In [107…
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [7]:
```
#Load the data set
df=pd.read_csv('Crop_recommendation.csv')
```

In [19]:
```
df.head()
```

Out[19]:

	N	P	K	temperature	humidity	ph	rainfall	label
0	90	42	43	20.879744	82.002744	6.502985	202.935536	rice
1	85	58	41	21.770462	80.319644	7.038096	226.655537	rice
2	60	55	44	23.004459	82.320763	7.840207	263.964248	rice
3	74	35	40	26.491096	80.158363	6.980401	242.864034	rice
4	78	42	42	20.130175	81.604873	7.628473	262.717340	rice

In [25]:
```
#Shape of the data
df.shape
```

Out[25]:
```
(2200, 8)
```

In [13]:
```
#Check the missing Values
df.isnull().sum()
```

Out[13]:
```
N              0
P              0
K              0
temperature    0
humidity       0
ph             0
rainfall       0
label          0
dtype: int64
```

In [ ]:
```

```

In [15]:
```
#checking of the duplicated valuess
df.duplicated().sum()
```

Out[15]:
```
0
```

In [63]:
```
# Seperating the Input and target variables
x=df.iloc[:,:7]
y=df.iloc[:,7:]
y.value_counts()
```

Out[63]:
```
label
apple          100
banana         100
rice           100
pomegranate    100
pigeonpeas     100
papaya         100
orange         100
muskmelon      100
mungbean       100
mothbeans      100
mango          100
maize          100
lentil         100
kidneybeans    100
jute           100
grapes         100
cotton         100
coffee         100
coconut        100
chickpea       100
blackgram      100
watermelon     100
Name: count, dtype: int64
```

In [51]:
```
#First of all we apply the Label Encoder on y variable
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
y_labeled=le.fit_transform(y)
y_labeled
```

```
C:\Users\ALI\anaconda3\Lib\site-packages\sklearn\preprocessing\_label.py:114: DataCo
nversionWarning: A column-vector y was passed when a 1d array was expected. Please c
hange the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[51]:
```
array([20, 20, 20, ...,  5,  5,  5])
```

In [65]:
```
#Checking of the values assign to catagorical Values
print(le.classes_)
print(le.transform(le.classes_))
```

```
['apple' 'banana' 'blackgram' 'chickpea' 'coconut' 'coffee' 'cotton'
 'grapes' 'jute' 'kidneybeans' 'lentil' 'maize' 'mango' 'mothbeans'
 'mungbean' 'muskmelon' 'orange' 'papaya' 'pigeonpeas' 'pomegranate'
 'rice' 'watermelon']
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21]
```

In [74]:
```
x.corr()
```

Out[74]:

	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

In [72]:
```
#Convert the Y_labeled in to dataframe
y_lab=pd.DataFrame(y_labeled)
y_lab
```

Out[72]:

	0
0	20
1	20
2	20
3	20
4	20
...	...
2195	5
2196	5
2197	5
2198	5
2199	5

2200 rows × 1 columns

In [81]:
```
#train test split
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(x,y_lab,test_size=0.2,random_state=4
```

In [86]:
```
#Now apply pipeline for fitting Model
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
```

In [98]:
```
#Applying the Pipline Function
pipeline=Pipeline([
    ('scalar',StandardScaler()),
    ('model' ,LogisticRegression())
])
pipeline.fit(X_train,y_train)
y_pred=pipeline.predict(X_test)
```

```
C:\Users\ALI\anaconda3\Lib\site-packages\sklearn\utils\validation.py:1339: DataConve
rsionWarning: A column-vector y was passed when a 1d array was expected. Please chan
ge the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

In [100…
```
#Checking the accuracy of Logistic Regression
from sklearn.metrics import accuracy_score
accuracy=accuracy_score(y_pred,y_test)
accuracy
```

Out[100…
```
0.9636363636363636
```

In [102…
```
#saving the model in pickle
import pickle
pickle.dump(pipeline,open('Crop_Recommendation.pkl','wb'))
```

In [ ]:
```
```