23122104-postmidem-cia

May 16, 2024

[51]:

```
import pandas as pd
df=pd.read_csv("crop_recommendation.csv")
print(df)
```

```
      Nitrogen  Phosphorus  Potassium  Temperature   Humidity  pH_Value  \
0           90          42         43    20.879744  82.002744  6.502985
1           85          58         41    21.770462  80.319644  7.038096
2           60          55         44    23.004459  82.320763  7.840207
3           74          35         40    26.491096  80.158363  6.980401
4           78          42         42    20.130175  81.604873  7.628473
…          …          …          …          …          …        …
2195       107          34         32    26.774637  66.413269  6.780064
2196        99          15         27    27.417112  56.636362  6.086922
2197       118          33         30    24.131797  67.225123  6.362608
2198       117          32         34    26.272418  52.127394  6.758793
2199       104          18         30    23.603016  60.396475  6.779833

        Rainfall    Crop
0     202.935536    Rice
1     226.655537    Rice
2     263.964248    Rice
3     242.864034    Rice
4     262.717340    Rice
…            …      …
2195  177.774507  Coffee
2196  127.924610  Coffee
2197  173.322839  Coffee
2198  127.175293  Coffee
2199  140.937041  Coffee

[2200 rows x 8 columns]
```

[52]:

```
#print first 5 rows of the dataframe
df.head()
```

[52]:

```
   Nitrogen  Phosphorus  Potassium  Temperature   Humidity  pH_Value  \
0        90          42         43    20.879744  82.002744  6.502985
1        85          58         41    21.770462  80.319644  7.038096
```

```
2        60          55          44    23.004459  82.320763  7.840207
3        74          35          40    26.491096  80.158363  6.980401
4        78          42          42    20.130175  81.604873  7.628473

     Rainfall  Crop
0  202.935536  Rice
1  226.655537  Rice
2  263.964248  Rice
3  242.864034  Rice
4  262.717340  Rice
```

[53]:

```
#print last 5 rows of the dataframe
df.tail()
```

[53]:

```
      Nitrogen  Phosphorus  Potassium  Temperature   Humidity  pH_Value  \
2195       107          34         32    26.774637  66.413269  6.780064
2196        99          15         27    27.417112  56.636362  6.086922
2197       118          33         30    24.131797  67.225123  6.362608
2198       117          32         34    26.272418  52.127394  6.758793
2199       104          18         30    23.603016  60.396475  6.779833

        Rainfall    Crop
2195  177.774507  Coffee
2196  127.924610  Coffee
2197  173.322839  Coffee
2198  127.175293  Coffee
2199  140.937041  Coffee
```

[54]:

```
# shape of the dataframe
df.shape
```

[54]:

```
(2200, 8)
```

[55]:

```
#summary of the dataframe
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2200 entries, 0 to 2199
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Nitrogen     2200 non-null   int64
 1   Phosphorus   2200 non-null   int64
 2   Potassium    2200 non-null   int64
 3   Temperature  2200 non-null   float64
 4   Humidity     2200 non-null   float64
 5   pH_Value     2200 non-null   float64
 6   Rainfall     2200 non-null   float64
 7   Crop         2200 non-null   object
dtypes: float64(4), int64(3), object(1)
memory usage: 137.6+ KB
```

[56]:
```
#columns of the dataframe
df.columns
```

[56]:
```
Index(['Nitrogen', 'Phosphorus', 'Potassium', 'Temperature', 'Humidity',
       'pH_Value', 'Rainfall', 'Crop'],
      dtype='object')
```

[57]:
```
#check for null values
df.isnull().sum()
```

[57]:
```
Nitrogen       0
Phosphorus     0
Potassium      0
Temperature    0
Humidity       0
pH_Value       0
Rainfall       0
Crop           0
dtype: int64
```

[58]:
```
df
```

[58]:
```
      Nitrogen  Phosphorus  Potassium  Temperature   Humidity  pH_Value  \
0           90          42         43    20.879744  82.002744  6.502985
1           85          58         41    21.770462  80.319644  7.038096
2           60          55         44    23.004459  82.320763  7.840207
3           74          35         40    26.491096  80.158363  6.980401
4           78          42         42    20.130175  81.604873  7.628473
…          …          …         …          …          …        …
2195       107          34         32    26.774637  66.413269  6.780064
2196        99          15         27    27.417112  56.636362  6.086922
2197       118          33         30    24.131797  67.225123  6.362608
2198       117          32         34    26.272418  52.127394  6.758793
2199       104          18         30    23.603016  60.396475  6.779833

        Rainfall    Crop
0     202.935536    Rice
1     226.655537    Rice
2     263.964248    Rice
3     242.864034    Rice
4     262.717340    Rice
…          …      …
```

```
2195  177.774507  Coffee
2196  127.924610  Coffee
2197  173.322839  Coffee
2198  127.175293  Coffee
2199  140.937041  Coffee

[2200 rows x 8 columns]
```

