CROP RECOMMENDATION USING MACHINE LEARNING

Group Two :

-Dilnawaz Ahmed (28954320011)
-Ashutosh Kumar Ankit (28954320013)

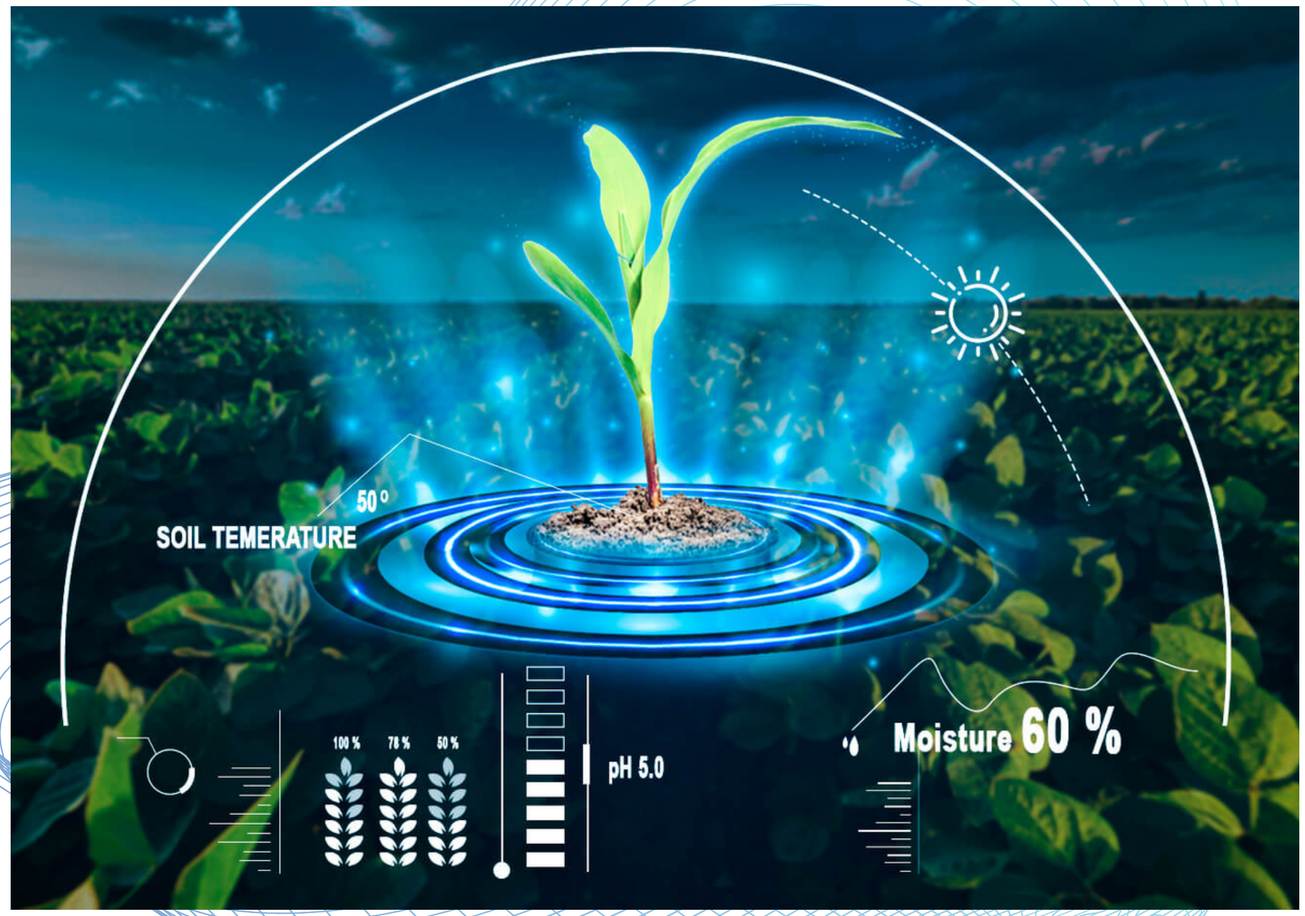
Subject : Major Project(BSCDA 602)

TABLE OF CONTENT

INTRODUCTION

The project is about Precision Farming which involves strategic ways of guiding farmers in crop rotation, optimal planting or harvesting times and soil management to improve crop productivity and efficiency while redcucing environmemtal impact.

