MACHINE LEARNING ASSIGNMENT – 2

1. Develop Logistic regression on prediction of Crop-Recommendation dataset.

Data Set Description (No of features-Continous/Discrete) – DISCRETE VALUES

About Dataset: Precision agriculture is in trend nowadays. It helps the farmers to get informed decision about the farming strategy. Here, I present you a dataset which would allow the users to build a predictive model to recommend the most suitable crops to grow in a particular farm based on various parameters.

This dataset was build by augmenting datasets of rainfall, climate and fertilizer data available for India.

Data fields

- N - ratio of Nitrogen content in soil
- P - ratio of Phosphorous content in soil
- K - ratio of Potassium content in soil
- temperature - temperature in degree Celsius
- humidity - relative humidity in %
- ph - ph value of the soil
- rainfall - rainfall in mm

1) Principal Component Analysis (PCA):

Principal Component Analysis(PCA) technique was introduced by the mathematician Karl Pearson in 1901. It works on the condition that while the data in a higher dimensional space is mapped to data in a lower dimension space, the variance of the data in the lower dimensional space should be maximum.

- Principal Component Analysis (PCA) is a statistical procedure that uses an orthogonal transformation that converts a set of correlated variables to a set of uncorrelated variables. PCA is the most widely used tool in exploratory data analysis and in machine learning for predictive models.

- Principal Component Analysis (PCA) is an unsupervised learning algorithm technique used to examine the interrelations among a set of variables. It is also known as a general factor analysis where regression determines a line of best fit.

- The main goal of Principal Component Analysis (PCA) is to reduce the dimensionality of a dataset while preserving the most important patterns or relationships between the variables without any prior knowledge of the target variables.

PROGRAM :

1. **Import Packages**

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.decomposition import PCA
df=pd.read_csv("Crop.csv")
df
```

2. **Data Pre-processing**

```
df.describe()
df.info()
df.isnull().sum()

values, counts = np.unique(df['label'], return_counts=True)
print('number of unique labels is %i\n'% len(values))
for (i, j) in zip(values, counts):
    print('number of %s Labels in dataset is %i'% (i, j))

X=df.iloc[:, :-1]
y=df.iloc[:, -1]
print(X)
print(y)

from sklearn.preprocessing import StandardScaler
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)
scaler=StandardScaler()
X_train_scaled=scaler.fit_transform(X_train)
X_test_scaled=scaler.fit_transform(X_test)
```

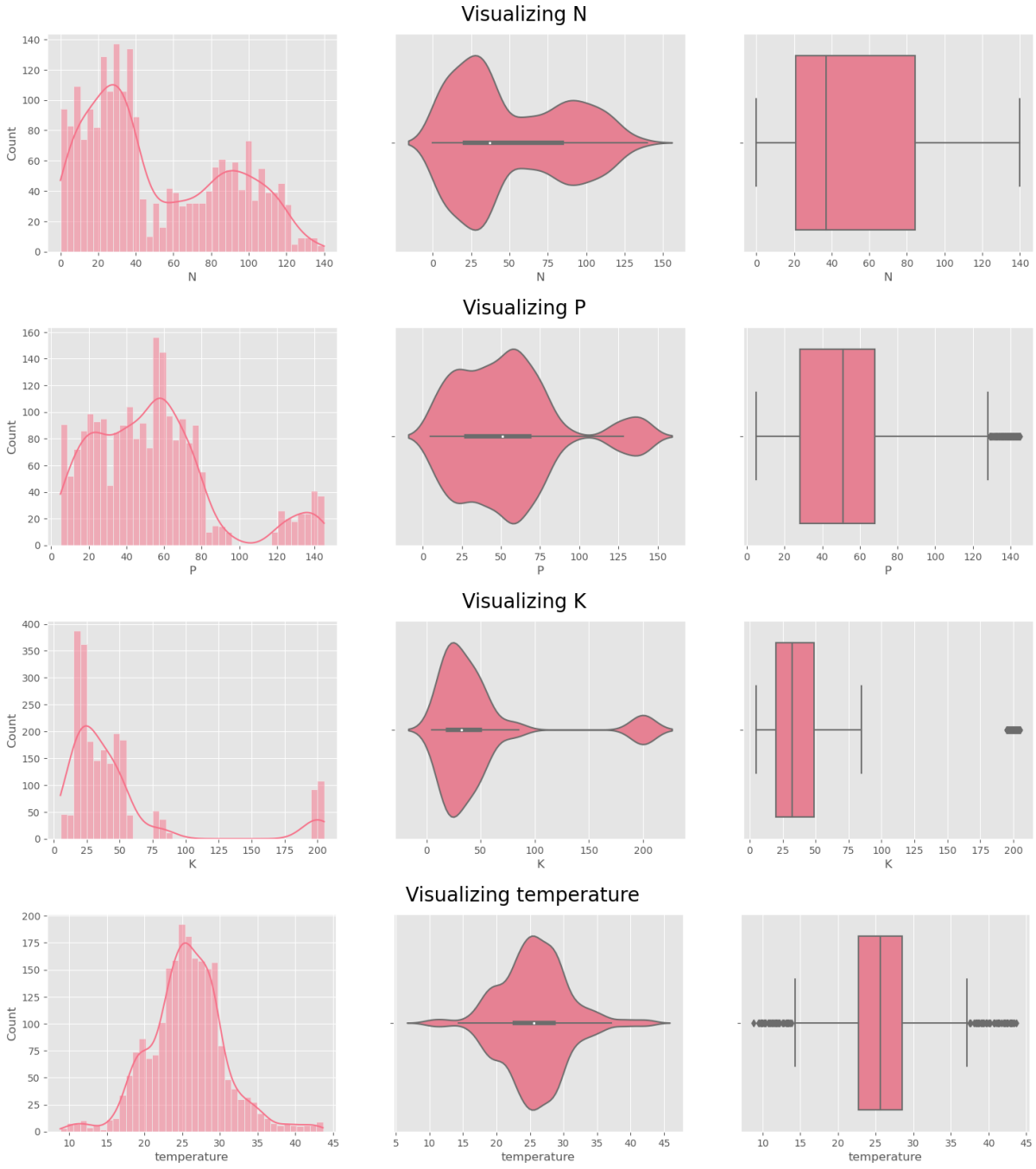
3. **Logistic Regression Model**

