Exercise 8 – Naïve Bayes and KNN Classification

S.DEIVANAYAKI (21BCS003)

1. **Python Program to build Naïve Bayes Classification with random data points without API.**

```
import pandas as pd
import numpy as np

data = pd.DataFrame({'Outlook':
np.random.choice(['Sunny','Overcast','Rain'],150),'Temperature':
np.random.choice(['Hot','Cold','Mild'],150),'Humidity':
np.random.choice(['Normal', 'High'],150),'Wind': np.random.choice(['Weak',
'Strong'],150),'PlayChess': np.random.choice(['Yes','No'],150),})
data.head()
```

	Outlook	Temperature	Humidity	Wind	PlayChess
0	Sunny	Cold	High	Weak	No
1	Overcast	Mild	High	Strong	Yes
2	Sunny	Cold	High	Weak	Yes
3	Sunny	Mild	Normal	Strong	No
4	Sunny	Mild	Normal	Strong	Yes

```
pred=[]
for i in range(len(data.columns)-1):
    print("Specify",data.columns[i])
    pred.append(input())
```

```
Specify Outlook
 Sunny
Specify Temperature
 Hot
Specify Humidity
 Normal
```

```
Specify Wind
 Weak

pred

['Sunny', 'Hot', 'Normal', 'Weak']

p1=1
p2=1
for i in range(len(data.columns)-1):
    df=data[data.iloc[:,i]==pred[i]]
    p1 *= (df['PlayChess']=='Yes').sum()/(data['PlayChess']=='Yes').sum()
    p2 *= (df['PlayChess']=='No').sum()/(data['PlayChess']=='No').sum()

p1 *= (data['PlayChess']=='Yes').sum() / len(data)
p2 *= (data['PlayChess']=='No').sum() / len(data)

if p1>p2 :
    print("Yes")
else:
    print("No")

Yes
```

2. Python Program to build Naïve Bayes Classification with random data points with API.

```
from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()
data['Outlook']= label_encoder.fit_transform(data['Outlook'])
data['Temperature']= label_encoder.fit_transform(data['Temperature'])
data['Humidity']= label_encoder.fit_transform(data['Humidity'])
data['Wind']= label_encoder.fit_transform(data['Wind'])
data['PlayChess']= label_encoder.fit_transform(data['PlayChess'])
data.head()
```

	Outlook	Temperature	Humidity	Wind	PlayChess
0	2	0	0	1	0
1	0	2	0	0	1
2	2	0	0	1	1

3	2	2	1	0	0
4	2	2	1	0	1

```
X=data.iloc[:,:4]
y=data.iloc[:,4]
```

```
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_state=1)
```

```
from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
model.fit(X_train, y_train)
```

```
y_pred = model.predict(X_test)
from sklearn import metrics
print("Gaussian Naive Bayes model accuracy(in %):",
metrics.accuracy_score(y_test, y_pred)*100)
```

```
Gaussian Naive Bayes model accuracy(in %): 71.11111111111111
```

```
y_test[:5]
```

```
14     0
98     0
75     0
16     0
131    1
Name: PlayChess, dtype: int32
```

```
pred[:5]
```

```
array([1, 1, 0, 0, 0])
```

3. Python Program to build KNN Classification with random data points without API.

```
data = pd.DataFrame({'AadharId': np.random.choice(range(100),100),'Age':
np.random.choice(range(5,80),100),'Gender':
np.random.choice([0,1],100),'Class':
np.random.choice(['Cricket','Badminton','Kabadi','Tennis','Athelete'],100)})
data.head()
```