An Intelligent Crop Recommendation system using Machine Learning that predicts crop suitability by factoring all relevant data such as temperature, rainfall, location, and soil condition.



OBJECTIVE

Problem Statement: Build a Predictive Model so as to suggest the most suitable Crops to grow based on the available Climatic and Soil conditions.

Goal: Achieve Precision Farming by Optimising the Agricultural Production

DATASET DESCRIPTION

The Dataset consist of a single CSV file. The dataset consist of 2200 rows and 8 columns. There are in total seven input features and target variable are as follows:

N - Ratio of Nitrogen content in the soil.

P - Ratio of Phosphorous content in the soil.

K - Ratio of Potassium content in the soil.

Temperature - The temperature in degrees Celsius.

Humidity - Relative humidity in %.

pH - It is the pH value of the soil.

Rainfall - Rainfall in mm.

Labels - Rice, Maize, Jute, Cotton, Coconut, Papaya, Orange, Apple, Muskmelon, Watermelon, Grapes, Mango, Banana, Pomegranate, Lentils, Blackgram, Mungbean, Mothbeans, Pigeonpeas, Kidneybeans, Chickpea, Coffee.

OBSERVATION USING TOOLS AND SOFTWARE

Descriptive Statistics is a means of describing features of a dataset by generating summaries about data samples that are used to describe the measures like central tendency and measures of dispersion. This table gives information about count mean, std, min, 25,50 and 75 percentile and max of input features.

In [11]:
```
# Let's do the descriptive statistics of the dataset
df.describe()
```

Out[11]:

	N	P	K	temperature	humidity	ph	rainfall
count	2200.000000	2200.000000	2200.000000	2200.000000	2200.000000	2200.000000	2200.000000
mean	50.286364	53.362727	48.149091	25.613182	71.670491	6.495000	103.458182
std	37.037324	32.985883	50.647931	5.080069	22.010161	0.786408	54.964423
min	0.000000	5.000000	5.000000	9.000000	14.000000	4.000000	20.000000
25%	21.000000	28.000000	20.000000	23.000000	60.000000	6.000000	65.000000
50%	37.000000	51.000000	32.000000	26.000000	80.000000	6.000000	95.000000
75%	84.000000	68.000000	49.000000	29.000000	90.000000	7.000000	124.000000
max	140.000000	145.000000	205.000000	44.000000	100.000000	10.000000	299.000000

A statistical distribution, describes how values are distributed for a field shows where the data lies and gives which values are common and uncommon.
There are many kinds of statistical distributions, including the bell-shaped normal distribution.

The following observations are taken from the distribution chart:

- There are some crops which require very high proportion of phosphorous and potassium because the distribution chart is very skewed.