```
num_components=6
pca=PCA(n_components=num_components)
X_train_pca=pca.fit_transform(X_train_scaled)
X_test_pca=pca.fit_transform(X_test_scaled)
model = LogisticRegression()
model.fit(X_train_pca, y_train)
y_pred = model.predict(X_test_pca)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)

import matplotlib.pyplot as plt
```

```
import seaborn as sns
plt.style.use('ggplot')
sns.set_palette("husl")
for i in df.columns[:-1]:
    fig,ax=plt.subplots(1,3,figsize=(18,4))
    sns.histplot(data=df,x=i,kde=True,bins=40,ax=ax[0])
    sns.violinplot(data=df,x=i,ax=ax[1])
    sns.boxplot(data=df,x=i,ax=ax[2])
    plt.suptitle(f'Visualizing {i}',size=20)
```

Output:



```
grouped = df.groupby(by='label').mean().reset_index()
grouped
```

output:

	label	N	P	K	temperature	humidity	ph	rainfall
0	apple	20.80	134.22	199.89	22.630942	92.333383	5.929663	112.654779
1	banana	100.23	82.01	50.05	27.376798	80.358123	5.983893	104.626980
2	blackgram	40.02	67.47	19.24	29.973340	65.118426	7.133952	67.884151
3	chickpea	40.09	67.79	79.92	18.872847	16.860439	7.336957	80.058977
4	coconut	21.98	16.93	30.59	27.409892	94.844272	5.976562	175.686646
5	coffee	101.20	28.74	29.94	25.540477	58.869846	6.790308	158.066295
6	cotton	117.77	46.24	19.56	23.988958	79.843474	6.912675	80.398043
7	grapes	23.18	132.53	200.11	23.849575	81.875228	6.025937	69.611829
8	jute	78.40	46.86	39.99	24.958376	79.639864	6.732778	174.792798
9	kidneybeans	20.75	67.54	20.05	20.115085	21.605357	5.749411	105.919778
10	lentil	18.77	68.36	19.41	24.509052	64.804785	6.927932	45.680454
11	maize	77.76	48.44	19.79	22.389204	65.092249	6.245190	84.766988
12	mango	20.07	27.18	29.92	31.208770	50.156573	5.766373	94.704515
13	mothbeans	21.44	48.01	20.23	28.194920	53.160418	6.831174	51.198487
14	mungbean	20.99	47.28	19.87	28.525775	85.499975	6.723957	48.403601
15	muskmelon	100.32	17.72	50.08	28.663066	92.342802	6.358805	24.689952
16	orange	19.58	16.55	10.01	22.765725	92.170209	7.016957	110.474969
17	papaya	49.88	59.05	50.04	33.723859	92.403388	6.741442	142.627839
18	pigeonpeas	20.73	67.73	20.29	27.741762	48.061633	5.794175	149.457564
19	pomegranate	18.87	18.75	40.21	21.837842	90.125504	6.429172	107.528442
20	rice	79.89	47.58	39.87	23.689332	82.272822	6.425471	236.181114
21	watermelon	99.42	17.00	50.22	25.591767	85.160375	6.495778	50.786219

