## 7.Develop a program to implement Naive Bayes classifier model and analyze the model using confusion matrix

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn import metrics
import seaborn as sns
```

```
data=pd.read_excel("Navietext.xlsx",names=['Message','Label'])
print("Dataset:\n", data)
```

```
Dataset:
                                   Message Label
0                This is an amazing place   pos
1      I feel very good about these beers   pos
2                    This is my best work   pos
3                    What an awesome view   pos
4           I do not like this restaurant   neg
5                I am tired of this stuff   neg
6                  I can’t deal with this   neg
7                    He is my sworn enemy   neg
8                     My boss is horrible   neg
9                This is an awesome place   pos
10  I do not like the taste of this juice   neg
11                        I love to dance   pos
12      I am sick and tired of this place   neg
13                   What a great holiday   pos
14         That is a bad locality to stay   neg
15         We will have good fun tomorrow   pos
16       I went to my enemy’s house today   neg
```

```
print('The dimensions of the dataset',data.shape)
```

```
The dimensions of the dataset (17, 2)
```

```
data['Labelnum']=data.Label.map({'pos':1,'neg':0})
x=data.Message
y=data.Labelnum
print(x)
print(y)


vectorizer=TfidfVectorizer()
data=vectorizer.fit_transform(x)
data
```

```
#splitting the dataset into train and test data
print("\n Train Test Split:\n")
xtrain,xtest,ytrain,ytest=train_test_split(data,y,test_size=0.3,random_state=2)
print('\n The total number of Training Data:',ytrain.shape)
print('\n The total number of Test Data:',ytest.shape)
```

```
 Train Test Split:
```

```
 The total number of Training Data: (11,)
 The total number of Test Data: (6,)
```

```
#Training Navie Bayes classifier on training data
#ytest= actual label
clf= MultinomialNB().fit(xtrain, ytrain)
predicted = clf.predict(xtest)
#printing accuracy, Confusion matrix, Precision and Recall
print("\n Accuracy of the classifier is:", metrics.accuracy_score(ytest,predicted))
print("\nConfusion Matrix is:", metrics.confusion_matrix(ytest,predicted))
```

```
Accuracy of the classifier is: 0.6666666666666666

Confusion Matrix is: [[3 0]
 [2 1]]
```

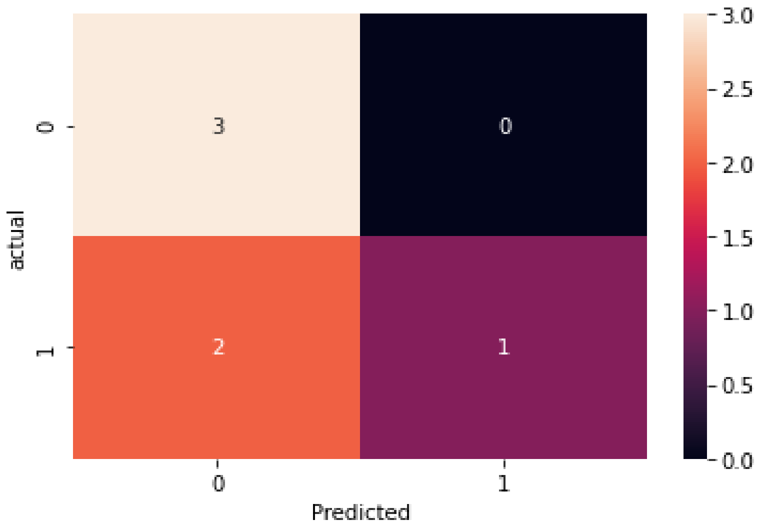
```
predicted
```

```
array([0, 0, 0, 0, 0, 1], dtype=int64)
```

```
ytest #actual
```

```
12    0
4     0
5     0
0     1
9     1
3     1
```

```
cnf_matrix=pd.crosstab(ytest,predicted, rownames=['actual'], colnames=['Predicted'])
sns.heatmap(cnf_matrix, annot=True)
```


actual
Predicted