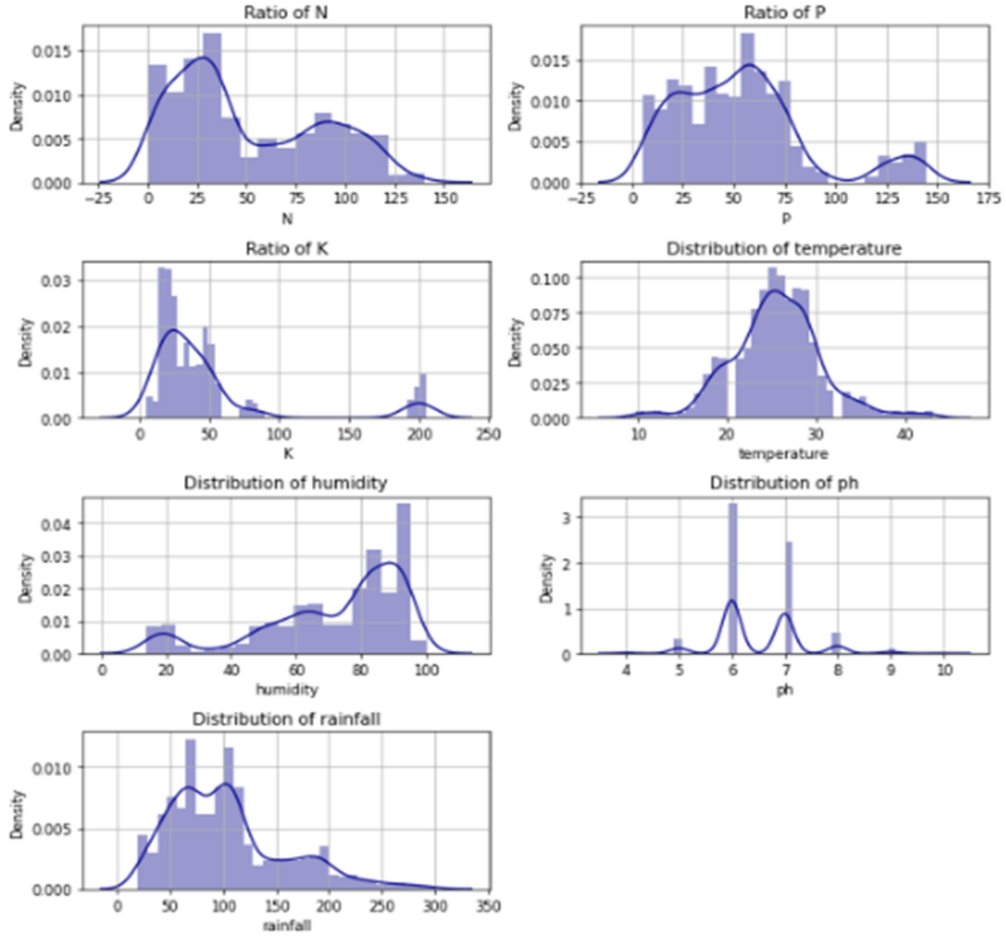
- There are some crops which requires very low and very high temperature .

- There are some crops which requires low or high pH in the soil.

In [12]:
```
plt.rcParams['figure.figsize'] = (10,10)
plt.rcParams['figure.dpi'] = 60

features = ['N','P','K','temperature','humidity','ph','rainfall']

for i, feat in enumerate(features):
    plt.subplot(4,2,i + 1)
    sns.distplot(df[feat],color='darkblue')
    if i < 3:
        plt.title(f'Ratio of {feat}',fontsize = 12)
    else:
        plt.title(f'Distribution of {feat}',fontsize = 12)
    plt.tight_layout()
    plt.grid()
```

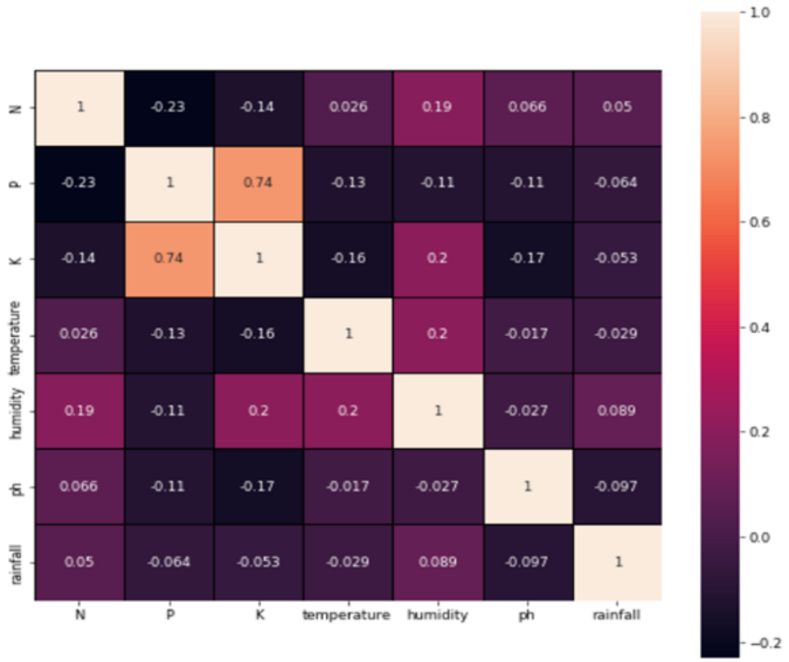


Correlation Matrix between features

A correlation matrix is simply a table showing the correlation coefficients between variables. e.g.
Height vs. Weight are examples of positive correlation.
Time Spent Watching TV vs. Exam Scores are examples of negative correlation.
Coffee Consumption vs. Intelligence and Shoe Size vs. Movies Watched are examples of no correlation.

```
In [13]: # Let's check the correlation between the variables
         sns.heatmap(df.corr(),annot=True,cbar=True,linewidth=1,linecolor='black',square=True)

Out[13]: <AxesSubplot:>
```



Inference from the above heatmap is that apart from K vs. P, there are no two highly correlated features. The correlation between K vs. P is +0.74.

