1. EXPERIMENT NO: 6

2. TITLE: **DOCUMENT CLASSIFICATION USING NAÏVE BAYESIAN CLASSIFIER**

3. LEARNING OBJECTIVES:

- Make use of Data sets in implementing the machine learning algorithms.
- Implement ML concepts and algorithms in Python

4. AIM:

- Assuming a set of documents that need to be classified, use the naïve Bayesian Classifier model to perform this task. Built-in Java classes/API can be used to write the program. Calculate the accuracy, precision, and recall for your data set.

5. THEORY:

For the theoey of the naive bayesian classifier refer Experiment No. 5. Theory of performance anaysis analysis is ellaborated here.

Analysis of Document Classification

		Predicted	
		Negative	**Positive**
Actual	**Negative**	True Negative	False Positive
	Positive	False Negative	True Positive

- For classification tasks, the terms true positives, true negatives, false positives, and false negatives compare the results of the classifier under test with trusted external judgments. The terms positive and negative refer to the classifier's prediction (sometimes known as the expectation), and the terms true and false refer to whether that prediction corresponds to the external judgment (sometimes known as the observation).
- Precision - Precision is the ratio of correctly predicted positive documents to the total predicted positive documents. High precision relates to the low false positive rate.

$$\text{Precision} = (\Sigma \text{ True positive}) / (\Sigma \text{ True positive} + \Sigma \text{ False positive})$$

- Recall (Sensitivity) - Recall is the ratio of correctly predicted positive documents to the all observations in actual class.

$$\text{Recall} = (\Sigma \text{ True positive}) / (\Sigma \text{ True positive} + \Sigma \text{ False negative})$$

- Accuracy - Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations. One may think that, if we have high accuracy then our model is best. Yes, accuracy is a great measure but only when you have symmetric datasets where values of false positive and false negatives are almost same. Therefore, you have to look at other parameters to evaluate the performance of your model. For our model, we have got 0.803 which means our model is approx. 80% accurate.

$$\text{Accuracy} = (\Sigma \text{ True positive} + \Sigma \text{ True negative}) / \Sigma \text{ Total population}$$

6. PROCEDURE / PROGRAMME :

```
import pandas as pd
msg=pd.read_csv('data6.csv',names=['message','label']) #Tabular form data
print('Total instances in the dataset:',msg.shape[0])

msg['labelnum']=msg.label.map({'pos':1,'neg':0})
X=msg.message
Y=msg.labelnum

print('\nThe message and its label of first 5 instances are listed below')
X5, Y5 = X[0:5], msg.label[0:5]
for x, y in zip(X5,Y5):
    print(x,',',y)

# Splitting the dataset into train and test data
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest=train_test_split(X,Y)
print('\nDataset is split into Training and Testing samples')
print('Total training instances :', xtrain.shape[0])
print('Total testing  instances :', xtest.shape[0])

# Output of count vectoriser is a sparse matrix
# CountVectorizer - stands for 'feature extraction'
from sklearn.feature_extraction.text import CountVectorizer
count_vect = CountVectorizer()
xtrain_dtm = count_vect.fit_transform(xtrain) #Sparse matrix
xtest_dtm  = count_vect.transform(xtest)
print('\nTotal features extracted using CountVectorizer:',xtrain_dtm.shape[1])

print('\nFeatures for first 5 training instances are listed below')
df=pd.DataFrame(xtrain_dtm.toarray(),columns=count_vect.get_feature_names())
print(df[0:5])#tabular representation
#print(xtrain_dtm) #Same as above but sparse matrix representation

# Training Naive Bayes (NB) classifier on training data.
from sklearn.naive_bayes import MultinomialNB
clf = MultinomialNB().fit(xtrain_dtm,ytrain)
predicted = clf.predict(xtest_dtm)

print('\nClassstification results of testing samples are given below')
for doc, p in zip(xtest, predicted):
   pred = 'pos' if p==1 else 'neg'
   print('%s -> %s ' % (doc, pred))

#printing accuracy metrics
from sklearn import metrics
print('\nAccuracy metrics')
print('Accuracy of the classifer is',metrics.accuracy_score(ytest,predicted))

print('Recall   :',metrics.recall_score(ytest,predicted),
        '\nPrecison :',metrics.precision_score(ytest,predicted))
print('Confusion matrix')
print(metrics.confusion_matrix(ytest,predicted))
```

7. RESULTS & CONCLUSIONS:

Data set

I love this sandwich,pos
This is an amazing place,pos
I feel very good about these beers,pos
This is my best work,pos
What an awesome view,pos
I do not like this restaurant,neg
I am tired of this stuff,neg
I can't deal with this,neg
He is my sworn enemy,neg
My boss is horrible,neg
This is an awesome place,pos
I do not like the taste of this juice,neg
I love to dance,pos
I am sick and tired of this place,neg
What a great holiday,pos
That is a bad locality to stay,neg
We will have good fun tomorrow,pos
I went to my enemy's house today,neg

Output

Total instances in the dataset: 18

The message and its label of first 5 instances are listed below
I love this sandwich , pos
This is an amazing place , pos
I feel very good about these beers , pos
This is my best work , pos
What an awesome view , pos

Dataset is split into Training and Testing samples
Total training instances : 13
Total testing instances : 5

Total features extracted using CountVectorizer: 46

Features for first 5 training instances are listed below

	am	amazing	an	and	awesome	bad	...	view	we	went	what	will	with
0	1	0	0	1	0	0	...	0	0	0	0	0	0
1	0	0	0	0	0	0	...	0	0	0	0	0	0
2	0	0	1	0	1	0	...	1	0	0	1	0	0
3	0	1	1	0	0	0	...	0	0	0	0	0	0
4	0	0	0	0	0	1	...	0	0	0	0	0	0

Classstification results of testing samples are given below
This is an awesome place -> pos
I love this sandwich -> pos
I love to dance -> pos
This is my best work -> pos
I feel very good about these beers -> pos

Accuracy metrics
Accuracy of the classifer is 0.4
Recall : 0.4
Precison : 1.0
Confusion matrix

```
[[0 0]
 [3 2]]
```

8. LEARNING OUTCOMES :

- The student will be able to apply naive baysian classifier for document classification and analyse the results.

9. APPLICATION AREAS:

- Applicable in document classification

10. REMARKS: