCREENSHOTS AND OUTPUTS



```
In [1]: import numpy as np
        import pandas as pd
        import seaborn as sns
        import matplotlib.pyplot as plt
        from matplotlib import pyplot as plt
        import sklearn
        from sklearn import tree
        from sklearn.model_selection import train_test_split
        from sklearn.tree import DecisionTreeClassifier
        from sklearn.metrics import confusion_matrix
        from sklearn.metrics import accuracy_score
        from sklearn.metrics import classification_report

In [2]: forest_data = pd.read_csv("AFFA.csv")

In [3]: forest_data = pd.DataFrame(forest_data)

In [4]: forest_data

Out[4]:
     day  month  year  Temperature  Relative_Humidity  Wind_Speed  Rain  Fine_Fuel_Moisture_Code  Druff_Moisture_Code  Drought_Code  Initial_Spread_Index
  0    1      6  2012           32                 71          12   0.7                     57.1                  2.5           8.2                   0.6
  1    2      6  2012           30                 73          13   4.0                     55.7                  2.7           7.8                   0.6
  2    3      6  2012           29                 80          14   2.0                     48.7                  2.2           7.6                   0.3
  3    4      6  2012           30                 64          14   0.0                     79.4                  5.2          15.4                   2.2
  4    5      6  2012           32                 60          14   0.2                     77.1                  6.0          17.6                   1.8
...  ...    ...   ...          ...                ...         ...   ...                      ...                  ...           ...                   ...
117   26      9  2012           30                 65          14   0.0                     85.4                 16.0          44.5                   4.5
118   27      9  2012           28                 87          15   4.4                     41.1                  6.5           8.0                   0.1
119   28      9  2012           27                 87          29   0.5                     45.9                  3.5           7.9                   0.4
120   29      9  2012           24                 54          18   0.1                     79.7                  4.3          15.2                   1.7
121   30      9  2012           24                 64          15   0.2                     67.3                  3.8          16.5                   1.2
```



```
In [5]: forest_data.shape

Out[5]: (122, 14)

In [6]: forest_data.head()

Out[6]:
   day  month  year  Temperature  Relative_Humidity  Wind_Speed  Rain  Fine_Fuel_Moisture_Code  Druff_Moisture_Code  Drought_Code  Initial_Spread_Index  E
0    1      6  2012           32                 71          12   0.7                     57.1                  2.5           8.2                   0.6
1    2      6  2012           30                 73          13   4.0                     55.7                  2.7           7.8                   0.6
2    3      6  2012           29                 80          14   2.0                     48.7                  2.2           7.6                   0.3
3    4      6  2012           30                 64          14   0.0                     79.4                  5.2          15.4                   2.2
4    5      6  2012           32                 60          14   0.2                     77.1                  6.0          17.6                   1.8

In [7]: forest_data = forest_data.drop(['year'], axis=1)

In [8]: forest_data.shape

Out[8]: (122, 13)

In [9]: forest_data.head()

Out[9]:
   day  month  Temperature  Relative_Humidity  Wind_Speed  Rain  Fine_Fuel_Moisture_Code  Druff_Moisture_Code  Drought_Code  Initial_Spread_Index  Buildup
0    1      6           32                 71          12   0.7                     57.1                  2.5           8.2                   0.6
1    2      6           30                 73          13   4.0                     55.7                  2.7           7.8                   0.6
2    3      6           29                 80          14   2.0                     48.7                  2.2           7.6                   0.3
3    4      6           30                 64          14   0.0                     79.4                  5.2          15.4                   2.2
4    5      6           32                 60          14   0.2                     77.1                  6.0          17.6                   1.8

In [10]: X = np.asarray(forest_data[['month', 'Temperature', 'Relative_Humidity', 'Wind_Speed', 'Fine_Fuel_Moisture_Code', 'Druff_Moisture
         Y = np.asarray(forest_data['Output'])
```