That means as the requirement of Potassium increases with the increase in Phosphorous and as the requirement of Potassium decreases with the decrease in Phosphorous.

In [15]:
```
# Let's chech the Crop Summary I
df.groupby(['label'])[['N','P','K']].agg(['min','mean','max'])
```

Out[15]:

	N			P			K		
	min	mean	max	min	mean	max	min	mean	max
label									
apple	0.0	20.80	40.0	120	134.22	145	195	199.89	205
banana	80.0	100.23	120.0	70	82.01	95	45	50.05	55
blackgram	20.0	40.02	60.0	55	67.47	80	15	19.24	25
chickpea	20.0	40.09	60.0	55	67.79	80	75	79.92	85
coconut	0.0	21.98	40.0	5	16.93	30	25	30.59	35
coffee	80.0	101.20	120.0	15	28.74	40	25	29.94	35
cotton	100.0	117.77	140.0	35	46.24	60	15	19.56	25
grapes	0.0	23.18	40.0	120	132.53	145	195	200.11	205
jute	60.0	78.40	100.0	35	46.86	60	35	39.99	45
kidneybeans	0.0	20.63	40.0	55	67.54	80	15	20.05	25
lentil	0.0	18.77	40.0	55	68.36	80	15	19.41	25
maize	0.0	76.09	99.0	35	48.44	60	15	19.79	25
mango	0.0	20.07	40.0	15	27.18	40	25	29.92	35
mothbeans	0.0	21.44	40.0	35	48.01	60	15	20.23	25
mungbean	0.0	20.99	40.0	35	47.28	60	15	19.87	25
muskmelon	80.0	100.32	120.0	5	17.72	30	45	50.08	55
orange	0.0	19.58	40.0	5	16.55	30	5	10.01	15
papaya	31.0	49.88	70.0	46	59.05	70	45	50.04	55
pigeonpeas	0.0	19.12	40.0	55	67.73	80	15	20.29	25
pomegranate	0.0	18.87	40.0	5	18.75	30	35	40.21	45
rice	0.0	77.45	99.0	35	47.58	60	35	39.87	45
watermelon	80.0	99.42	120.0	5	17.00	30	45	50.22	55

[16]:
```
# Let's chech the Crop Summary II
df.groupby(['label'])[['humidity','ph','rainfall','temperature']].agg(['min','mean','max'])
```

t[16]:

	humidity			ph			rainfall			temperature		
	min	mean	max	min	mean	max	min	mean	max	min	mean	max
label												
apple	90.000000	92.360000	95.000000	6	6.00	6	100	112.71	125	21	22.61	24
banana	75.000000	80.330000	85.000000	6	6.00	6	90	104.57	120	25	27.33	30
blackgram	60.000000	65.150000	70.000000	7	7.20	8	60	67.84	75	25	29.96	35
chickpea	14.000000	20.676934	71.670491	6	7.33	9	65	80.14	95	17	18.81	21
coconut	90.000000	94.840000	100.000000	6	6.00	6	131	175.72	226	25	27.45	30
coffee	50.000000	58.850000	70.000000	6	6.73	7	115	158.06	199	23	25.56	28
cotton	75.000000	79.880000	85.000000	6	6.88	8	61	80.42	100	22	23.99	26
grapes	80.000000	81.930000	84.000000	6	6.00	6	65	69.60	75	9	23.85	42
jute	71.000000	79.620000	90.000000	6	6.65	7	150	174.76	200	23	24.92	27
kidneybeans	18.000000	22.076705	71.670491	6	6.00	6	60	105.90	150	15	20.14	25
lentil	60.000000	64.780000	70.000000	6	6.93	8	35	45.64	55	18	24.49	30
maize	55.000000	65.360115	75.000000	6	6.32	7	61	84.77	110	18	22.42	27
mango	45.000000	50.160000	55.000000	5	5.75	7	89	94.69	101	27	31.20	36
mothbeans	40.000000	53.190000	65.000000	4	6.86	10	31	51.21	74	24	28.17	32
mungbean	80.000000	85.500000	90.000000	6	6.72	7	36	48.39	60	27	28.54	30
muskmelon	90.000000	92.370000	95.000000	6	6.29	7	20	24.66	30	27	28.70	30
orange	90.000000	92.170000	95.000000	6	7.00	8	100	110.44	120	10	22.77	35
papaya	90.000000	92.410000	95.000000	7	7.00	7	40	142.63	249	23	33.71	44
pigeonpeas	30.000000	48.413410	71.670491	5	5.84	7	90	149.46	199	18	27.74	37
pomegranate	85.000000	90.100000	95.000000	6	6.48	7	103	107.52	112	18	21.86	25
rice	71.670491	81.433639	85.000000	5	6.43	8	183	236.19	299	20	23.70	27
watermelon	80.000000	85.150000	90.000000	6	6.48	7	40	50.76	60	24	25.57	27

