Name: Abhishek Mali
Roll No.: 213070
PRN No.: 22320024

Tutorial No.5

IMPLEMENT FOLLOWING ALGORITHMS USING PYTHON ON SUITABLE DATA SETS. I. DECISION TREE II. NAÏVE BAYES III. RANDOM FOREST

---

[3]:
```
import numpy as np
import pandas as pd
```

[81]:
```
dm=pd.read_csv('bankloan.csv')
```

[82]:
```
dm
```

[82]:

	ID	Age	Experience	Income	ZIP.Code	Family	CCAvg	Education	Mortgage	Personal.Loan	Securities.Account	CD.Account	Online	CreditCard
0	1	25	1	49	91107	4	1.6	1	0	0	1	0	0	0
1	2	45	19	34	90089	3	1.5	1	0	0	1	0	0	0
2	3	39	15	11	94720	1	1.0	1	0	0	0	0	0	0
3	4	35	9	100	94112	1	2.7	2	0	0	0	0	0	0
4	5	35	8	45	91330	4	1.0	2	0	0	0	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4995	4996	29	3	40	92697	1	1.9	3	0	0	0	0	1	0
4996	4997	30	4	15	92037	4	0.4	1	85	0	0	0	1	0
4997	4998	63	39	24	93023	2	0.3	3	0	0	0	0	0	0
4998	4999	65	40	49	90034	3	0.5	2	0	0	0	0	1	0
4999	5000	28	4	83	92612	3	0.8	1	0	0	0	0	1	1

5000 rows × 14 columns

[83]:
```
import numpy as np
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics
```

[84]:
```
dm.head()
```

[84]:

	ID	Age	Experience	Income	ZIP.Code	Family	CCAvg	Education	Mortgage	Personal.Loan	Securities.Account	CD.Account	Online	CreditCard
0	1	25	1	49	91107	4	1.6	1	0	0	1	0	0	0
1	2	45	19	34	90089	3	1.5	1	0	0	1	0	0	0
2	3	39	15	11	94720	1	1.0	1	0	0	0	0	0	0
3	4	35	9	100	94112	1	2.7	2	0	0	0	0	0	0
4	5	35	8	45	91330	4	1.0	2	0	0	0	0	0	1

[85]:
```
from sklearn.preprocessing import LabelEncoder
```

[86]:
```
le = LabelEncoder()
```

[87]:
```
cols =['Outlook', 'Temperature', 'Humidity', 'Wind','Play Tennis']
```

[20]:
```
for col in cols:
  dm[col]=le.fit_transform(df[col])
  dm.head()
```

Name: Abhishek Mali
Roll No.: 213070
PRN No.: 22320024

[22]:
```
dm
```

[22]:

	Outlook	Temperature	Humidity	Wind	Play Tennis
0	2	1	0	1	0
1	2	1	0	0	0
2	0	1	0	1	1
3	1	2	0	1	1
4	1	0	1	1	1
5	1	0	1	0	0
6	0	0	1	0	1
7	2	2	0	1	0
8	2	0	1	1	1
9	1	2	1	1	1
10	2	2	1	0	1
11	0	2	0	0	1
12	0	1	1	1	1
13	1	2	0	0	0

[24]:
```
X = dm.drop(['Play Tennis'], axis = 1)
y = dm['Play Tennis']
```

[25]:
```
X
```

[25]:

	Outlook	Temperature	Humidity	Wind
0	2	1	0	1
1	2	1	0	0
2	0	1	0	1
3	1	2	0	1
4	1	0	1	1
5	1	0	1	0
6	0	0	1	0
7	2	2	0	1
8	2	0	1	1
9	1	2	1	1
10	2	2	1	0
11	0	2	0	0
12	0	1	1	1
13	1	2	0	0

[26]:
```
y
```

[26]:
```
0     0
```

Name: Abhishek Mali
Roll No.: 213070
PRN No.: 22320024

[26]:
```
y
```

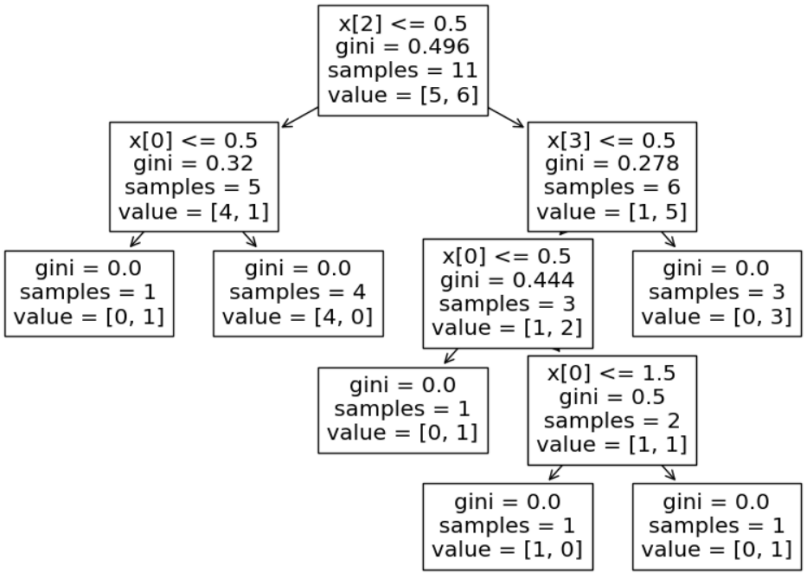
[26]:
```
0     0
1     0
2     1
3     1
4     1
5     0
6     1
7     0
8     1
9     1
10    1
11    1
12    1
13    0
Name: Play Tennis, dtype: int64
```