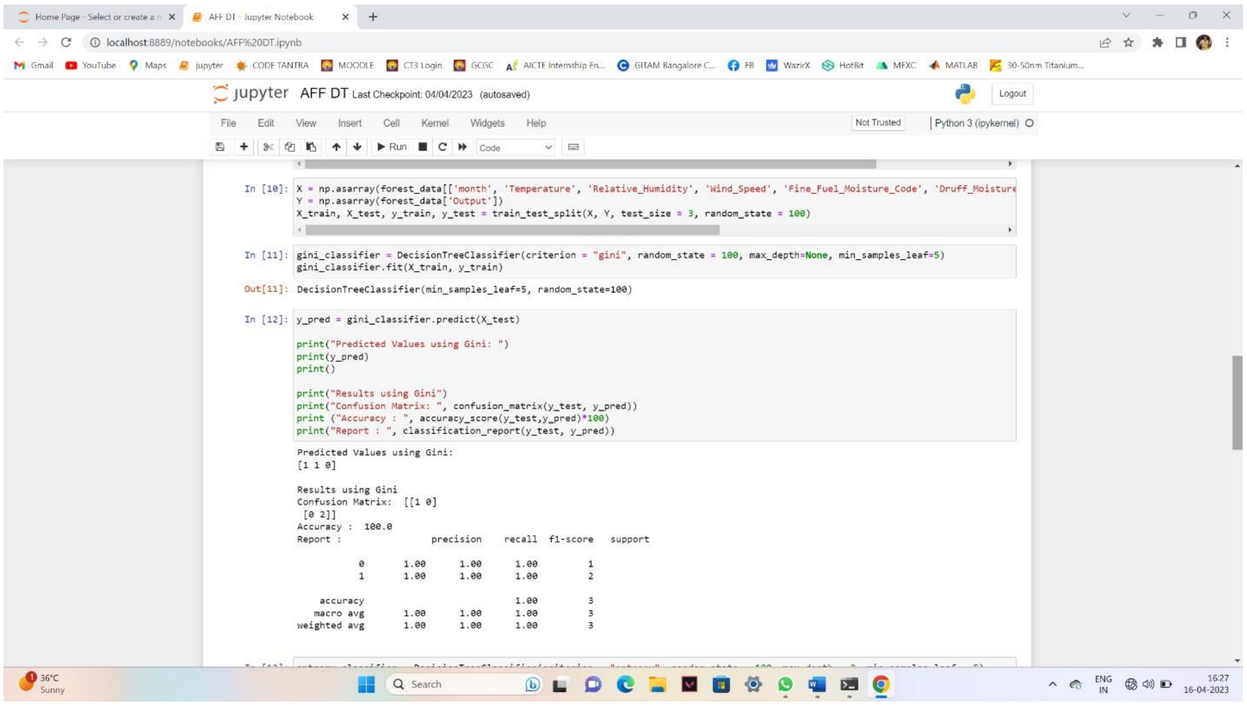
```
   macro avg       1.00      1.00      1.00         3
weighted avg       1.00      1.00      1.00         3
```

```
In [13]: entropy_classifier = DecisionTreeClassifier(criterion = "entropy", random_state = 100, max_depth = 3, min_samples_leaf = 5)
         entropy_classifier.fit(X_train, y_train)

Out[13]: DecisionTreeClassifier(criterion='entropy', max_depth=3, min_samples_leaf=5,
                                random_state=100)
```

```
In [14]: y_pred1 = entropy_classifier.predict(X_test)

         print("Predicted Values using entropy: ")
         print(y_pred1)
         print()
         print("Confusion Matrix: ", confusion_matrix(y_test, y_pred1))
         print ("Accuracy : ", accuracy_score(y_test,y_pred1)*100)
         print("Report : ", classification_report(y_test, y_pred1))
```

```
Predicted Values using entropy:
[1 1 0]

Confusion Matrix:  [[1 0]
 [0 2]]
Accuracy :  100.0
Report :                precision    recall  f1-score   support

           0       1.00      1.00      1.00         1
           1       1.00      1.00      1.00         2

    accuracy                           1.00         3
   macro avg       1.00      1.00      1.00         3
weighted avg       1.00      1.00      1.00         3
```

```
In [15]: fig = plt.figure(figsize=(25,20))
         t = tree.plot_tree(gini_classifier, filled = True)
```