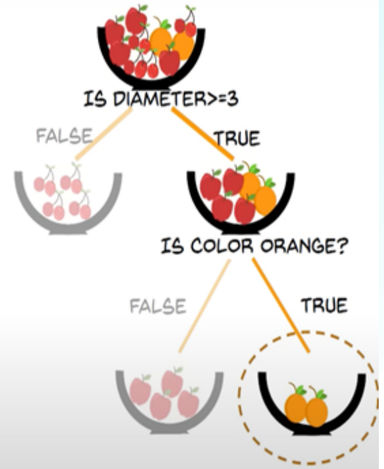
This crop summary give details about min, mean and max for each crops

METHODOLOGY

The algorithm used is Random Forest which is a supervised Machine Learning algorithm especially used for classification problem.
It works on the principle of combining the output of multiple decision trees to reach to a single output.
It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

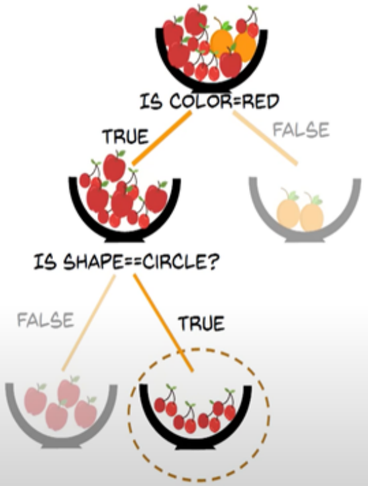
1. Select random samples from a given data or training set.
2. This algorithm will construct a decision tree for every training data.
3. Voting will take place by averaging the decision tree.
4. Finally, select the most voted prediction result as the final prediction result.