[59]:

```
import numpy as np
X = np.array(df)
y = X[:,7]  ## all rows but column 9
X = X[:,0:7]  ## all rows but column from 0 to 9


print(X)
print(y)
```

```
[[90 42 43 … 82.00274423 6.502985292 202.9355362]
 [85 58 41 … 80.31964408 7.038096361 226.6555374]
 [60 55 44 … 82.3207629 7.840207144 263.9642476]
 …
 [118 33 30 … 67.22512329 6.362607851 173.3228386]
 [117 32 34 … 52.12739421 6.758792552 127.1752928]
 [104 18 30 … 60.39647474 6.779832611 140.9370415]]
['Rice' 'Rice' 'Rice' … 'Coffee' 'Coffee' 'Coffee']
```

[60]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y ,
                                   random_state=42,
                                   test_size=0.7,
                                   shuffle=True)

print(X_test)
print(y_train)
```

```
[[101 17 47 … 94.72981338 6.185053234 26.30820876]
 [98 8 51 … 86.52258079 6.25933595 49.43050977]
 [59 62 49 … 93.35191636 6.941496806 114.778071]
 …
 [131 38 19 … 75.68339729 6.814341946 90.4547185]
 [22 18 31 … 47.93791463 5.956027059 90.38503469]
 [60 49 44 … 84.49774397 6.244841491 240.0810647]]
['Apple' 'Grapes' 'Apple' 'Blackgram' 'Pomegranate' 'Papaya' 'Maize'
 'Coconut' 'Mango' 'Apple' 'Grapes' 'Orange' 'Pomegranate' 'PigeonPeas'
 'Pomegranate' 'Banana' 'Banana' 'MothBeans' 'Blackgram' 'MungBean'
 'ChickPea' 'Blackgram' 'Coffee' 'Cotton' 'Mango' 'Papaya' 'Orange'
 'Cotton' 'Pomegranate' 'Grapes' 'Banana' 'Maize' 'Coconut' 'MothBeans'
 'Mango' 'Mango' 'Apple' 'Muskmelon' 'Lentil' 'Pomegranate' 'Banana'
```