[27]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 69)
print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)
```

```
(11, 4) (3, 4) (11,) (3,)
```

[28]:
```
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier(random_state = 42)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

[29]:
```
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import tree
fig = plt.figure(figsize = (10, 7))
tree.plot_tree(model)
plt.show()
```



[32]:
```
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```
```
Accuracy: 0.6666666666666666
```

[33]:
```
X_pred = model.predict(X)
X_pred == y
```

[33]:
```
0      True
1      True
2      True
3     False
4      True
5      True
6      True
7      True
8      True
9      True
10     True
11     True
12     True
13     True
Name: Play Tennis, dtype: bool
```

Naïve Bayes Theorem

[34]:
```
#naive
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score
```

[52]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 43)
```

[56]:
```
model = MultinomialNB()
```

[58]:
```
dm.dtypes
```

[58]:
```
Outlook        int64
Temperature    int64
Humidity       int64
Wind           int64
Play Tennis    int64
dtype: object
```

[54]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

[59]:
```
model = MultinomialNB()
```

[61]:
```
df_encoded = pd.get_dummies(df.drop('Play Tennis', axis=1))
```

[62]:
```
X = df_encoded
```

[64]:
```
y = df['Play Tennis']
```

[65]:
```
model = MultinomialNB()
```

[66]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

[67]:
```
model.fit(X_train, y_train)
```

[67]:
```
MultinomialNB()
```

[68]:
```
y_pred = model.predict(X_test)
```

[69]:
```
print("Accuracy:", accuracy_score(y_test, y_pred))
```
```
Accuracy: 0.6
```

[71]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

Random Forest

[70]:
```
#m the randomforet
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

# Load the dataset into a pandas DataFrame
data = {
    'Outlook': ['Sunny', 'Sunny', 'Overcast', 'Rain', 'Rain', 'Rain', 'Overcast', 'Sunny', 'Sunny', 'Rain', 'Sunny', 'Overcast', 'Overcast', 'Rain'],
    'Temperature': ['Hot', 'Hot', 'Hot', 'Mild', 'Cool', 'Cool', 'Cool', 'Mild', 'Cool', 'Mild', 'Mild', 'Mild', 'Hot', 'Mild'],
    'Humidity': ['High', 'High', 'High', 'High', 'Normal', 'Normal', 'Normal', 'High', 'Normal', 'Normal', 'Normal', 'High', 'Normal', 'High'],
    'Wind': ['Weak', 'Strong', 'Weak', 'Weak', 'Weak', 'Strong', 'Strong', 'Weak', 'Weak', 'Weak', 'Strong', 'Strong', 'Weak', 'Strong'],
    'Play Tennis': ['No', 'No', 'Yes', 'Yes', 'Yes', 'No', 'Yes', 'No', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'No']
}

df = pd.DataFrame(data)

# Perform one-hot encoding on categorical variables
df = pd.get_dummies(df)

# Define features (X) and target variable (y)
X = df.drop('Play Tennis_Yes', axis=1)  # Exclude one of the encoded columns to avoid multicollinearity
y = df['Play Tennis_Yes']

# Split dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Create Random Forest classifier
rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42)

# Train the classifier using the training sets
rf_classifier.fit(X_train, y_train)
```

```
# Train the classifier using the training sets
rf_classifier.fit(X_train, y_train)

# Predict the response for test dataset
y_pred = rf_classifier.predict(X_test)

# Model Accuracy, how often is the classifier correct?
print("Accuracy:", accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.8
```

[72]:
```
data = {
    'Outlook': ['Sunny', 'Sunny', 'Overcast', 'Rain', 'Rain', 'Rain', 'Overcast', 'Sunny', 'Sunny', 'Rain', 'Sunny', 'Overcast', 'Overcast', 'Rain'],
    'Temperature': ['Hot', 'Hot', 'Hot', 'Mild', 'Cool', 'Cool', 'Cool', 'Mild', 'Cool', 'Mild', 'Mild', 'Mild', 'Hot', 'Mild'],
    'Humidity': ['High', 'High', 'High', 'High', 'Normal', 'Normal', 'Normal', 'High', 'Normal', 'Normal', 'Normal', 'High', 'Normal', 'High'],
    'Wind': ['Weak', 'Strong', 'Weak', 'Weak', 'Weak', 'Strong', 'Strong', 'Weak', 'Weak', 'Weak', 'Strong', 'Strong', 'Weak', 'Strong'],
    'Play Tennis': ['No', 'No', 'Yes', 'Yes', 'Yes', 'No', 'Yes', 'No', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'No']
}
```

[73]:
```
df = pd.DataFrame(data)
```

[74]:
```
df = pd.get_dummies(df)
```

[75]:
```
X = df.drop('Play Tennis_Yes', axis=1)  # Exclude one of the encoded columns to avoid multicollinearity
y = df['Play Tennis_Yes']
```

Name: Abhishek Mali
Roll No.: 213070
PRN No.: 22320024

[76]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

[77]:
```
rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42)
```

[78]:
```
rf_classifier.fit(X_train, y_train)
```

[78]:
```
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

[79]:
```
y_pred = rf_classifier.predict(X_test)
```

[80]:
```
print("Accuracy:", accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.8
```