TREE 1 CLASSIFIES
IT AS AN ORANGE



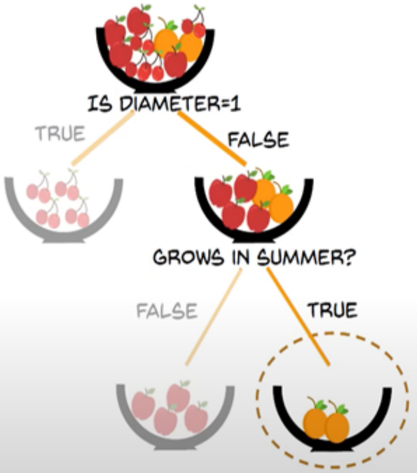
DIAMETER = 3
COLOUR = ORANGE
GROWS IN SUMMER = YES
SHAPE = CIRCLE

TREE 2 CLASSIFIES
IT AS CHERRIES

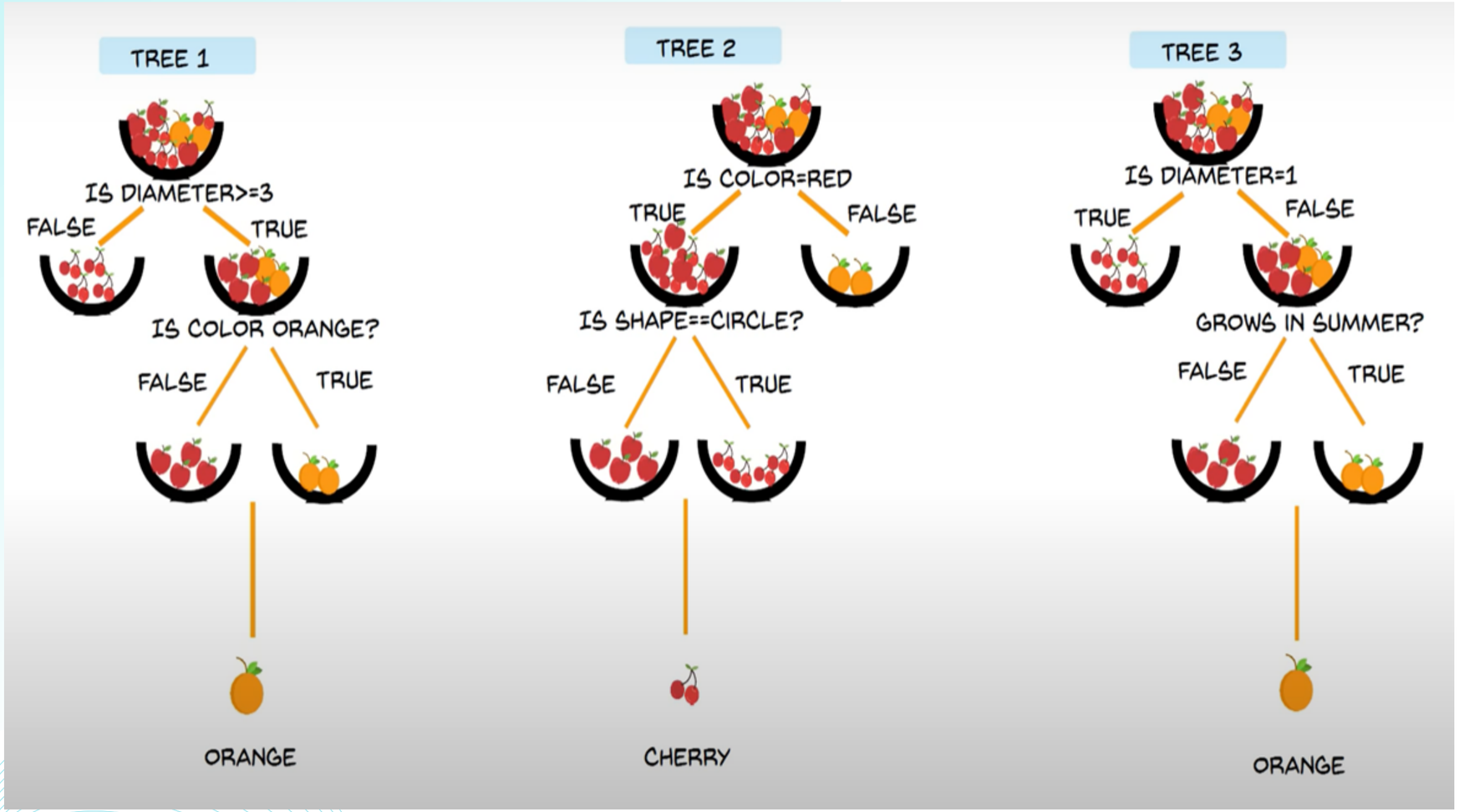


DIAMETER = 3
COLOUR = ORANGE
GROWS IN SUMMER = YES
SHAPE = CIRCLE

TREE 3 CLASSIFIES
IT AS ORANGE



DIAMETER = 3
COLOUR = ORANGE
GROWS IN SUMMER = YES
SHAPE = CIRCLE


TREE 1
IS DIAMETER>=3
FALSE
TRUE
IS COLOR ORANGE?
FALSE
TRUE
ORANGE
TREE 2
IS COLOR=RED
TRUE
FALSE
IS SHAPE==CIRCLE?
FALSE
TRUE
CHERRY
TREE 3
IS DIAMETER=1
TRUE
FALSE
GROWS IN SUMMER?
FALSE
TRUE
ORANGE

```
In [19]: # Let's split our dataset into training and testing dataset
         from sklearn.model_selection import train_test_split