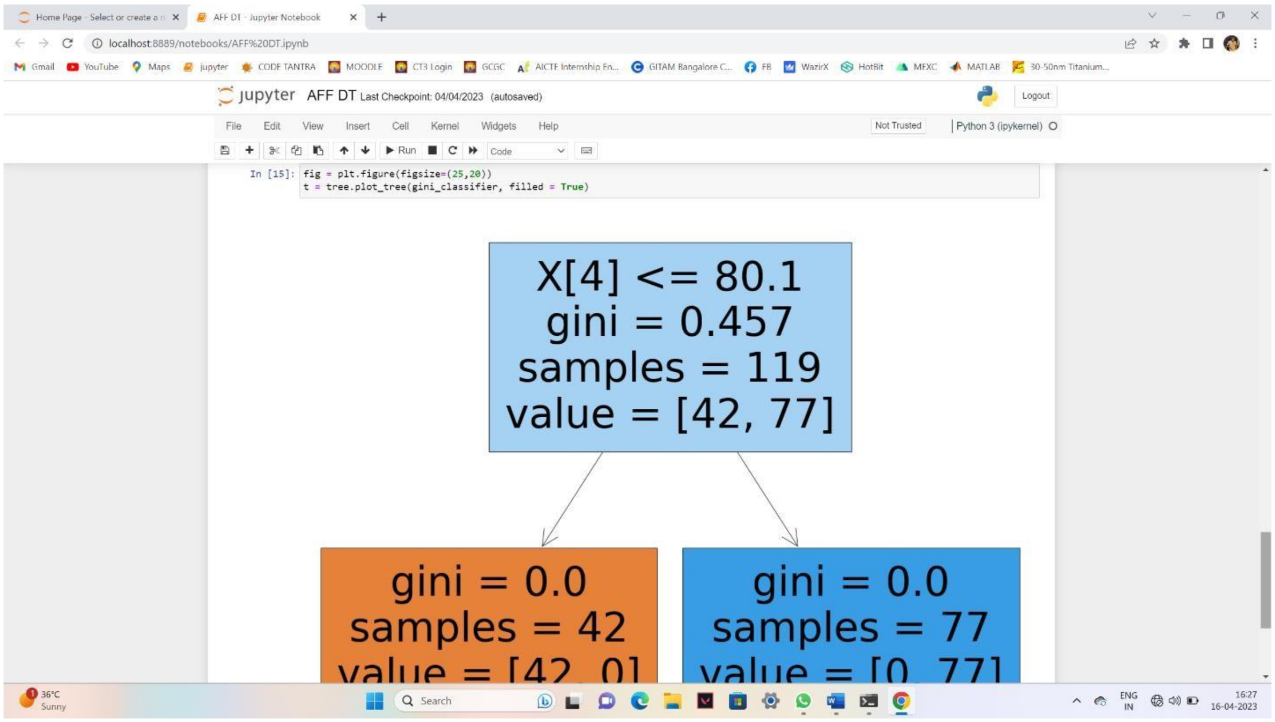


```
In [10]: X = np.asarray(forest_data[['month', 'Temperature', 'Relative_Humidity', 'Wind_Speed', 'Fine_Fuel_Moisture_Code', 'Druff_Moisture
         Y = np.asarray(forest_data['Output'])
         X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 3, random_state = 100)
```