	AadharId	Age	Gender	Class
0	74	40	0	Cricket
1	3	12	1	Cricket
2	41	27	1	Cricket
3	18	77	0	Cricket
4	6	62	0	Tennis

```
pred=[]
for i in range(len(data.columns)-1):
    print("Specify",data.columns[i])
    pred.append(input())
pred
```

```
Specify AadharId
 5
Specify Age
 23
Specify Gender
 0
['5', '23', '0']
```

```
dist=((data['Age']-int(pred[1]))**2 + (data['Gender']-int(pred[2]))**2 ) **
0.5
dist
```

```
0     17.000000
1     11.045361
2      4.123106
3     54.000000
4     39.000000
        ...
95     1.000000
96    20.024984
97    15.000000
98    46.010868
99    36.000000
Length: 100, dtype: float64
```

```
import numpy as np
ind=np.argmin(dist)
ind
```

```
38
```

```
data.iloc[ind,3]
```

```
'Tennis'
```

4. Python Program to build KNN Classification with random data points with API.

```
from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()
data['Class']= label_encoder.fit_transform(data['Class'])
data.head()
```

	AadharId	Age	Gender	Class
0	74	40	0	2
1	3	12	1	2
2	41	27	1	2
3	18	77	0	2
4	6	62	0	4

```
X=data.iloc[:,:3]
y=data.iloc[:,3]
```

```
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_state=1)
```

```
from sklearn.neighbors import KNeighborsClassifier
knnmodel = KNeighborsClassifier(3)
knnmodel.fit(X_train,y_train)
```

```
pred = knnmodel.predict(X_test)
from sklearn import metrics
print("K Nearest Neighbor model accuracy(in %):",
metrics.accuracy_score(y_test, pred)*100)
```

```
K Nearest Neighbor model accuracy(in %): 16.666666666666664
```

```
df = pd.read_csv("C:/Users/DELL/Downloads/car.data.txt")
df.head()
```

	vhigh	vhigh.1	2	2.1	small	low	unacc
0	vhigh	vhigh	2	2	small	med	unacc
1	vhigh	vhigh	2	2	small	high	unacc
2	vhigh	vhigh	2	2	med	low	unacc
3	vhigh	vhigh	2	2	med	med	unacc
4	vhigh	vhigh	2	2	med	high	unacc

5. Python Program to build Naïve Bayes and KNN Classification with car evaluation dataset with API.

```
from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()
df['vhigh']= label_encoder.fit_transform(df['vhigh'])
df['vhigh.1']= label_encoder.fit_transform(df['vhigh.1'])
df['2']= label_encoder.fit_transform(df['2'])
df['2.1']= label_encoder.fit_transform(df['2.1'])
df['small']= label_encoder.fit_transform(df['small'])
df['low']= label_encoder.fit_transform(df['low'])
df['unacc']= label_encoder.fit_transform(df['unacc'])
df
```

	vhigh	vhigh.1	2	2.1	small	low	unacc
0	3	3	0	0	2	2	2
1	3	3	0	0	2	0	2
2	3	3	0	0	1	1	2
3	3	3	0	0	1	2	2
4	3	3	0	0	1	0	2
...	...	...	...	...	...	...	...

1722	1	1	3	2	1	2	1
1723	1	1	3	2	1	0	3
1724	1	1	3	2	0	1	2
1725	1	1	3	2	0	2	1
1726	1	1	3	2	0	0	3

1727 rows × 7 columns

```
X=df.iloc[:,:6]
y=df.iloc[:,6]
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_state=1)

ch=int(input("0 : Gaussian Naive Bayes\n1 : K-Nearest Neighbor\nYour Choice :
"))
if(ch==0):
    from sklearn.naive_bayes import GaussianNB
    model = GaussianNB()
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    from sklearn import metrics
    print("Gaussian Naive Bayes model accuracy(in %):",
metrics.accuracy_score(y_test, y_pred)*100)
else:
    from sklearn.neighbors import KNeighborsClassifier
    knnmodel = KNeighborsClassifier(3)
    knnmodel.fit(X_train,y_train)
    pred = knnmodel.predict(X_test)
    from sklearn import metrics
    print("K Nearest Neighbor model accuracy(in %):",
metrics.accuracy_score(y_test, pred)*100)
```

```
0 : Gaussian Naive Bayes
1 : K-Nearest Neighbor
 0


Gaussian Naive Bayes model accuracy(in %): 63.969171483622354
```