

## Text-Classifier

### Code

#### Program-6

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 the program. Calculate the accuracy, precision, and recall for your data set.

```
import pandas as pd
msg = pd.read_csv('naivetest.txt', names=['message', 'label'])
print('The dimensions of the dataset', msg.shape)
```

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

```
X = msg.message
```

```
y = msg.labelnum
```

```
print(X)
```

```
print(y)
```

```
from sklearn.model_selection import train_test_split
xtrain, xtest, ytrain, ytest = train_test_split(X, y)
```

```
print(xtrain.shape)
```

```
print(xtest.shape)
```

```
print(ytrain.shape)
```

```
print(ytest.shape)
```

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

```
xtrain_dtm = count_vect.fit_transform(xtrain)
```

```
xtest_dtm = count_vect.transform(xtest)
```

```
print(count_vect.get_feature_names())
```

```
df = pd.DataFrame(xtrain_dtm.toarray(), columns=
count_vect.get_feature_names())
```

```
2. doc, category in zip(docs_news, predicted_news):  
    print('%s → %s' % (doc, category))
```

## OUTPUT

### 1) Positive-negative sentence output

The first screenshot shows the Jupyter Notebook interface with the following code and output:

```
In [24]: # 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)
```

```
In [25]: #printing accuracy metrics
from sklearn import metrics
print('Accuracy metrics')
print('Accuracy of the classifier is',metrics.accuracy_score(ytest,predicted))
print('Confusion matrix')
print(metrics.confusion_matrix(ytest,predicted))
print('Recall and Precision ')
print(metrics.recall_score(ytest,predicted))
print(metrics.precision_score(ytest,predicted))
```

Accuracy metrics  
Accuracy of the classifier is 0.6  
Confusion matrix  
[[2 2]  
 [0 1]]  
Recall and Precision  
1.0  
0.3333333333333333

The second screenshot shows the continuation of the notebook with the following code and output:

```
print(metrics.precision_score(ytest,predicted))
```

Accuracy metrics  
Accuracy of the classifier is 0.6  
Confusion matrix  
[[2 2]  
 [0 1]]  
Recall and Precision  
1.0  
0.3333333333333333

```
In [27]: docs_new = ['I like this place', 'My boss is not my saviour']
X_new_counts = count_vect.transform(docs_new)
predictednew = clf.predict(X_new_counts)
print(predictednew)
for doc, category in zip(docs_new, predictednew):
    print('%s->%s' % (doc, category))
```

[1 0]  
I like this place->1  
My boss is not my saviour->0

### 2) Corona-NLP dataset output



The image displays two screenshots of a Jupyter Notebook interface, showing the training and testing of a Naive Bayes classifier on the MLL-6-Corona dataset.

**Top Screenshot:**

- Cell [48]:** Training Naive Bayes (NB) classifier on training data.

```
# 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)
```

- Cell [49]:** Printing accuracy metrics.

```
#printing accuracy metrics
from sklearn import metrics
print('Accuracy metrics')
print('Accuracy of the classifier is',metrics.accuracy_score(ytest,predicted))
print('Confusion matrix')
print(metrics.confusion_matrix(ytest,predicted))
print('Recall and Precision ')
print(metrics.recall_score(ytest,predicted,average="macro"))
print(metrics.precision_score(ytest,predicted,average="macro"))
```

Output for Cell [49]:

```
Accuracy metrics
Accuracy of the classifier is 0.45840621963070943
Confusion matrix
[[ 277  832  11  207   8]
 [ 79 1368  56  858  55]
 [ 14  443  513  936  40]
 [ 19  578  85 2060  216]
 [ 1  123   9 1003  499]]
Recall and Precision
0.40778975078914437
0.5794584829274766
```

- Cell [69]:** Testing the classifier on new data.

```
docs_new = ['I like this place','I am becoming very negative in life','Its beautiful','Sky is ugly']
X_new_counts = count_vect.transform(docs_new)
predictednew = clf.predict(X_new_counts)
```

**Bottom Screenshot:**

  - Cell [69]:** Testing the classifier on new data (continued).

```
docs_new = ['I like this place','I am becoming very negative in life','Its beautiful','Sky is ugly']
X_new_counts = count_vect.transform(docs_new)
predictednew = clf.predict(X_new_counts)
print(predictednew)
for doc, category in zip(docs_new, predictednew):
    print('%s->%s' % (doc, category))
```

Output for Cell [69]:

```
[ 1 -2  2 -1]
I like this place->1
I am becoming very negative in life->-2
Its beautiful->2
Sky is ugly->-1
```

### 3) Ham-Spam dataset output

Acharya Live | Dig | MLL/ MLL-6 - Jupyter N MLL-6-HamSpam MLL-6-Corona - J (2) WhatsApp MLL-6 - Google D

localhost:8888/notebooks/MLL/MLL-6-HamSpam.ipynb

jupyter MLL-6-HamSpam Last Checkpoint: an hour ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

```
In [34]: # 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)

In [35]: #printing accuracy metrics
from sklearn import metrics
print('Accuracy metrics')
print('Accuracy of the classifier is',metrics.accuracy_score(ytest,predicted))
print('Confusion matrix')
print(metrics.confusion_matrix(ytest,predicted))
print('Recall and Precision ')
print(metrics.recall_score(ytest,predicted))
print(metrics.precision_score(ytest,predicted))

Accuracy metrics
Accuracy of the classifier is 0.9842067480258435
Confusion matrix
[[ 149  18]
 [  4 1222]]
Recall and Precision
0.9967373572593801
0.9854838709677419

In [37]: docs_new = ['This is a mail', 'Free gift for you']
X_new_counts = count_vect.transform(docs_new)
predictednew = clf.predict(X_new_counts)
print(predictednew)
for doc, category in zip(docs_new, predictednew):
    print('%s->%s' % (doc, category))
```

Type here to search

ENG US 13:08 04-11-2020

Acharya Live | Dig | MLL/ MLL-6 - Jupyter N MLL-6-HamSpam MLL-6-Corona - J (2) WhatsApp MLL-6 - Google D

localhost:8888/notebooks/MLL/MLL-6-HamSpam.ipynb

jupyter MLL-6-HamSpam Last Checkpoint: an hour ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

```
print(metrics.precision_score(ytest,predicted))

Accuracy metrics
Accuracy of the classifier is 0.9842067480258435
Confusion matrix
[[ 149  18]
 [  4 1222]]
Recall and Precision
0.9967373572593801
0.9854838709677419

In [37]: docs_new = ['This is a mail', 'Free gift for you']
X_new_counts = count_vect.transform(docs_new)
predictednew = clf.predict(X_new_counts)
print(predictednew)
for doc, category in zip(docs_new, predictednew):
    print('%s->%s' % (doc, category))

[1 0]
This is a mail->1
Free gift for you->0

In [ ]:
```

Type here to search

ENG US 13:08 04-11-2020