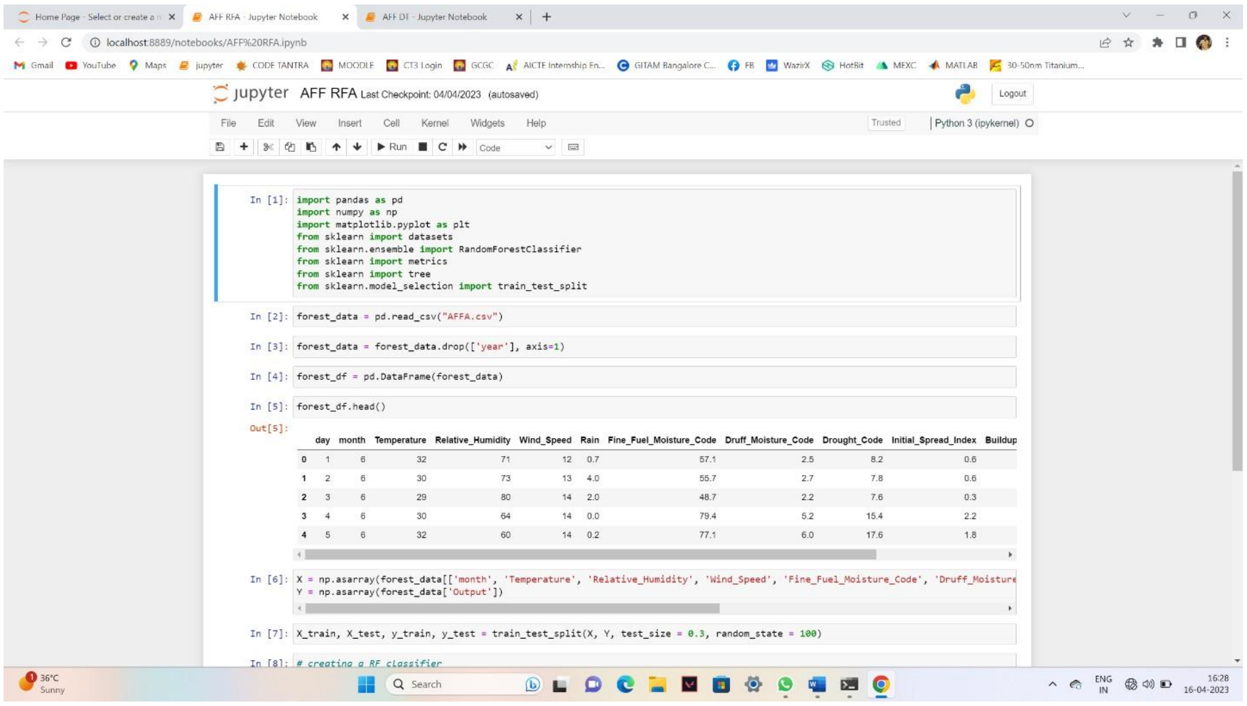
```
In [11]: gini_classifier = DecisionTreeClassifier(criterion = "gini", random_state = 100, max_depth=None, min_samples_leaf=5)
         gini_classifier.fit(X_train, y_train)

Out[11]: DecisionTreeClassifier(min_samples_leaf=5, random_state=100)
```

```
In [12]: y_pred = gini_classifier.predict(X_test)

         print("Predicted Values using Gini: ")
         print(y_pred)
         print()

         print("Results using Gini")
         print("Confusion Matrix: ", confusion_matrix(y_test, y_pred))
         print ("Accuracy : ", accuracy_score(y_test,y_pred)*100)
         print("Report : ", classification_report(y_test, y_pred))
```

```
Predicted Values using Gini:
[1 1 0]

Results using Gini
Confusion Matrix:  [[1 0]
 [0 2]]
Accuracy :  100.0
Report :                precision    recall  f1-score   support

           0       1.00      1.00      1.00         1
           1       1.00      1.00      1.00         2

    accuracy                           1.00         3
   macro avg       1.00      1.00      1.00         3
weighted avg       1.00      1.00      1.00         3
```



```
In [15]: fig = plt.figure(figsize=(25,20))
         t = tree.plot_tree(gini_classifier, filled = True)
```

