**Experiment-10**

**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 dataset.**

import pandas as pd

msg = pd.read\_csv('E:/data.csv', names=['message', 'label'])

print('The dimensions of the dataset:', msg.shape)

# Check the dataset before dropping rows with missing values

print(msg)

msg.dropna(subset=['label'], inplace=True) # Drop rows with missing label

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

# Check the dataset after mapping label to labelnum

print(msg)

msg.dropna(subset=['labelnum'], inplace=True) # Drop rows with missing labelnum

print(msg)

X = msg.message

y = msg.labelnum

# Check if the series X and y are non-empty

print(X)

print(y)

from sklearn.model\_selection import train\_test\_split

if len(X) > 0 and len(y) > 0:

x\_train, x\_test, y\_train, y\_test = train\_test\_split(X, y)

print(x\_test.shape)

print(x\_train.shape)

print(y\_test.shape)

print(y\_train.shape)

from sklearn.feature\_extraction.text import CountVectorizer

count\_vect = CountVectorizer()

x\_train\_dtm = count\_vect.fit\_transform(x\_train)

x\_test\_dtm = count\_vect.transform(x\_test)

print(count\_vect.get\_feature\_names\_out())

df = pd.DataFrame(x\_train\_dtm.toarray(), columns=count\_vect.get\_feature\_names\_out())

print(df)

from sklearn.naive\_bayes import MultinomialNB

clf = MultinomialNB().fit(x\_train\_dtm, y\_train)

predicted = clf.predict(x\_test\_dtm)

from sklearn import metrics

print('Accuracy metrics:')

print('Accuracy of the classifier is', metrics.accuracy\_score(y\_test, predicted))

print('Confusion matrix:')

print(metrics.confusion\_matrix(y\_test, predicted))

print('Recall and Precision:')

print(metrics.recall\_score(y\_test, predicted))

print(metrics.precision\_score(y\_test, predicted))

else:

print("The dataset is empty after cleaning. Please check your input data.")

Output:

The dimensions of the dataset: (18, 2)

message label

0 message,label NaN

1 I love this sandwich,pos NaN

2 This is an amazing place,pos NaN

3 I feel very good about these beers,pos NaN

4 What an awesome view,pos NaN

5 I do not like this restaurant,neg NaN

6 I am tired of this stuff,neg NaN

7 I can't deal with this,neg NaN

8 He is my sworn enemy,neg NaN

9 My boss is horrible,neg NaN

10 This is an awesome place,pos NaN

11 I do not like the taste of this juice,neg NaN

12 I love to dance,pos NaN

13 I am sick and tired of this place,neg NaN

14 What a great holiday,pos NaN

15 That is a bad locality to stay,neg NaN

16 We will have good fun tomorrow,pos NaN

17 I went to my enemy's house today,neg NaN

Empty DataFrame

Columns: [message, label, labelnum]

Index: []

Empty DataFrame

Columns: [message, label, labelnum]

Index: []

Series([], Name: message, dtype: object)

Series([], Name: labelnum, dtype: int64)

The dataset is empty after cleaning. Please check your input data.