         X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

The next step in model training is splitting the data into training and testing sets. The training set is where the model sees the input and predict the outputs.
To perform the split, we will use the built-in function train_test_split from the sklearn library. In this project, we will use 80% of the data for training and the remaining 20% for testing.
1760 rows goes into the training phase and 440 rows goes into the testing phase.

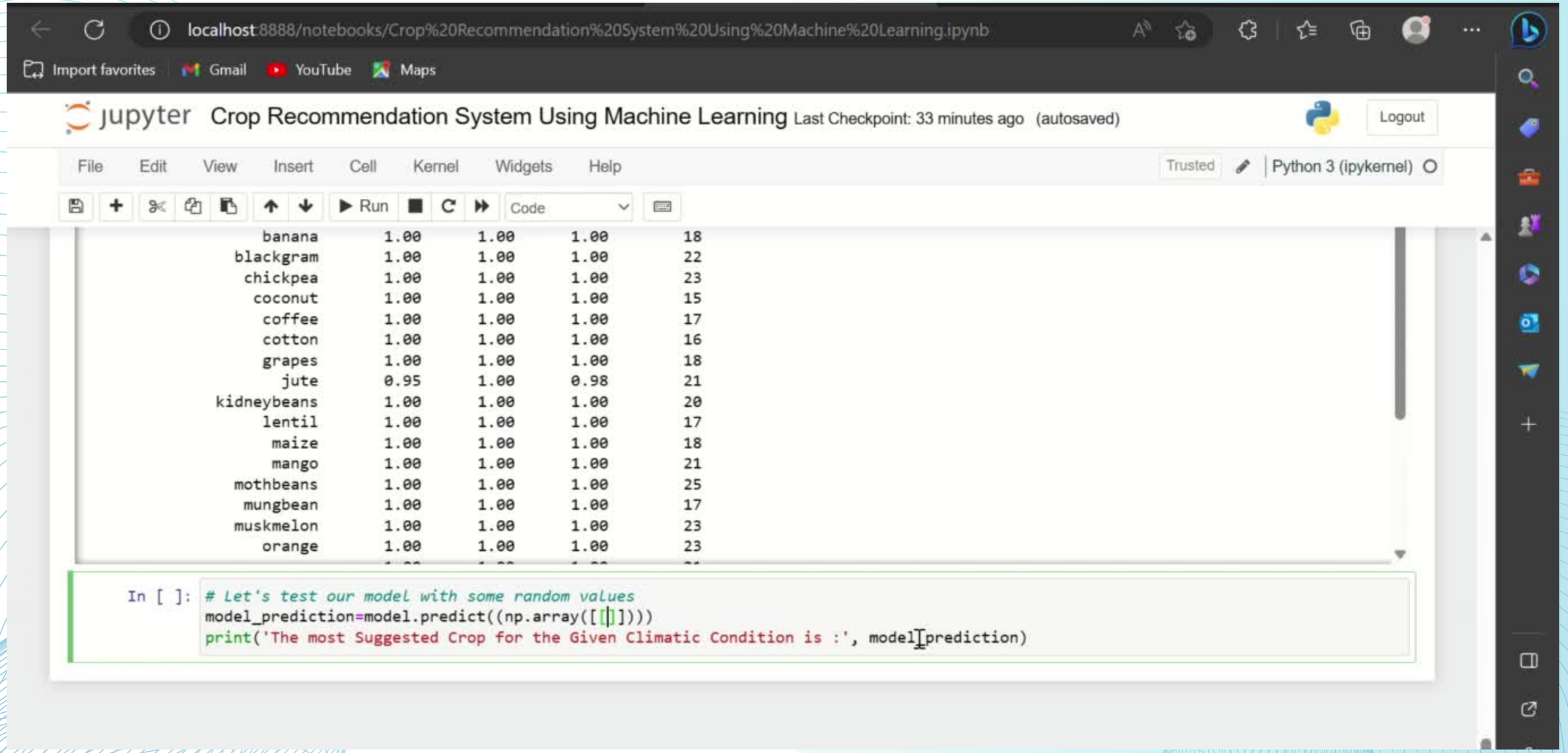
```
In [24]: # Importing Random Forest algorithm from sklearn
         from sklearn.ensemble import RandomForestClassifier
         # Let's train our model
         model=RandomForestClassifier()
         model.fit(X_train,y_train)