```
'Apple' 'Orange' 'Papaya' 'Lentil' 'Cotton' 'Coffee' 'Coconut' 'Lentil'
'Apple' 'Muskmelon' 'Blackgram' 'Blackgram' 'PigeonPeas' 'ChickPea'
'Rice' 'Cotton' 'PigeonPeas' 'KidneyBeans' 'Maize' 'Banana' 'Jute'
'Orange' 'Papaya' 'Coconut' 'MothBeans' 'Rice' 'Grapes' 'Lentil' 'Papaya'
'ChickPea' 'ChickPea' 'Apple' 'Watermelon' 'Rice' 'Papaya' 'MungBean'
'Jute' 'Papaya' 'Pomegranate' 'Apple' 'Jute' 'Blackgram' 'Maize'
'Pomegranate' 'PigeonPeas' 'Muskmelon' 'MungBean' 'Blackgram' 'Jute'
'Coconut' 'Blackgram' 'Cotton' 'Watermelon' 'PigeonPeas' 'Banana' 'Apple'
'Apple' 'Lentil' 'Papaya' 'MungBean' 'Maize' 'PigeonPeas' 'Coffee'
'Orange' 'Pomegranate' 'MungBean' 'Coffee' 'Rice' 'Lentil' 'Coconut'
'Coffee' 'Blackgram' 'PigeonPeas' 'Grapes' 'Watermelon' 'Orange' 'Rice'
'KidneyBeans' 'ChickPea' 'Coffee' 'Banana' 'Mango' 'Muskmelon' 'Orange'
'Jute' 'Jute' 'Muskmelon' 'MungBean' 'Cotton' 'Apple' 'Coffee' 'MungBean'
'Lentil' 'Muskmelon' 'Maize' 'Lentil' 'Coconut' 'ChickPea' 'KidneyBeans'
'Coffee' 'Banana' 'MothBeans' 'Watermelon' 'Banana' 'Blackgram' 'Coffee'
'PigeonPeas' 'Jute' 'Blackgram' 'Grapes' 'Grapes' 'Papaya' 'Mango'
'MothBeans' 'Rice' 'KidneyBeans' 'Grapes' 'Watermelon' 'Orange' 'Cotton'
'KidneyBeans' 'PigeonPeas' 'PigeonPeas' 'ChickPea' 'Pomegranate' 'Grapes'
'Coconut' 'Banana' 'Pomegranate' 'Rice' 'Banana' 'Banana' 'Banana'
'Cotton' 'Watermelon' 'MungBean' 'Mango' 'Pomegranate' 'Muskmelon'
'Orange' 'Maize' 'Banana' 'Jute' 'Coffee' 'Blackgram' 'MothBeans'
'Muskmelon' 'Blackgram' 'Grapes' 'Grapes' 'Cotton' 'Grapes' 'PigeonPeas'
'Orange' 'MothBeans' 'Mango' 'Jute' 'Mango' 'Lentil' 'MothBeans'
'MungBean' 'Papaya' 'Orange' 'Banana' 'Coconut' 'Mango' 'Blackgram'
'Maize' 'Muskmelon' 'Blackgram' 'Jute' 'Banana' 'Orange' 'Lentil' 'Maize'
'MungBean' 'Cotton' 'Cotton' 'Coffee' 'MungBean' 'KidneyBeans' 'ChickPea'
'MungBean' 'Pomegranate' 'Lentil' 'Lentil' 'Rice' 'Watermelon' 'Jute'
'Pomegranate' 'Lentil' 'Blackgram' 'Grapes' 'Mango' 'Rice' 'Jute'
'Banana' 'Cotton' 'Jute' 'ChickPea' 'Coconut' 'MungBean' 'PigeonPeas'
'Banana' 'Banana' 'Lentil' 'MungBean' 'Papaya' 'KidneyBeans' 'Watermelon'
'Coconut' 'KidneyBeans' 'Maize' 'Mango' 'Grapes' 'MothBeans' 'Papaya'
'Orange' 'Rice' 'Coconut' 'Blackgram' 'Lentil' 'Watermelon' 'Mango'
'Mango' 'PigeonPeas' 'Pomegranate' 'MungBean' 'KidneyBeans' 'MothBeans'
'Coconut' 'KidneyBeans' 'Rice' 'MungBean' 'Rice' 'Pomegranate' 'Jute'
'Lentil' 'MothBeans' 'PigeonPeas' 'Pomegranate' 'MothBeans' 'Blackgram'
'Papaya' 'Coffee' 'Banana' 'Mango' 'ChickPea' 'Muskmelon' 'Mango'
'Blackgram' 'Muskmelon' 'PigeonPeas' 'Maize' 'Orange' 'Pomegranate'
'Orange' 'Lentil' 'Watermelon' 'MungBean' 'Coconut' 'MungBean' 'Coconut'
'MothBeans' 'MothBeans' 'Papaya' 'Coconut' 'Rice' 'Maize' 'Watermelon'
'Grapes' 'Coffee' 'Pomegranate' 'Pomegranate' 'ChickPea' 'Maize' 'Mango'
'Pomegranate' 'MungBean' 'Cotton' 'MungBean' 'Maize' 'KidneyBeans'
'PigeonPeas' 'Jute' 'Coffee' 'Cotton' 'Jute' 'Cotton' 'MothBeans'
'Watermelon' 'Banana' 'PigeonPeas' 'Maize' 'Grapes' 'Mango' 'Rice'
'Apple' 'Banana' 'Coffee' 'MothBeans' 'Apple' 'MungBean' 'Cotton'
'Coconut' 'Banana' 'Cotton' 'Apple' 'Pomegranate' 'Coffee' 'ChickPea'
'Orange' 'Blackgram' 'Cotton' 'ChickPea' 'Rice' 'Jute' 'Papaya'
'PigeonPeas' 'Coffee' 'Watermelon' 'ChickPea' 'Rice' 'Orange' 'Blackgram'
'Banana' 'Watermelon' 'PigeonPeas' 'Cotton' 'Jute' 'Coffee' 'MungBean'
```

