Naive Bayes

Using the Tennis Data sets

1- Provide a model based on Gaussian Naive bayes and calculate the accuracy.
2- What is the prediction for the following cases?

In [81]:

```
Case1=     ["Rain","Mild","Strong","Weak"]
Case2=     ["Overcast","Mild","Normal","Strong"]
Case3=     ["Sunny","Hot","High","Weak"]
```

In [71]:

```
# Required Python Machine learning Packages

import numpy as np
import pandas as pd
from sklearn.naive_bayes import GaussianNB
from sklearn.preprocessing import Imputer
from sklearn import preprocessing
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.naive_bayes import GaussianNB
import matplotlib.pyplot as plt
%matplotlib notebook
```

In [80]:

```
# Reading the .CSV file except row 1

dataset = pd.read_csv('F:/Seagate_Sync/VOL/VOL00/VIVEK/Big Data Analytics Cert
ification/BDA 102/Final Exam/NaiveBayes/NaiveBayes/Tennis.csv', dtype = 'unico
de')
```

In [82]:

```
# Printing the first 10 rows

print(dataset.head(10))
```

```
    Outlook Temperature Humidity    Wind Play Tennis
0     Sunny         Hot     High    Weak          No
1     Sunny         Hot     High  Strong          No
2  Overcast         Hot     High    Weak         Yes
3      Rain        Mild     High    Weak         Yes
4      Rain        Cool   Normal    Weak         Yes
5      Rain        Cool   Normal  Strong          No
6  Overcast        Cool   Normal  Strong         Yes
7     Sunny        Mild     High    Weak          No
8     Sunny        Cool   Normal    Weak         Yes
9      Rain        Mild   Normal    Weak         Yes
```

In [83]:

```
# Printing exploratory analysis using describe function

print(dataset.describe())
```

```
       Outlook Temperature Humidity  Wind Play Tennis
count       14          14       14    14          14
unique       3           3        2     2           2
top      Sunny        Mild   Normal  Weak         Yes
freq         5           6        7     8           9
```

In [84]:

```
# Converting categorical data from the CSV table into numbers using LabelEncoder

number = LabelEncoder()
dataset['Outlook'] = number.fit_transform(dataset['Outlook'])
dataset['Temperature'] = number.fit_transform(dataset['Temperature'])
dataset['Humidity'] = number.fit_transform(dataset['Humidity'])
dataset['Wind'] = number.fit_transform(dataset['Wind'])
dataset['Play Tennis'] = number.fit_transform(dataset['Play Tennis'])
```

In [85]:

```
# Splitting the file into features and target

array = dataset.values
features = array[:, 0:4]
target = array[:, 4]

#features = ["Outlook", "Temperature", "Humidity", "Wind"]
#target = "Play Tennis"
```

In [86]:

```
# Printing array

print(array)
```

```
[[2 1 0 1 0]
 [2 1 0 0 0]
 [0 1 0 1 1]
 [1 2 0 1 1]
 [1 0 1 1 1]
 [1 0 1 0 0]
 [0 0 1 0 1]
 [2 2 0 1 0]
 [2 0 1 1 1]
 [1 2 1 1 1]
 [2 2 1 0 1]
 [0 2 0 0 1]
 [0 1 1 1 1]
 [1 2 0 0 0]]
```

In [87]:
```
# Printing features

print(features)
```

```
[[2 1 0 1]
 [2 1 0 0]
 [0 1 0 1]
 [1 2 0 1]
 [1 0 1 1]
 [1 0 1 0]
 [0 0 1 0]
 [2 2 0 1]
 [2 0 1 1]
 [1 2 1 1]
 [2 2 1 0]
 [0 2 0 0]
 [0 1 1 1]
 [1 2 0 0]]
```

In [88]:
```
# Printing targets

print(target)
```

```
[0 0 1 1 1 0 1 0 1 1 1 1 1 0]
```

In [89]:
```
# Split the data into training and test set

features_train, features_test, target_train, target_test = train_test_split(fe
atures,

                                                                            ta
rget, test_size = 0.33, random_state = 54)
```

In [94]:
```
# Creating the model

clf = GaussianNB()

clf.fit(features_train, target_train)
```

Out[94]:
```
GaussianNB(priors=None)
```

In [95]:
```
# Model performance and accuracy

target_pred = clf.predict(features_test)
accuracy = accuracy_score(target_test, target_pred)
```

In [96]:
```
print(accuracy)
```

```
0.8
```

In [99]:

```
# Prediction for Case1= ["Rain","Mild","Strong","Weak"]

print (clf.predict([[1,2,0,1]]))
```

```
[1]
```

In [100]:

```
# Prediction for Case2= ["Overcast","Mild","Normal","Strong"]

print (clf.predict([[0,2,1,0]]))
```

```
[1]
```

In [101]:

```
# Prediction for Case3= ["Sunny","Hot","High","Weak"]

print (clf.predict([[2,1,0,1]]))
```

```
[0]
```