Out[24]: RandomForestClassifier()
```

Importing the Random Forest algorithm from sklearn and building the model.

- This is the actual step where the model is being trained over training dataset
- Based on this training the model will know how to predict the values in testing dataset

EXPECTED OUTCOME



```
In [28]: # Lets's calculate the score of our model
         model.score(X_test,y_test)*100

Out[28]: 99.31818181818181
```

The score is calculated between the X_test and y_test. It works on the principal that X_test is given to the model and predicts the values and these values are then compared with the actual y_test which was not given to the model during actual training. The score function is used to accomplish this task.
The score of the model is found to be 99.31

1

Process 01

We are entering the values for Nitrogen, Phosphorous, Potassium, Temperature, Humidity, pH , Rainfall.

2

Process 02

The respective values are [18, 20, 26, 31, 52, 6, 90]

3

Process 03

The outcome generated by the Random Forest Model based on these values is mango which is accurate with the testing dataset .

Confusion Matrix

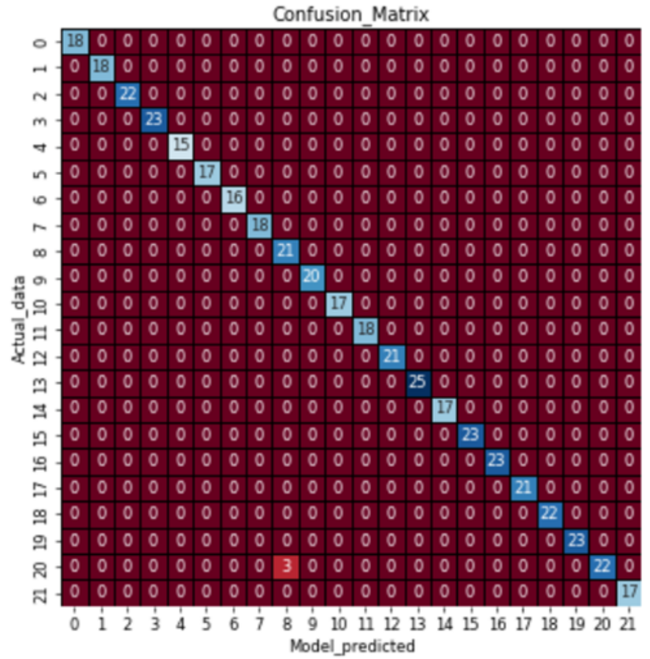
18 times the actual label was Apple and the model predicted it as Apple and so on.

3 times the actual label was Rice and the model predicted it as Jute.

This is the place where our model lost the accuracy of 0.69%

```
In [29]: # for creating confusion matrix
         from sklearn.metrics import confusion_matrix
         # Let's check where our Model fails
         plt.figure(figsize=(7,7))
         cm=confusion_matrix(y_test,y_predicted)
         sns.heatmap(cm,annot=True,cbar=False,linewidth=1,linecolor='black',cmap='RdBu',square=True)
         plt.xlabel('Model_predicted')
         plt.ylabel('Actual_data')
         plt.title('Confusion_Matrix')
```

```
Out[29]: Text(0.5, 1.0, 'Confusion_Matrix')
```



CONCLUSION

The Solution will benefit farmer to maximize productivity in agriculture, reduce soil degradation in cultivated fields, and reduce fertilizer use in crop production by recommending the right crop by considering various attributes.

The model's expectations for the predicted crop yield were all significantly impacted by these variables, proving that the original assumption about them was accurate. In conclusion, crop yield prediction using machine learning has the potential to revolutionize the agriculture industry.

REFERENCES

https://www.kaggle.com/datasets/atharvaingle/crop-recommendation-dataset

https://www.geeksforgeeks.org/crop-recommendation-system-using-tensorflow/

https://ieeexplore.ieee.org/document/9418351

https://www.ijert.org/crop-recommendation-using-machine-learning-techniques

https://towardsdatascience.com/farmeasy-crop-recommendation-portal-for-farmers-48a8809b421c

https://www.javatpoint.com/crop-recommendation-system-using-tensorflow