```
'Banana' 'MungBean' 'Jute' 'Grapes' 'Mango' 'Muskmelon' 'Apple' 'Cotton'
'Watermelon' 'Coffee' 'Lentil' 'Orange' 'Maize' 'Papaya' 'Pomegranate'
'Papaya' 'Mango' 'Lentil' 'Papaya' 'Cotton' 'Lentil' 'Cotton'
'Pomegranate' 'Coconut' 'Maize' 'Rice' 'Jute' 'MothBeans' 'Mango'
'Grapes' 'Cotton' 'Coconut' 'Maize' 'Muskmelon' 'Muskmelon' 'Orange'
'MothBeans' 'MungBean' 'Cotton' 'Orange' 'Mango' 'Rice' 'Maize'
'PigeonPeas' 'Mango' 'ChickPea' 'Coffee' 'Orange' 'Mango' 'Grapes' 'Rice'
'Banana' 'Jute' 'Banana' 'PigeonPeas' 'Orange' 'ChickPea' 'Watermelon'
'Lentil' 'Jute' 'Lentil' 'Apple' 'Coffee' 'MungBean' 'KidneyBeans' 'Jute'
'Papaya' 'ChickPea' 'Orange' 'Watermelon' 'Muskmelon' 'Maize' 'Cotton'
'PigeonPeas' 'Cotton' 'Coconut' 'MungBean' 'Lentil' 'Coffee' 'MothBeans'
'Jute' 'Cotton' 'Pomegranate' 'PigeonPeas' 'Coconut' 'Papaya' 'Jute'
'Mango' 'Maize' 'Jute' 'Cotton' 'Watermelon' 'PigeonPeas' 'Coffee'
'Pomegranate' 'Coffee' 'MungBean' 'Papaya' 'Watermelon' 'PigeonPeas'
'Watermelon' 'Maize' 'Mango' 'Grapes' 'MungBean' 'Maize' 'Blackgram'
'Apple' 'Cotton' 'Pomegranate' 'Maize' 'Orange' 'Lentil' 'Coconut'
'ChickPea' 'ChickPea' 'Blackgram' 'PigeonPeas' 'Maize' 'Grapes'
'ChickPea' 'Muskmelon' 'Lentil' 'ChickPea' 'Muskmelon' 'Papaya' 'Orange'
'Blackgram' 'Papaya' 'Coffee' 'Mango' 'Orange' 'Muskmelon' 'KidneyBeans'
'Grapes' 'Papaya' 'Lentil' 'Apple' 'MungBean' 'Rice' 'MungBean' 'Orange'
'Grapes' 'Jute' 'Apple' 'Coconut' 'Mango' 'Coffee' 'Orange' 'Cotton'
'MungBean' 'MungBean' 'Grapes' 'KidneyBeans' 'MungBean' 'Muskmelon'
'Muskmelon' 'Blackgram' 'Grapes' 'Papaya' 'Pomegranate' 'Watermelon'
'Muskmelon' 'Muskmelon' 'Muskmelon' 'Jute' 'Apple' 'Orange' 'Watermelon'
'Coffee' 'Coffee' 'Orange' 'Banana' 'Blackgram' 'Apple' 'Coconut'
'Papaya' 'Mango' 'Banana' 'MothBeans' 'Orange' 'Papaya' 'Cotton' 'Rice'
'Banana' 'MothBeans' 'Lentil' 'Muskmelon' 'Coconut' 'Banana' 'ChickPea'
'KidneyBeans' 'Lentil' 'KidneyBeans' 'Coffee' 'Banana' 'Muskmelon'
'Pomegranate' 'Apple' 'Coconut' 'Lentil' 'Papaya' 'MothBeans' 'Blackgram'
'Coconut' 'ChickPea' 'Jute' 'Papaya' 'KidneyBeans' 'PigeonPeas' 'Orange'
'KidneyBeans' 'Apple' 'Mango' 'Muskmelon' 'Jute' 'Mango' 'Watermelon'
'Rice' 'Jute' 'Mango' 'PigeonPeas' 'KidneyBeans' 'Blackgram' 'Blackgram'
'Banana' 'Lentil' 'KidneyBeans' 'Banana' 'Grapes' 'PigeonPeas' 'Lentil'
'Pomegranate' 'Cotton' 'ChickPea' 'Maize' 'Apple' 'Blackgram' 'Apple'
'Mango' 'Banana' 'Apple' 'Cotton' 'Rice' 'Blackgram' 'Muskmelon'
'Watermelon' 'Coffee' 'Watermelon' 'Jute' 'ChickPea' 'Jute' 'MungBean'
'Grapes' 'Jute' 'MungBean' 'Apple' 'Grapes' 'Coconut' 'MothBeans' 'Maize'
'Coconut' 'Pomegranate' 'Jute' 'Banana' 'PigeonPeas' 'Blackgram' 'Rice'
'PigeonPeas' 'Mango' 'Pomegranate' 'Grapes' 'Apple' 'Blackgram' 'Orange'
'Maize' 'Coffee' 'Muskmelon' 'KidneyBeans' 'Grapes' 'PigeonPeas' 'Coffee'
'Orange' 'Banana' 'Mango' 'Grapes' 'Lentil']
```

[61]:

```
from sklearn.ensemble import RandomForestClassifier
# Initialize Random Forest classifier
rf_classifier = RandomForestClassifier(n_estimators=100, max_depth=4,␣
 ↪random_state=42)
```

[62]:
```
# Train the classifier
rf_classifier.fit(X_train, y_train)
```

[62]:
```
RandomForestClassifier(max_depth=4, random_state=42)
```

[63]:
```
# Make predictions on the test set
predictions = rf_classifier.predict(X_test)
```

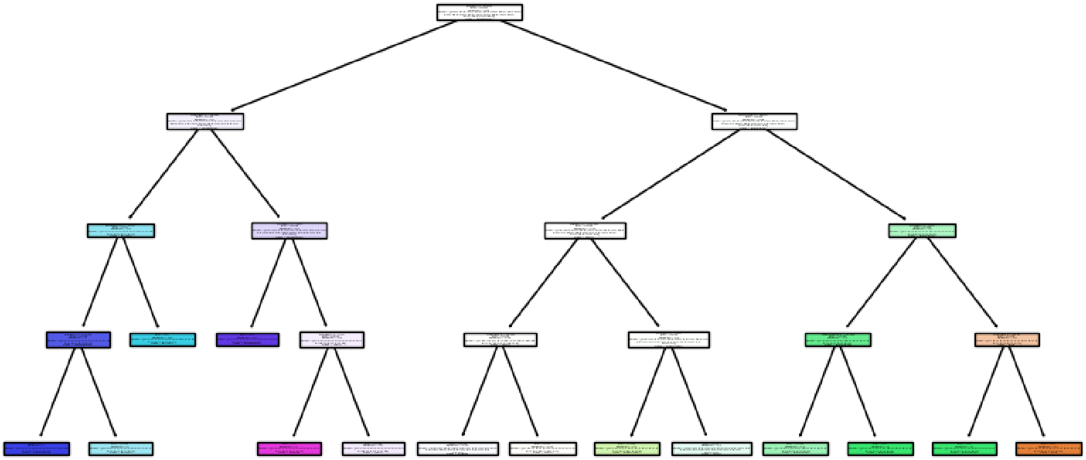
[64]:
```
# Evaluate the model
accuracy = rf_classifier.score(X_test, y_test)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.9090909090909091
```

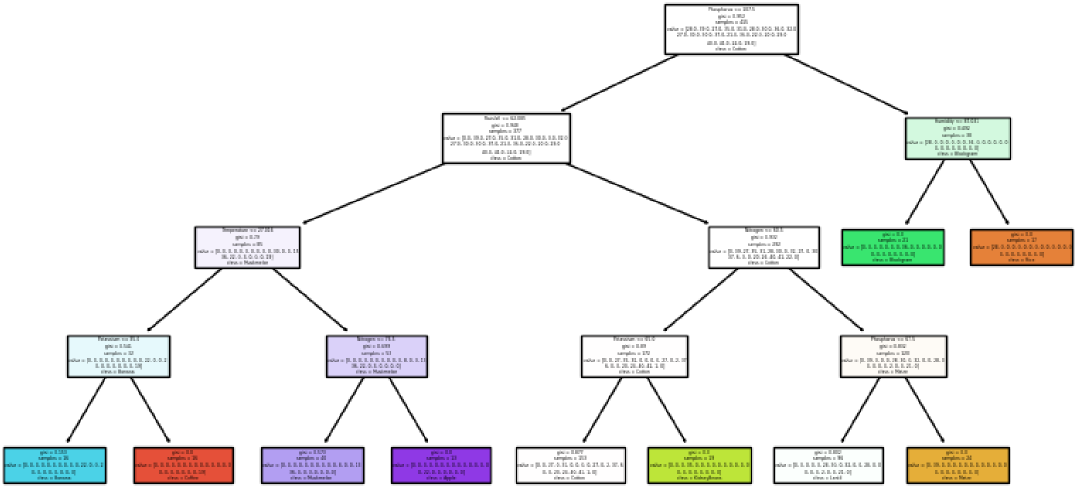
Accuracy is 90%

[66]:
```
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree

# Assuming you have loaded your data into a DataFrame called df
feature_names = df.columns[:-1]  # Assuming the last column is the target␣
 ↪variable
target_names = df[df.columns[-1]].unique()

