* Assuming a set 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 dataset.

*

```
import pandas as pd
msg = pd.read_csv('c:/Users/Mili/OneDrive/Desktop/Mile/4MT17CS058_
                    Milind/lab6.csv') names=['message','label']
print("Total instance in the dataset:", msg.shape[0])

msg['labelname'] = msg.label.map({'pos':1,'neg':0})
x=msg.message
y= msg.labelnam
print("\n The message and its label of first 5 instance 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("Data set is split into Training and Testing samples")
print("Total training instance:", xtrain.shape[0])
print("Total testing instance:", xtest.shape[0])
```

Teacher's Signature _____

```
from sklearn.feature_extraction.text import CountVectorizer
count_vect = CountVectorizer()
xtrain_dtm = count_vect.fit_transform(xtrain)
xtest_dtm = count_vect.transform(xtest)
print("\n Total features extracted extracting CountVectorizer:",xtrain_dtm.
                                                    shape[1])
print("\n Features for first 5 having 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
clf = MultinomialNB().fit(xtrain_dtm, ytrain)
predict("\n Classifications results of testing sample 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("\n Accuracy matrices")
print("\n Accuracy 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))
```

Teacher's Signature _____

output

Total instance in the dataset : 18

The message and its label of first 5 instance 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 in Training & Testing Sample

Total training instance : 13

Total testing instance : 5

Total Feature extracted using countVectorizer : 46

Feature for first 5 training instance are listed below

	about	am	an	awesome	beer	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	are	coast	what	coill	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]

Classfication result of testing sample are gives below:

I love to dance → pos

Iam 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 metrices

Accuracy of the Classifier is 1.0

Recall : 1.0

Precision: 1.0

Confusion matrix :

[[2 0]

[0 3]]