```
df.corr()
```

	N	P	K	temperature	humidity	ph	rainfall
N	1.000000	-0.231460	-0.140512	0.026504	0.190688	0.096683	0.059020
P	-0.231460	1.000000	0.736232	-0.127541	-0.118734	-0.138019	-0.063839
K	-0.140512	0.736232	1.000000	-0.160387	0.190859	-0.169503	-0.053461
temperature	0.026504	-0.127541	-0.160387	1.000000	0.205320	-0.017795	-0.030084
humidity	0.190688	-0.118734	0.190859	0.205320	1.000000	-0.008483	0.094423
ph	0.096683	-0.138019	-0.169503	-0.017795	-0.008483	1.000000	-0.109069
rainfall	0.059020	-0.063839	-0.053461	-0.030084	0.094423	-0.109069	1.000000

```
figure = plt.figure(figsize=(12, 6))
sns.heatmap(df.corr(),annot=True)
```

Output:-

```
<Axes: >
```


```
from sklearn.decomposition import PCA
import plotly.express as px
pca=PCA(n_components=2)
df_pca=pca.fit_transform(df.drop(['label'],axis=1))
df_pca=pd.DataFrame(df_pca)
fig = px.scatter(x=df_pca[0],y=df_pca[1],color=df['label'],title="Decomposed using PCA")
fig.show()
```

Output:-

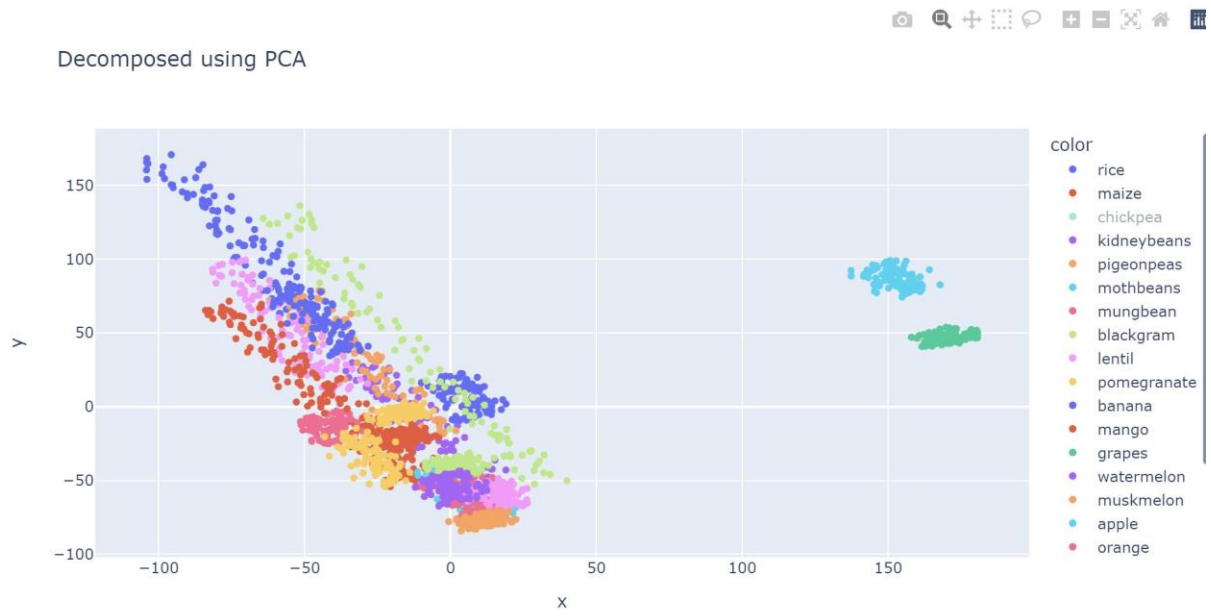


fig = px.scatter(x=df['N'],y=df['P'],color=df['label'],title="Nitrogen VS Phosphorus")

fig.show()

Output:-

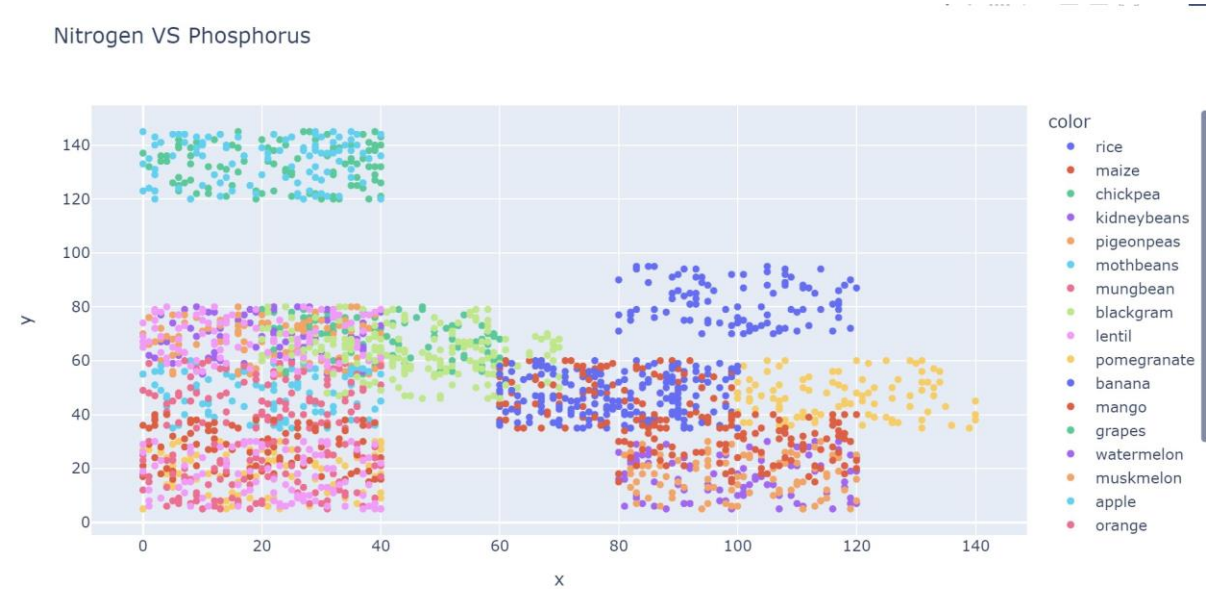
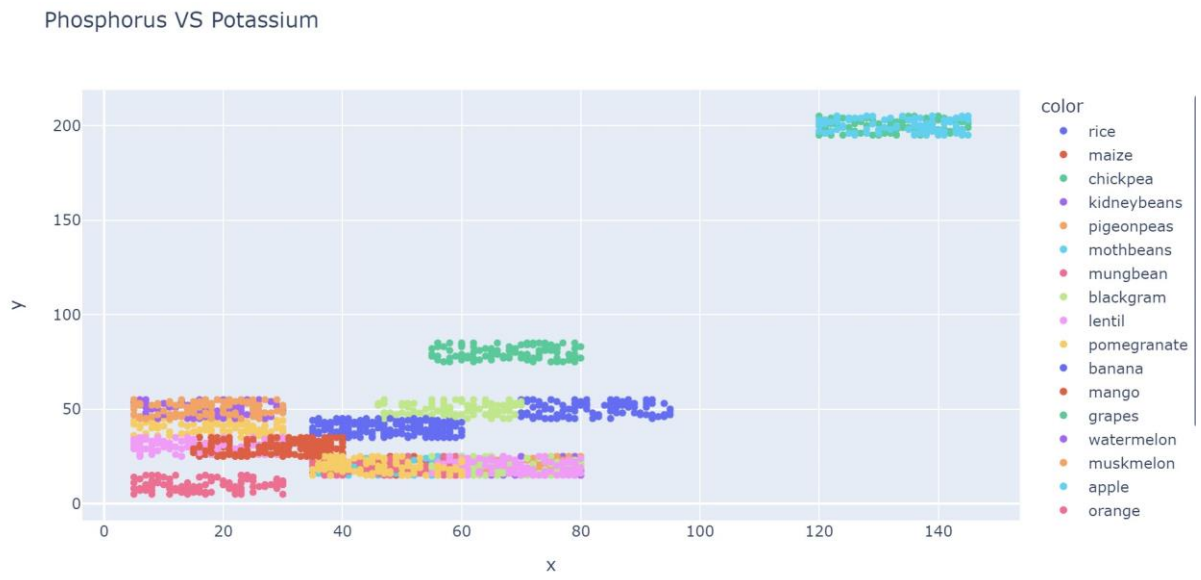


fig = px.scatter(x=df['P'],y=df['K'],color=df['label'],title="Phosphorus VS Potassium")

fig.show()

Output:




```
X=df.drop(['label'],axis=1)

y=df['label']

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3,shuffle = True, random_state = 42,stratify=y)

from sklearn.preprocessing import StandardScaler

scaler=StandardScaler()

X_train=scaler.fit_transform(X_train)

X_train=pd.DataFrame(X_train,columns=X.columns)

X_test=scaler.transform(X_test)

X_train.head()

from sklearn.metrics import accuracy_score

model.fit(X_train,y_train)

y_pred=model.predict(X_test)

score=accuracy_score(y_test,y_pred)

score
```

Output:

```
0.9727272727272728
```

4. **Evaluation Metrics :**

```
from sklearn.metrics import classification_report, confusion_matrix
report=classification_report(y_pred,y_test)
print(report)
```

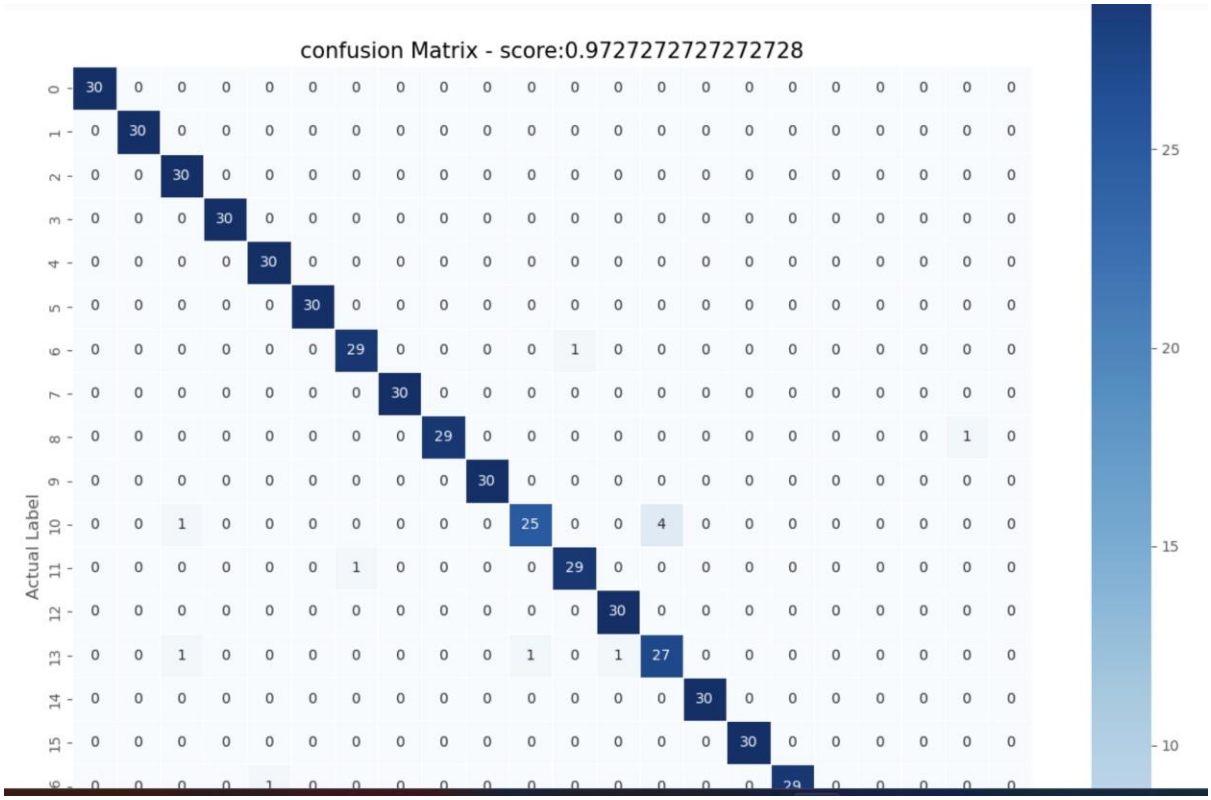
Output:

```
print(report)
```

	precision	recall	f1-score	support
apple	1.00	1.00	1.00	30
banana	1.00	1.00	1.00	30
blackgram	1.00	0.94	0.97	32
chickpea	1.00	1.00	1.00	30
coconut	1.00	0.97	0.98	31
coffee	1.00	1.00	1.00	30
cotton	0.97	0.97	0.97	30
grapes	1.00	1.00	1.00	30
jute	0.97	0.85	0.91	34
kidneybeans	1.00	1.00	1.00	30
lentil	0.83	0.96	0.89	26
maize	0.97	0.97	0.97	30
mango	1.00	0.97	0.98	31
mothbeans	0.90	0.87	0.89	31
mungbean	1.00	1.00	1.00	30
muskmelon	1.00	1.00	1.00	30
orange	0.97	1.00	0.98	29
papaya	0.97	1.00	0.98	29
pigeonpeas	1.00	1.00	1.00	30
pomegranate	1.00	1.00	1.00	30
rice	0.83	0.93	0.88	27
watermelon	1.00	1.00	1.00	30
accuracy			0.97	660
macro avg	0.97	0.97	0.97	660
weighted avg	0.97	0.97	0.97	660

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,y_pred)
plt.figure(figsize=(15,15))
sns.heatmap(cm,annot=True,fmt=".0f",linewidth=.5, square=True, cmap='Blues');
plt.ylabel('Actual Label');
plt.xlabel('Predicted Label');
all_sample_title = 'confusion Matrix - score:'+str(accuracy_score(y_test,y_pred))
plt.title(all_sample_title, size=15);
plt.show()
```

Output :



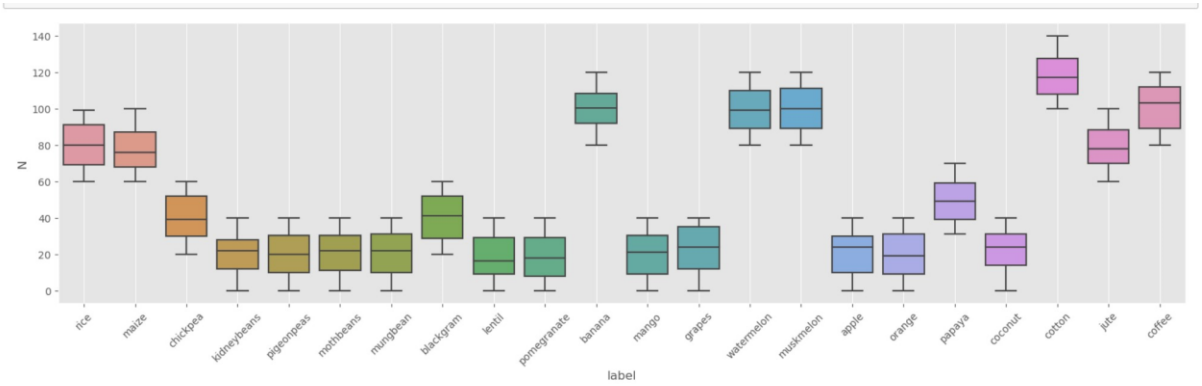

```
from sklearn.metrics import cohen_kappa_score
# Calculate Cohen's Kappa score
kappa = cohen_kappa_score(y_test, y_pred)
print("Cohen's Kappa Score:", kappa)
```

Output:

```
Cohen's Kappa Score: 0.9714285714285714
```

```
plt.figure(figsize=(20,5))
plt.xticks(rotation=45)
sns.boxplot(data=df, x = 'label', y = 'N')
plt.grid()
plt.show()
```

Output:


N
label
rice
maize
chickpea
kidneybeans
pigeonpeas
mothbeans
mungbean
blackgram
lentil
pomegranate
banana
mango
grapes
watermelon
muskmelon
apple
orange
papaya
coconut
cotton
jute
coffee