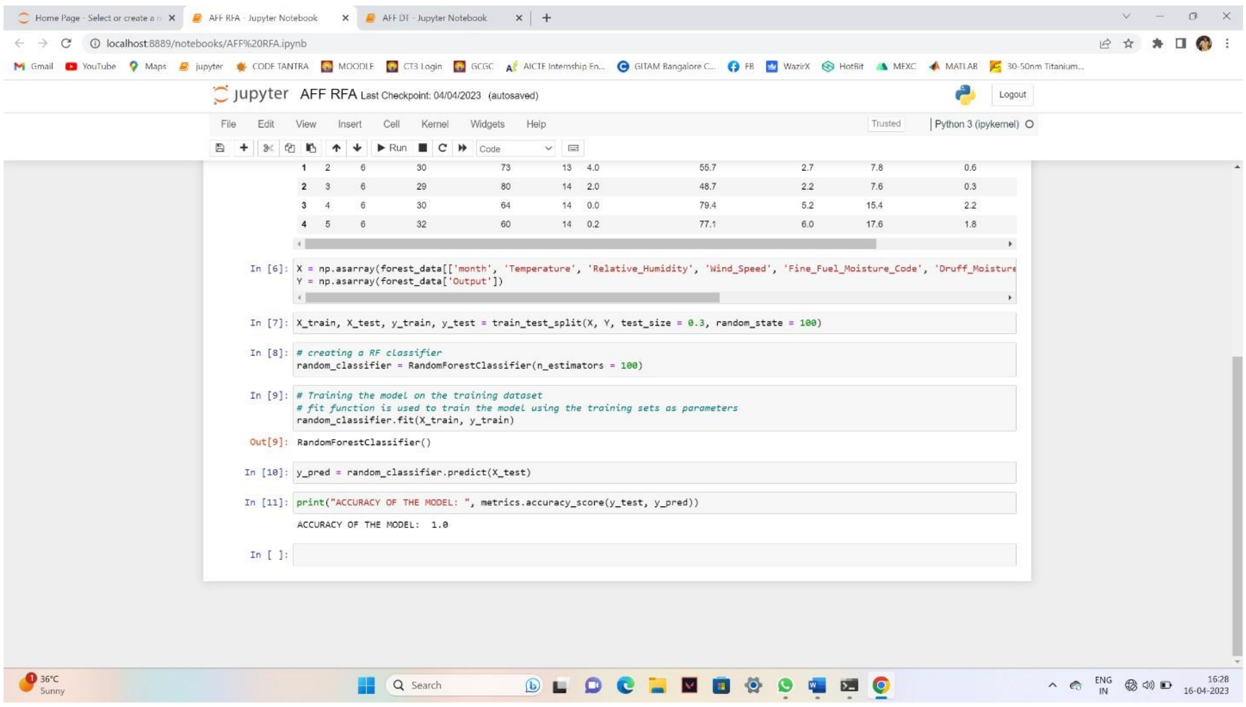


```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        from sklearn import datasets
        from sklearn.ensemble import RandomForestClassifier
        from sklearn import metrics
        from sklearn import tree
        from sklearn.model_selection import train_test_split

In [2]: forest_data = pd.read_csv("AFFA.csv")

In [3]: forest_data = forest_data.drop(['year'], axis=1)

In [4]: forest_df = pd.DataFrame(forest_data)

In [5]: forest_df.head()
```

Out[5]:

	day	month	Temperature	Relative_Humidity	Wind_Speed	Rain	Fine_Fuel_Moisture_Code	Druff_Moisture_Code	Drought_Code	Initial_Spread_Index	Buildup
0	1	6	32	71	12	0.7	57.1	2.5	8.2	0.6	
1	2	6	30	73	13	4.0	55.7	2.7	7.8	0.6	
2	3	6	29	80	14	2.0	48.7	2.2	7.6	0.3	
3	4	6	30	64	14	0.0	79.4	5.2	15.4	2.2	
4	5	6	32	60	14	0.2	77.1	6.0	17.6	1.8	

```
In [6]: X = np.asarray(forest_data[['month', 'Temperature', 'Relative_Humidity', 'Wind_Speed', 'Fine_Fuel_Moisture_Code', 'Druff_Moisture
        Y = np.asarray(forest_data['Output'])

In [7]: X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.3, random_state = 100)

In [8]: # creating a RF classifier
```



jupyter AFF RFA Last Checkpoint: 04/04/2023 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code

1	2	6	30	73	13	4.0	55.7	2.7	7.8	0.6
2	3	6	29	80	14	2.0	48.7	2.2	7.6	0.3
3	4	6	30	64	14	0.0	79.4	5.2	15.4	2.2
4	5	6	32	60	14	0.2	77.1	6.0	17.6	1.8

```
In [6]: X = np.asarray(forest_data[['month', 'Temperature', 'Relative_Humidity', 'Wind_Speed', 'Fine_Fuel_Moisture_Code', 'Druff_Moisture
        Y = np.asarray(forest_data['Output'])

In [7]: X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.3, random_state = 100)

In [8]: # creating a RF classifier
        random_classifier = RandomForestClassifier(n_estimators = 100)

In [9]: # Training the model on the training dataset
        # fit function is used to train the model using the training sets as parameters
        random_classifier.fit(X_train, y_train)

Out[9]: RandomForestClassifier()

In [10]: y_pred = random_classifier.predict(X_test)

In [11]: print("ACCURACY OF THE MODEL: ", metrics.accuracy_score(y_test, y_pred))

         ACCURACY OF THE MODEL:  1.0

In [ ]:
```

SOURCE CODE

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib import pyplot as plt
import sklearn
from sklearn import tree
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
from sklearn.model_selection import train_test_split
forest_data = pd.read_csv("AFFA.csv")
forest_data = pd.DataFrame(forest_data)
forest_data
forest_data.shape
forest_data.head()
forest_data = forest_data.drop(['year'], axis=1)
forest_data.shape
forest_data.head()
X = np.asarray(forest_data[['month', 'Temperature', 'Relative_Humidity', 'Wind_Speed',
'Fine_Fuel_Moisture_Code', 'Druff_Moisture_Code', 'Drought_Code',
'Initial_Spread_Index', 'Buildup_Index', 'Fire_Weather_Index']])
Y = np.asarray(forest_data['Output'])
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 3, random_state = 100)
gini_classifier = DecisionTreeClassifier(criterion = "gini", random_state = 100,
max_depth=None, min_samples_leaf=5)
gini_classifier.fit(X_train, y_train)
y_pred = gini_classifier.predict(X_test)
```

```
print("Predicted Values using Gini: ")
print(y_pred)
print()

print("Results using Gini")
print("Confusion Matrix: ", confusion_matrix(y_test, y_pred))
print ("Accuracy : ", accuracy_score(y_test,y_pred)*100)
print("Report : ", classification_report(y_test, y_pred))
entropy_classifier = DecisionTreeClassifier(criterion = "entropy", random_state = 100,
max_depth = 3, min_samples_leaf = 5)
entropy_classifier.fit(X_train, y_train)
y_pred1 = entropy_classifier.predict(X_test)

print("Predicted Values using entropy: ")
print(y_pred1)
print()
print("Confusion Matrix: ", confusion_matrix(y_test, y_pred1))
print ("Accuracy : ", accuracy_score(y_test,y_pred1)*100)
print("Report : ", classification_report(y_test, y_pred1))
fig = plt.figure(figsize=(25,20))
t = tree.plot_tree(gini_classifier, filled = True)
X = np.asarray(forest_data[['month', 'Temperature', 'Relative_Humidity', 'Wind_Speed',
'Fine_Fuel_Moisture_Code', 'Druff_Moisture_Code', 'Drought_Code',
'Initial_Spread_Index', 'Buildup_Index', 'Fire_Weather_Index']])
Y = np.asarray(forest_data['Output'])
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.3, random_state =
100)
random_classifier = RandomForestClassifier(n_estimators = 100)
random_classifier.fit(X_train, y_train)
y_pred = random_classifier.predict(X_test)
print("ACCURACY OF THE MODEL: ", metrics.accuracy_score(y_test, y_pred))
```

------THE END-----