# Define the maximum number of plots
max_plots = 10
num_plots = min(max_plots, len(rf_classifier.estimators_))

# Plot decision boundaries of a subset of trees in the forest
for i in range(num_plots):
    plt.figure(figsize=(10, 5))
    plot_tree(rf_classifier.estimators_[i], filled=True,␣
 ↪feature_names=feature_names, class_names=target_names)
    plt.title('Decision Tree {}'.format(i+1))
    plt.show()
```

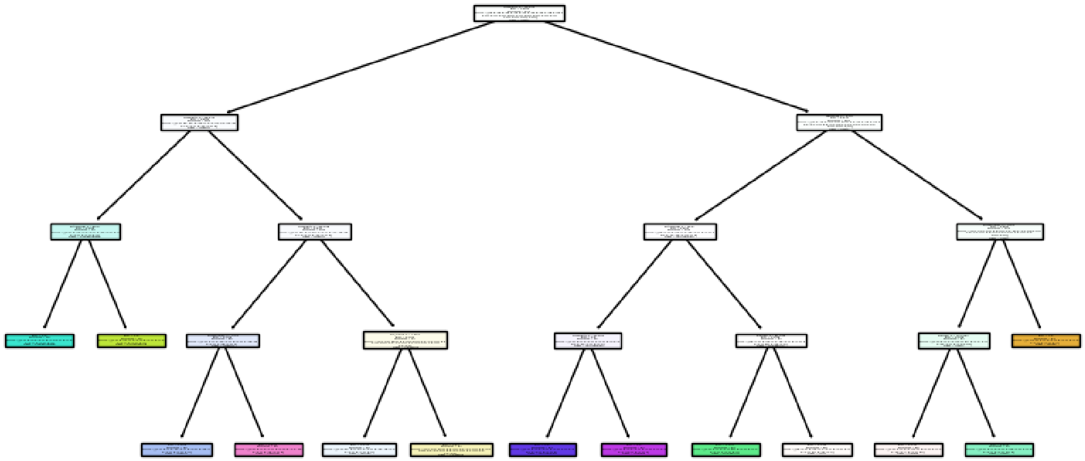
Decision Tree 1



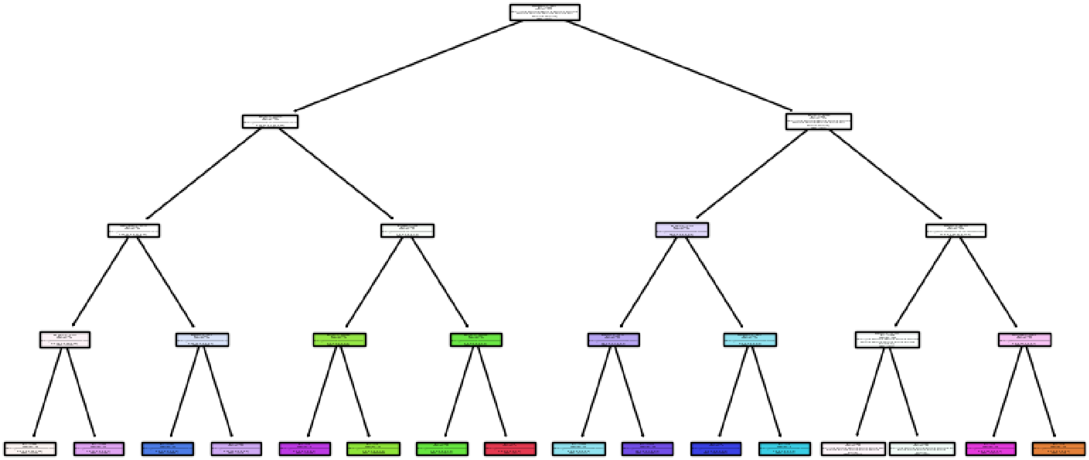
Decision Tree 2



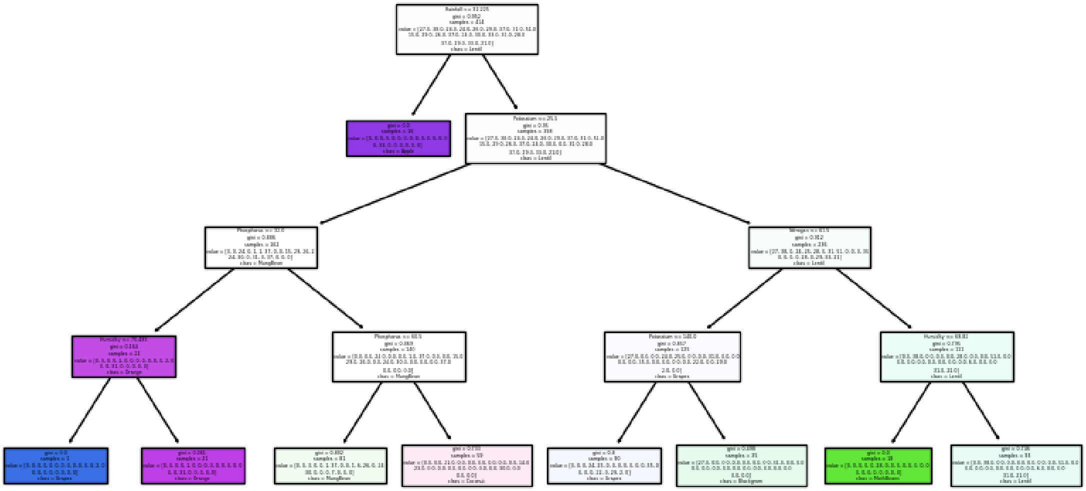
Decision Tree 3



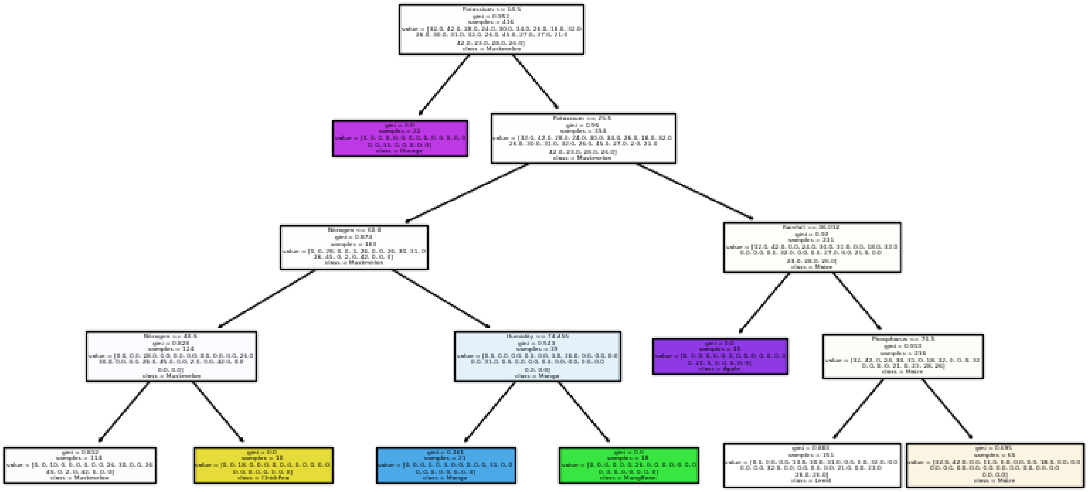
