EX.NO:3 IMPLEMENT NAÏVE BAYES CLASSIFICATION AND PREDICT THE CLASS LABEL FOR A GIVEN DATA

Aim:

To write a python program to implement Naive Bayes Classification and predict the class label for the given dataset.

Naive Bayes Classification:

Naive Bayes is a statistical classification technique based on Bayes Theorem. It is one of the simplest supervised learning algorithms. Naive Bayes classifier is the fast, accurate and reliable algorithm. Naive Bayes classifiers have high accuracy and speed on large datasets.

Naive Bayes classifier assumes that the effect of a particular feature in a class is independent of other features. For example, a loan applicant is desirable or not depending on his/her income, previous loan and transaction history, age, and location. Even if these features are interdependent, these features are still considered independently. This assumption simplifies computation, and that's why it is considered as naive. This assumption is called class conditional independence.

$$P(D/h) = P(h/D)P(D)/P(h)$$

1. P(h): the probability of hypothesis h being true (regardless of the data). This is known as the prior probability of h.

2. P(D): the probability of the data (regardless of the hypothesis). This is known as the prior probability.

3. P(h|D): the probability of hypothesis h given the data D. This is known as posterior probability.

4. P(D|h): the probability of data d given that the hypothesis h was true. This is known as posterior probability.

Procedure:

1. Step 1: Calculate the prior probability for given class labels

2. Step 2: Calculate conditional probability with each attribute for each class

3. Step 3: Multiply same class conditional probability

4. Step 4: Multiply prior probability with step 3 probability

5. Step 5: See which class has a higher probability, given the input belongs to the higher probability class.

Sample Dataset

Weather	Temperature	Play
Sunny	Hot	No
Sunny	Hot	No
Overcast	Hot	Yes
Rainy	Mild	Yes
Rainy	Cool	Yes
Rainy	Cool	No
Overcast	Cool	Yes
Sunny	Mild	No
Sunny	Cool	Yes
Rainy	Mild	Yes
Sunny	Mild	Yes
Overcast	Mild	Yes
Overcast	Hot	Yes
Rainy	Mild	No

To calculate the probability of playing when the weather is overcast, and the temperature is mild.

Probability of playing:

P(Play= Yes | Weather=Overcast, Temp=Mild) = P(Weather=Overcast, Temp=Mild

| Play= Yes)P(Play=Yes)   (1)

P(Weather=Overcast, Temp=Mild | Play= Yes)= P(Overcast |Yes) P(Mild |Yes) (2)

1. Calculate Prior Probabilities: P(Yes)= 9/14 = 0.64
2. Calculate Posterior Probabilities: P(Overcast |Yes) = 4/9 = 0.44 P(Mild |Yes) = 4/9 = 0.44
3. Put Posterior probabilities in equation (2) P(Weather=Overcast, Temp=Mild | Play= Yes) =

   0.44 * 0.44 = 0.1936(Higher)
4. Put Prior and Posterior probabilities in equation (1) P(Play= Yes |

   Weather=Overcast, Temp=Mild) = 0.1936*0.64 = 0.124

Similarly, you can calculate the probability of not playing:

Probability of not playing:

P(Play= No | Weather=Overcast, Temp=Mild) = P(Weather=Overcast, Temp=Mild

| Play= No)P(Play=No)   (3)

P(Weather=Overcast, Temp=Mild | Play= No)= P(Weather=Overcast |Play=No)

P(Temp=Mild | Play=No) (4)

1. Calculate Prior Probabilities: P(No)= 5/14 = 0.36
2. Calculate Posterior Probabilities: P(Weather=Overcast |Play=No) = 0/9 = 0

   P(Temp=Mild | Play=No)=2/5=0.4
3. Put posterior probabilities in equation (4) P(Weather=Overcast, Temp=Mild | Play=

   No) = 0 * 0.4= 0
4. Put prior and posterior probabilities in equation (3) P(Play= No |

   Weather=Overcast, Temp=Mild) = 0*0.36=0

The probability of a 'Yes' class is higher. So you can say here that if the weather is overcast than players will play the sport.

Sample Implementation:

```
weather=['Sunny','Sunny','Overcast','Rainy','Rainy','Rainy','Overcast','Sunny','Sunny',
'Rainy','Sunny','Overcast','Overcast','Rainy']
temp=['Hot','Hot','Hot','Mild','Cool','Cool','Cool','Mild','Cool','Mild','Mild','Mild','Hot','Mild']

play=['No','No','Yes','Yes','Yes','No','Yes','No','Yes','Yes','Yes','Yes','Yes','No']
# Import LabelEncoder
from sklearn import preprocessing
#creating labelEncoder
le = preprocessing.LabelEncoder()
```

```
# Converting string labels into numbers.
weather_encoded=le.fit_transform(weather)
print (weather_encoded)
# Converting string labels into numbers
temp_encoded=le.fit_transform(temp)
label=le.fit_transform(play)
print ("Temp:",temp_encoded)
print ("Play:",label)
#Combinig weather and temp into single listof tuples
features=zip(weather_encoded,temp_encoded)
features=list(features)
print (features)
#Import Gaussian Naive Bayes model
from sklearn.naive_bayes import GaussianNB

#Create a Gaussian Classifier
model = GaussianNB()

# Train the model using the training sets
model.fit(features,label)

#Predict Output
predicted= model.predict([[0,2]]) # 0:Overcast, 2:Mild
print ("Predicted Value:", predicted)
```

Sample Output:

```
[2 2 0 1 1 1 0 2 2 1 2 0 0 1]
Temp: [1 1 1 2 0 0 0 2 0 2 2 2 1 2]
Play: [0 0 1 1 1 0 1 0 1 1 1 1 1 0]
[(2, 1), (2, 1), (0, 1), (1, 2), (1, 0), (1, 0), (0, 0), (2, 2), (2, 0), (1, 2), (2, 2), (0, 2), (0, 1), (1, 2)]
Predicted Value: [1]
```

Dataset:

Student	Maths	Science	English
A	78	82	75
B	85	88	80
C	67	72	70
D	90	91	89
E	74	76	73

Implementation:

```
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
data = pd.read_csv("Student_Marks.csv")
print("Dataset Preview:")
print(data.head())
X = data.iloc[:, 1].values.reshape(-1, 1)
y = data.iloc[:, 3].values
print("\nMaths Marks (X):")
print(X[:5])
print("\nEnglish Marks (y):")
print(y[:5])
model = LinearRegression()
model.fit(X, y)
future_maths = np.array([[88]])
predicted_marks = model.predict(future_maths)
print("\nPredicted English Marks for Maths = 88:")
print(predicted_marks)
```

Output:

```
Maths Marks (X):
[[78]
 [85]
 [67]
 [90]
 [74]]

English Marks (y):
[75 80 70 89 73]
Predicted English Marks for Maths = 88: [84.9]
```

MARK ALLOCATION	
Conduct of Experiment(30)	
Record Observation (20)	
Viva (10)	
Total (60)	
Signature of the Faculty with Date	

Result:

Thus Naive Bayes Classification algorithm is implemented and predicted the class label for the given dataset.