7. Assuming a set of document that need to be classified, use the naive 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.

```
import pandas as pd
msg = pd.read_csv('lab6.csv', names=['message', 'label'])
print('Total instances in the data set: '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)

from sklearn.model_selection import train_test_split
xtrain, xtest, ytrain, ytest = train_test_split(X,Y)
print('Dataset is split into Training and Testing
        samples')
Print('Total training instances: ', xtrain.shape[0])
print('Total testing instances: ', xtest.shape[0])

from sklearn.feature_extraction.text import CountVectorizer
count_vect = CountVectorizer()
```

```
xtrain_dtm = count_vect.fit_transform(xtrain)
xtest_dtm = count_vect.transform(xtest)
print('\nTotal features extracted using CountVectorizer:',
     xtrain_dtm.shape[1])
print('\n Features for first 5 training instance are listed
     below')
df = pd.DataFrame(xtrain_dtm.toarray(), columns = count_
     vect.get_feature_names())
print(df[0:5])

from sklearn.naive_bayes import MultinomialNB
df = MultiNominalNB().fit(xtrain_dtm, ytrain)
predicted = df.predict(xtest_dtm)
print('\nClassification 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))

from sklearn import metrics
print('\nAccuracy metrics')
print('\nAccuracy of the classifier is', metrics.accuracy_
      score(ytest, predicted))
print('Recall: ', metrics.recall_score(ytest, predicted))
print('Precision: ', metrics.precision_score(ytest, predicted))
print('Confusion matrix')
print(metrics.confusion_matrix(ytest, predicted))
```

Output:-

```
Total instances in the data set: 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 instance are listed below.
   about  am  an  awesome  beers  best  boss  can  deal  do ... today
0    0     0   0     0       0      0     0     0    0    1 ...   0
1    0     0   0     0       0      0     0     0    0    0 ...   1
2    0     0   0     0       0      0     0     1    1    0 ...   0
3    0     0   1     1       0      0     0     0    0    0 ...   0
4    0     0   0     0       0      0     0     0    0    0 ...   0
```

```
   tomorrow  very  view  we  went  what  will  with  work
0     0       0     0     0    0     0     0     0     0
1     0       0     0     0    1     0     0     0     0
2     0       0     0     0    0     0     0     1     0
3     0       0     0     0    0     0     0     0     0
4     0       0     0     0    0     0     0     0     0

[5 rows x 46 columns]


Classifecation result of testing samples are given below
I love to dance -> pos
I am sick and tired of this place->neg
This is an amazing place -> pos
What a great holiday -> pos
This is a bad locality to stay -> neg

Accuracy metrics
Accuracy of the classifier is 1.0
Recall : 1.0
Precision : 1.0
Confusion matrix
[[2 0]
 [0 3]]
```

Dataset used:

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