Decision Tree 4



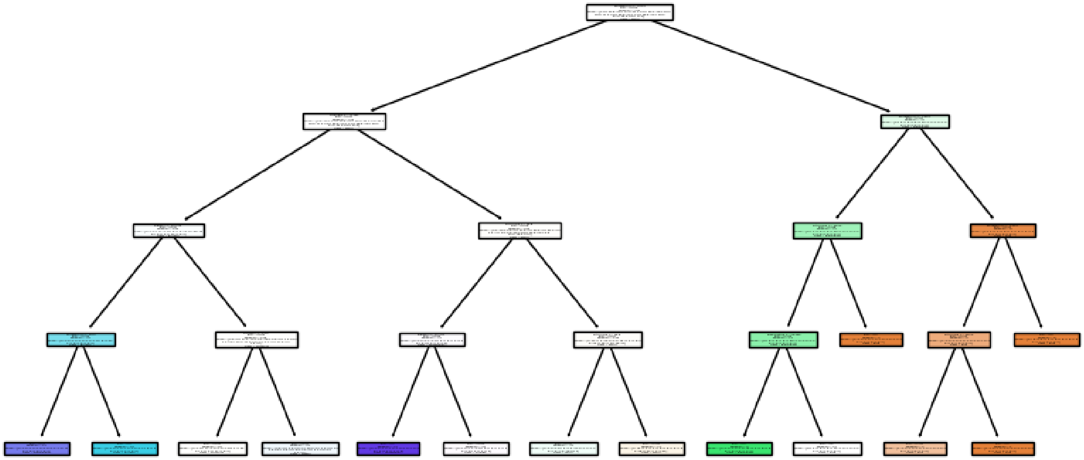
Decision Tree 5



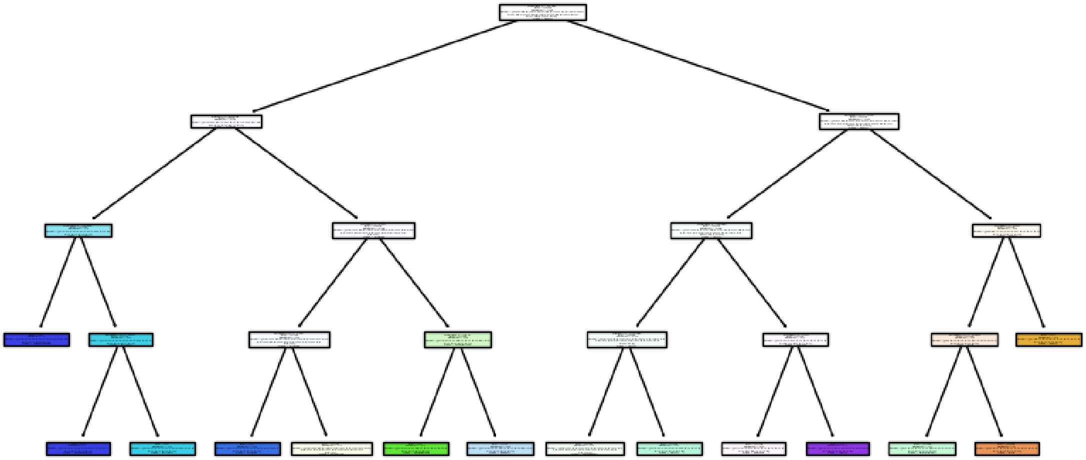
Decision Tree 6



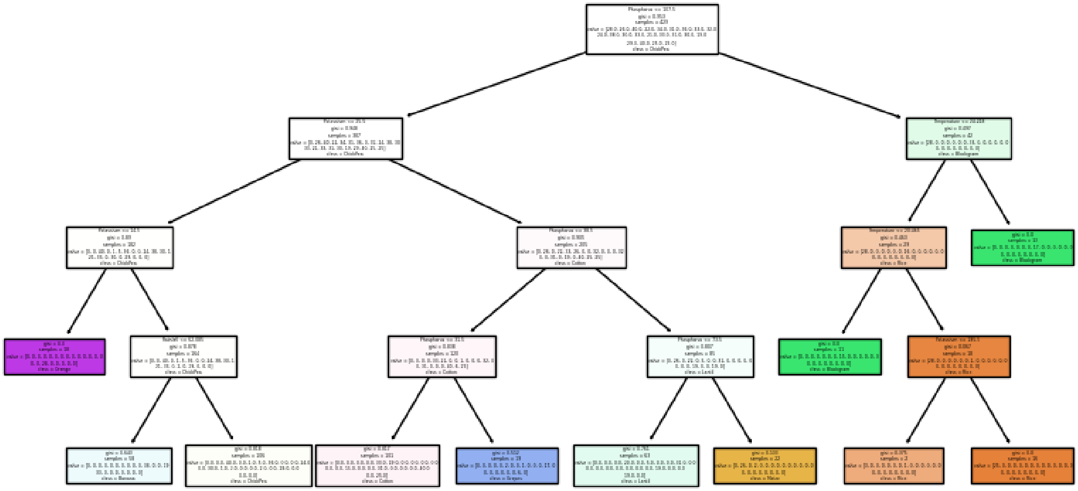
Decision Tree 7



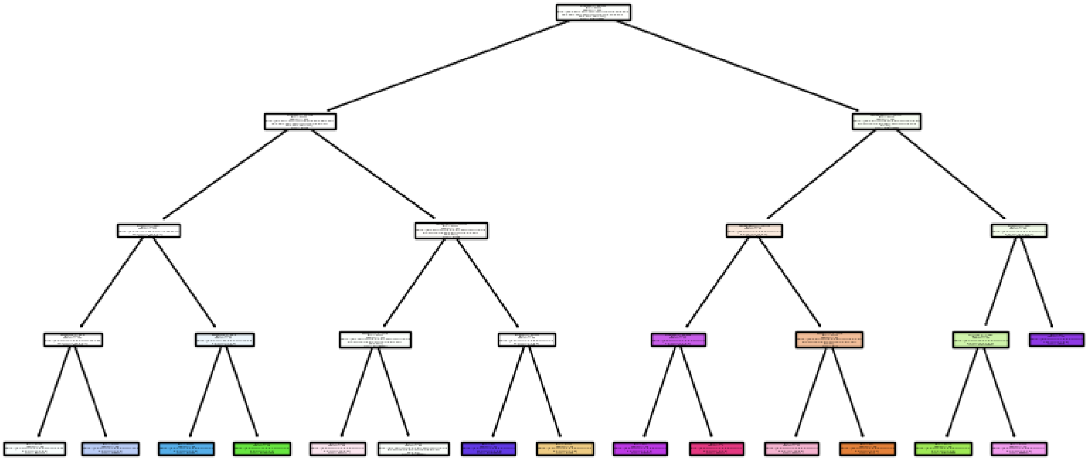
Decision Tree 8



Decision Tree 9



Decision Tree 10



[ ]: