5) Aim: Implement Naïve Bayes Classifier using sklearn library

In [89]:
```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
```

With Play-tennis Dataset

In [90]:
```
df = pd.read_csv("play-tennis.csv")
```

In [91]:
```
df
```

Out[91]:

	outlook	temperature	humidity	wind	answer
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
10	sunny	mild	normal	strong	yes
11	overcast	mild	high	strong	yes
12	overcast	hot	normal	weak	yes
13	rain	mild	high	strong	no

In [92]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14 entries, 0 to 13
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   outlook      14 non-null     object
 1   temperature  14 non-null     object
 2   humidity     14 non-null     object
 3   wind         14 non-null     object
 4   answer       14 non-null     object
dtypes: object(5)
memory usage: 692.0+ bytes
```

In [93]:
```
df.describe()
```

Out[93]:

	outlook	temperature	humidity	wind	answer
count	14	14	14	14	14
unique	3	3	2	2	2
top	sunny	mild	high	weak	yes
freq	5	6	7	8	9

In [94]:
```
from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
```

In [95]:
```
df['outlook'] = encoder.fit_transform(df['outlook'])
df['temperature'] = encoder.fit_transform(df['temperature'])
df['humidity'] = encoder.fit_transform(df['humidity'])
df['wind'] = encoder.fit_transform(df['wind'])
```

In [96]:
```
df.head
```

Out[96]:
```
<bound method NDFrame.head of     outlook  temperature  humidity  wind answer
0         2            1         0     1     no
1         2            1         0     0     no
2         0            1         0     1    yes
3         1            2         0     1    yes
4         1            0         1     1    yes
5         1            0         1     0     no
6         0            0         1     0    yes
7         2            2         0     1     no
8         2            0         1     1    yes
9         1            2         1     1    yes
10        2            2         1     0    yes
11        0            2         0     0    yes
12        0            1         1     1    yes
13        1            2         0     0     no>
```

In [97]:
```
df.describe()
```

Out[97]:

	outlook	temperature	humidity	wind
count	14.000000	14.000000	14.000000	14.000000
mean	1.071429	1.142857	0.500000	0.571429
std	0.828742	0.864438	0.518875	0.513553
min	0.000000	0.000000	0.000000	0.000000
25%	0.250000	0.250000	0.000000	0.000000
50%	1.000000	1.000000	0.500000	1.000000
75%	2.000000	2.000000	1.000000	1.000000
max	2.000000	2.000000	1.000000	1.000000

In [98]:
```
df.isnull().sum()
```

Out[98]:
```
outlook        0
temperature    0
humidity       0
wind           0
answer         0
dtype: int64
```

In [99]:
```
X = df.iloc[:,0:4]
Y = df.iloc[:,-1]
```

In [100…

```
X
```

Out[100…

	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

In [101…
```
Y.shape
```

Out[101… (14,)

In [102…
```
X.shape
```

Out[102… (14, 4)

In [103…
```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.33, random
```

In [104…
```
Y_test.shape
```

Out[104… (5,)

In [105…
```
X_train.shape
```

Out[105… (9, 4)

In [106…
```
Y_train.shape
```

Out[106… (9,)

In [107…
```
df.describe()
```

Out[107…

	outlook	temperature	humidity	wind
count	14.000000	14.000000	14.000000	14.000000
mean	1.071429	1.142857	0.500000	0.571429
std	0.828742	0.864438	0.518875	0.513553
min	0.000000	0.000000	0.000000	0.000000
25%	0.250000	0.250000	0.000000	0.000000
50%	1.000000	1.000000	0.500000	1.000000
75%	2.000000	2.000000	1.000000	1.000000
max	2.000000	2.000000	1.000000	1.000000

In [108…
```
from sklearn.naive_bayes import GaussianNB
dt=GaussianNB()
```

In [109…
```
dt.fit(X_train,Y_train)
```

Out[109… GaussianNB

```
GaussianNB()
```

In [110…
```
Y_pred = dt.predict(X_test)
```

In [111…
```
dt.score(X_test,Y_test)
```

Out[111… 1.0

In [112…
```
import numpy as np
from sklearn.metrics import accuracy_score,f1_score,recall_score,precision_score
```

In [113…
```
print('Accuracy: %.3f' % accuracy_score(Y_test, Y_pred))
print('f1 score: %.3f' % f1_score(Y_test, Y_pred,average='micro'))
print('recall: %.3f' % recall_score(Y_test, Y_pred, average='macro'))
print('Precision: %.3f' % precision_score(Y_test, Y_pred,average='macro'))
```

```
Accuracy: 1.000
f1 score: 1.000
recall: 1.000
Precision: 1.000
```

In [114…
```
predicted_classes = dt.predict(X_test)
predicted_classes
```

Out[114… array(['yes', 'no', 'yes', 'yes', 'yes'], dtype='<U3')

In [ ]:

In [115…
```
df.describe()
```

Out[115…

	outlook	temperature	humidity	wind
count	14.000000	14.000000	14.000000	14.000000
mean	1.071429	1.142857	0.500000	0.571429
std	0.828742	0.864438	0.518875	0.513553
min	0.000000	0.000000	0.000000	0.000000
25%	0.250000	0.250000	0.000000	0.000000
50%	1.000000	1.000000	0.500000	1.000000
75%	2.000000	2.000000	1.000000	1.000000
max	2.000000	2.000000	1.000000	1.000000

In [116…
```
import numpy as np

user_input = []

outlook = int(input("Enter outlook (0 for sunny, 1 for overcast, 2 for rainy): "
temperature = int(input("Enter temperature (0 for cool, 1 for mild, 2 for hot):
humidity = int(input("Enter humidity (0 for normal, 1 for high): "))
wind = int(input("Enter wind (0 for weak, 1 for strong): "))

user_input.append([outlook, temperature, humidity, wind])
user_input = np.array(user_input)
predicted_classes = dt.predict(user_input)
print(predicted_classes)
```

```
['no']
```

```
C:\Users\nayan\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn
\base.py:493: UserWarning: X does not have valid feature names, but GaussianNB wa
s fitted with feature names
  warnings.warn(
```

With Vehicle Dataset

In [117…
```
df1 = pd.read_csv("vehicle.csv")
```

In [118…
```
df1
```

Out[118…

	number_plate	brand	color	time	stoled
0	N001	BMW	black	night	yes
1	N002	Audi	black	night	no
2	N003	NISSAN	black	night	yes
3	N004	VEGA	red	day	yes
4	N005	BMW	blue	day	no
5	N006	Audi	black	day	yes
6	N007	VEGA	red	night	no
7	N008	Audi	blue	day	yes
8	N009	VEGA	black	day	yes
9	N010	NISSAN	blue	day	no
10	N011	BMW	black	night	yes
11	N012	NISSAN	red	day	no
12	N013	VEGA	black	night	yes
13	N014	BMW	red	day	no
14	N015	Audi	black	day	yes
15	N016	Audi	blue	night	yes
16	N017	Audi	red	day	no
17	N018	NISSAN	black	day	yes
18	N019	BMW	blue	day	yes
19	N020	BMW	red	night	yes

In [119…
```
df1.isnull().sum()
```

Out[119…
```
number_plate    0
brand           0
color           0
time            0
stoled          0
dtype: int64
```

In [120…
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   number_plate  20 non-null     object
 1   brand         20 non-null     object
 2   color         20 non-null     object
 3   time          20 non-null     object
 4   stoled        20 non-null     object
dtypes: object(5)
memory usage: 932.0+ bytes
```

In [121…
```
df1.describe()
```

Out[121…

	number_plate	brand	color	time	stoled
count	20	20	20	20	20
unique	20	4	3	2	2
top	N001	BMW	black	day	yes
freq	1	6	9	12	13

In [122…
```
from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
```

In [123…
```
df1['number_plate'] = encoder.fit_transform(df1['number_plate'])

df1['brand'] = encoder.fit_transform(df1['brand'])

df1['color'] = encoder.fit_transform(df1['color'])

df1['time'] = encoder.fit_transform(df1['time'])
```

In [124…
```
df1
```

Out[124...

	number_plate	brand	color	time	stoled
0	0	1	0	1	yes
1	1	0	0	1	no
2	2	2	0	1	yes
3	3	3	2	0	yes
4	4	1	1	0	no
5	5	0	0	0	yes
6	6	3	2	1	no
7	7	0	1	0	yes
8	8	3	0	0	yes
9	9	2	1	0	no
10	10	1	0	1	yes
11	11	2	2	0	no
12	12	3	0	1	yes
13	13	1	2	0	no
14	14	0	0	0	yes
15	15	0	1	1	yes
16	16	0	2	0	no
17	17	2	0	0	yes
18	18	1	1	0	yes
19	19	1	2	1	yes

In [125...
```
X = df1.iloc[:,1:4]
Y = df1.iloc[:,-1]
```

In [126...
```
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, random_
```

In [127...
```
from sklearn.naive_bayes import GaussianNB
dt=GaussianNB()
```

In [128...
```
dt.fit(X_train,Y_train)
```

Out[128...

GaussianNB

```
GaussianNB()
```

In [129...
```
Y_pred = dt.predict(X_test)
```

In [130...
```
dt.score(X_test,Y_test)
```

Out[130… 0.8333333333333334

In [131…
```
import numpy as np
from sklearn.metrics import accuracy_score,f1_score,recall_score,precision_score
```

In [132…
```
print('Accuracy: %.3f' % accuracy_score(Y_test, Y_pred))
print('f1 score: %.3f' % f1_score(Y_test, Y_pred,average='micro'))
print('recall: %.3f' % recall_score(Y_test, Y_pred, average='macro'))
print('Precision: %.3f' % precision_score(Y_test, Y_pred,average='macro'))
```

```
Accuracy: 0.833
f1 score: 0.833
recall: 0.900
Precision: 0.750
```

In [133…
```
predicted_classes = dt.predict(X_test)
predicted_classes
```

Out[133… array(['yes', 'no', 'no', 'yes', 'yes', 'yes'], dtype='<U3')

In [134…
```
import numpy as np
user_input = []
brand=1
color=0
time=1
user_input.append([brand,color,time])

user_input = np.array(user_input)

predicted_classes = dt.predict(user_input)

print(f"predicted class: {predicted_classes}")
```

```
predicted class: ['yes']
C:\Users\nayan\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn
\base.py:493: UserWarning: X does not have valid feature names, but GaussianNB wa
s fitted with feature names
  warnings.warn(
```

In [135…
```
X.describe()
```

Out[135…

	brand	color	time
count	20.000000	20.000000	20.000000
mean	1.300000	0.850000	0.400000
std	1.128576	0.875094	0.502625
min	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000
50%	1.000000	1.000000	0.000000
75%	2.000000	2.000000	1.000000
max	3.000000	2.000000	1.000000

In [136…

```
import numpy as np
user_input = []

brand = int(input("Enter brand (0 for Audi, 1 for BMW, 2 for NISSAN, 3 for VEGA)
color = int(input("Enter color (0 for black, 1 for blue, 2 for red): "))
time = int(input("Enter time (0 for day, 1 for night): "))

user_input.append([brand,color,time])
user_input = np.array(user_input)
predicted_classes = dt.predict(user_input)
print(predicted_classes)
```

```
['no']
```

```
C:\Users\nayan\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn
\base.py:493: UserWarning: X does not have valid feature names, but GaussianNB wa
s fitted with feature names
  warnings.warn(
```