Machine Learning Lab – 20ISL68A

Program 2 – For or a given set of training data examples stored in a .CSV file, implement and demonstrate the Document classifier using Naiive Bayes.

Step 1: Importing Libraries

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn import metrics
```

Step 2: Understanding Datasets

```
data=pd.read_excel(r"C:\Users\kvsuv\OneDrive\Desktop\dataset.csv.xlsx",
                   names=['Message','Label'])
print("Dataset:\n", data)
```

```
Dataset:
                                    Message Label
0                This is an amazing place   pos
1      I feel very good about these beers   pos
2                    This is my best work   pos
3                    What an awesome view   pos
4           I do not like this restaurant   neg
5                I am tired of this stuff   neg
6                  I can’t deal with this   neg
7                    He is my sworn enemy   neg
8                     My boss is horrible   neg
9                This is an awesome place   pos
10  I do not like the taste of this juice   neg
11                          I love to dance   pos
12      I am sick and tired of this place   neg
13                   What a great holiday   pos
14         That is a bad locality to stay   neg
15         We will have good fun tomorrow   pos
16       I went to my enemy’s house today   neg
```

Step 3: Printing Dimensions of data set

```
print('The dimensions of the dataset',data.shape)
```

```
The dimensions of the dataset (17, 2)
```

Step 4: Converting the labels to numerical data

```
data['Labelnum']=data.Label.map({'pos':1,'neg':0})
x=data.Message
y=data.Labelnum
print(x)
print(y)
```

```
0                 This is an amazing place
1        I feel very good about these beers
2                     This is my best work
3                     What an awesome view
4             I do not like this restaurant
5                  I am tired of this stuff
6                    I can’t deal with this
7                     He is my sworn enemy
8                      My boss is horrible
9                 This is an awesome place
10    I do not like the taste of this juice
11                          I love to dance
12        I am sick and tired of this place
13                    What a great holiday
14           That is a bad locality to stay
15           We will have good fun tomorrow
16         I went to my enemy’s house today
Name: Message, dtype: object
0     1
1     1
2     1
3     1
4     0
5     0
6     0
7     0
8     0
9     1
10    0
11    1
12    0
13    1
14    0
15    1
16    0
Name: Labelnum, dtype: int64
```

Step 5: Converting the message to numerical data

```
vectorizer=TfidfVectorizer()
data=vectorizer.fit_transform(x)
```

Step 6: Displaying TFIDF features

```
print("\n The TFIDF features of Dataset:\n")
df=pd.DataFrame(data.toarray(), columns=vectorizer.get_feature_names())
df.head()
```

The TFIDF features of Dataset:

	about	am	amazing	an	and	awesome	bad	beers	best	boss	...	today	tomorrow	very	view	we	went	what	will	with
0	0.000000	0.0	0.589549	0.461737	0.0	0.000000	0.0	0.000000	0.000000	0.0	...	0.0	0.0	0.000000	0.000000	0.0	0.0	0.000000	0.0	0.0
1	0.416578	0.0	0.000000	0.000000	0.0	0.000000	0.0	0.416578	0.000000	0.0	...	0.0	0.0	0.416578	0.000000	0.0	0.0	0.000000	0.0	0.0
2	0.000000	0.0	0.000000	0.000000	0.0	0.000000	0.0	0.000000	0.562609	0.0	...	0.0	0.0	0.000000	0.000000	0.0	0.0	0.000000	0.0	0.0
3	0.000000	0.0	0.000000	0.442107	0.0	0.492899	0.0	0.000000	0.000000	0.0	...	0.0	0.0	0.000000	0.564485	0.0	0.0	0.492899	0.0	0.0
4	0.000000	0.0	0.000000	0.000000	0.0	0.000000	0.0	0.000000	0.000000	0.0	...	0.0	0.0	0.000000	0.000000	0.0	0.0	0.000000	0.0	0.0

5 rows × 55 columns

Step 7: Dividing dataset into training and testing data

```
print("\n Train Test Split:\n")
xtrain,xtest,ytrain,ytest=train_test_split(data,y,test_size=0.3,random_state=2)
print('\n The total number of Training Data:',ytrain.shape)
print('\n The total number of Test Data:',ytest.shape)
```

```
 Train Test Split:


 The total number of Training Data: (11,)

 The total number of Test Data: (6,)
```

Step 8: Training Naïve Bayes classifier on training data

```
clf= MultinomialNB().fit(xtrain, ytrain)
predicted = clf.predict(xtest)

#printing accuracy, Confusion matrix, Precision and Recall
print("\n Accuracy of the classifier is:", metrics.accuracy_score(ytest,predicted))
print("\nConfision Matrix is:", metrics.confusion_matrix(ytest,predicted))
print("\nClassification Report:", metrics.classification_report(ytest,predicted))
print("\nThe value of Precision :", metrics.precision_score(ytest,predicted))
print("\nThe value of Recall:", metrics.recall_score(ytest,predicted))
```

```
 Accuracy of the classifier is: 0.6666666666666666

Confision Matrix is: [[3 0]
 [2 1]]

Classification Report:               precision    recall  f1-score   support

           0       0.60      1.00      0.75         3
           1       1.00      0.33      0.50         3

    accuracy                           0.67         6
   macro avg       0.80      0.67      0.62         6
weighted avg       0.80      0.67      0.62         6


The value of Precision : 1.0

The value of Recall: 0.3333333333333333
```

Description

- Naïve Bayes methods are a set of supervised learning algorithms
- Applications → Real time prediction, Multiclass prediction
- 3 types of Naïve Bayes → Gaussian, Multinomial, Bernoulli
- Tfidf Vectorizer → Term Frequency Inverse Document Frequency