Ex. No.: 12 Write a program for Naïve Baye's Algorithm

AIM:

To write a Python program to calculate the Naïve Baye's Algorithm by using the given series of files.

What is your age? (Age)	Which roads in Thoothukudi are the most dangerous for bike riders? Why? (Zone)		Are hospitals easily accessible from accident-prone highways? (Hospital)	
16	Bridges	1	YES	1
18	Thoothukudi - Madurai	2	YES	1
19	Thoothukudi - Madurai	2	YES	1
20	Bridges	1	YES	1
21	Tiruchendur - Thoothukudi	3	NO	0
22	Thoothukudi - Ramanathapuram	4	YES	1
23	Bridges	1	NO	0
24	Thoothukudi - Tirunelveli	5	YES	1
25	Bridges	1	YES	1
26	Thoothukudi - Tirunelveli	5	NO	0
27	Bridges	1	YES	1
28	Bridges	1	NO	0
29	Bridges	1	NO	0
30	Bridges	1	YES	1
31	Thoothukudi - Ramanathapuram	4	YES	1
32	Bridges	1	YES	1
33	Thoothukudi - Tirunelveli	5	YES	1
34	Bridges	1	YES	1
35	Bridges	1	YES	1
36	Bridges	1	NO	0
38	Thoothukudi - Tirunelveli	5	YES	1
39	Thoothukudi - Madurai	2	YES	1
42	Thoothukudi - Tirunelveli	5	YES	1
43	Bridges	1	YES	1
44	Tiruchendur - Thoothukudi	3	YES	1

Program:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score
data = {
    'Age':
[16,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,38,39,42,43
,44],
    'Zone': [1,2,2,1,3,4,1,5,1,5,1,1,1,1,4,1,5,1,1,1,5,2,5,1,3],
    'Hospital': [1,1,1,1,0,1,0,1,1,0,1,0,0,1,1,1,1,1,1,1,0,1,1,1,1,1] }
df = pd.DataFrame(data)
X = df[['Age', 'Zone']]
y = df['Hospital']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
model = GaussianNB()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy:.2f}")
new_person = pd.DataFrame({'Age': [17], 'Zone': [2]})
prediction = model.predict(new_person)
print(f"\nPrediction for new person (Age: 17, Zone: 2): {'Hospital' if
prediction[0] == 1 else 'No Hospital'}")
```

Output:

Accuracy: 0.80

Prediction for new person (Age: 17, Zone: 2): Hospital

test_size=0.3: This means 30% of the data will be used for testing the model and the remaining 70% will be used for training.

random_state is a parameter used in functions like train_test_split to ensure reproducibility by setting a seed for the random shuffling process.

When you specify a random_state (e.g., random_state=42), the split of data into training and test sets will be the same every time you run the code.

If you don't specify random_state, the data split will be different each time, leading to potentially different results.

Choosing a Value for Random State- Any integer works: 42 is a common choice (popularized by Douglas Adams' "Hitchhiker's Guide to the Galaxy"), but you can use any integer (like 0, 1, 123, etc.).

Reproducibility: Use a fixed random_state if you want others to reproduce your exact results.

Experimentation: If you're experimenting and don't care about reproducibility between runs, you might omit it.

Considerations- For reporting results in research or production, it's good practice to specify random_state for reproducibility.

There's nothing inherently "better" about 42 vs another number; it's about consistency.

In your case, random_state=42 is as good as any other fixed integer for ensuring reproducibility.