Date .....

Expt. No. ....7.... Page No. ....36....

Assuming a set of documents 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 program. You can use java/python ML library classes / API.

```
import pandas as pd
msg = pd.read_csv('C:\\Users\\P V Bhat\\Desktop\\
        csvfiles\\sampledata6.csv', names=['message',
                                           'label'])
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('In The 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('In 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('\n Total features extracted using CountVec-
                                    torizer: ', xtrain_dtm.shape[1])
print('\n Features for first 5 training instances 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)
predicted = clf.predict(xtest_dtm)
print('\n Classification 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('\n Accuracy metrics')
print('Accuracy of the classifier is', metrics.accuracy_
                                    score(ytest, predicted))
print('Recall:', metrics.recall_score(ytest, predicted),
      '\n Precision:', metrics.precision_score(ytest, predicted))
print('Confusion matrix')
print(metrics.confusion_matrix(ytest, predicted))
```

Dataset :

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 awsome 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
This is a bad location 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 : 19

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 proud about these bees, pos
This is my best work. pos
what an awsome view. pos

Dataset is split into Training and Testing Samples
Total training instances : 13
Total testing instances : 5
Total features extracted using CountVectorizer : 39
Features for first 5 training instances are listed below

	am	amazing	an	and	awesome	bad	but	can	dance	...
0	0	0	1	0	1	0	0	0	0	...
1	0	0	0	0	0	0	0	0	0	...
2	0	0	0	0	0	0	0	0	0	...
3	0	0	1	0	1	0	0	0	0	...
4	0	1	1	0	0	0	0	0	0	...

	the	this	!
	0	1	
	0	0	
	1	1	
	0	0	
	0	1	

	tired	to	today	view	went	what	with	work
0	0	0	0	0	0	0	0	0
1	0	1	1	0	1	0	0	0
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0

[5 rows x 38 columns]

Classification results of testing samples are given below

I feel very good about these beers → neg

my boss is horrible → pos

He is my sworn enemy → pos

We will have good fun tomorrow → neg

what a great holiday → pos

Accuracy metrics

Accuracy of the classifier is 0.2

Recall : 0.33333333333333333333

Precision : 0.33333333333333333333

confusion matrix

[[0 2]

[2 1]]