

SWETHA JENIFER S_27-2-23

LAB6_NLP

In []: *#STEP :1*

In [1]: `import pandas as pd`

In [2]: `df = pd.read_csv("SMSSpamCollection.csv",encoding='latin-1')`
`df.head()`

Out[2]:

	label	text	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

In [3]: `df.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'],axis=1,inplace=True)`

In [4]: `df.head()`

Out[4]:

	label	text
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

In []: *#STEP 2:*

In [5]: *#count the sms messages*
`df['text'].value_counts().sum()`

Out[5]: 5572

In []: *#STEP 3:*

```
In [6]: #use groupby()
df.groupby(['label']).count()
```

```
Out[6]:
```

	text
label	
ham	4825
spam	747

```
In [ ]: #STEP 4:
```

```
In [7]: y = df['label']
```

```
In [8]: X = df['text']
```

```
In [9]: #split the dataset into training and test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=42)
```

```
In [ ]: #STEP 5:
```

```
In [10]: #function to remove all punctuation and stopwords
from nltk.corpus import stopwords
def process_text(msg):
    punctuations = '!()-[]:,;\"\\<>./?@#${}%^_~*&'
    nopunc = [char for char in msg if char not in punctuations]
    nopunc = ''.join(nopunc)
    return [word for word in nopunc.split()
            if word.lower() not in stopwords.words('english')]
```

```
In [11]: import nltk
nltk.download('stopwords')

[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\1mscdsa42\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
```

```
Out[11]: True
```

```
In [ ]: #STEP 6:
```

```
In [12]: #create TfidfVectorizer and perform vectorization
from sklearn.feature_extraction.text import TfidfVectorizer
df1 = TfidfVectorizer(use_idf=True, analyzer = process_text, ngram_range=(1,3), min_df=1)
```

```
Out[12]: TfidfVectorizer(analyzer=<function process_text at 0x000002D087C039D0>,
                        ngram_range=(1, 3), stop_words='english')
```

```
In [ ]: #STEP 7:
```

```
In [13]: a = df1.fit_transform(X_train)
```

```
In [14]: a1 = df1.transform(X_test)
```

```
In [15]: #create multinomialNB model
from sklearn.naive_bayes import MultinomialNB
clf = MultinomialNB()
clf.fit(a,y_train)
```

```
Out[15]: MultinomialNB()
```

```
In [ ]: #STEP 8:
```

```
In [16]: #predict labels on test set
y_pred = clf.predict(a1)
y_pred
```

```
Out[16]: array(['ham', 'ham', 'ham', ..., 'ham', 'ham', 'spam'], dtype='<U4')
```

```
In [ ]: #STEP 9:
```

```
In [17]: #find confusion_matrix
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,y_pred)
```

```
Out[17]: array([[965,  0],
                [ 39, 111]], dtype=int64)
```

```
In [18]: #find classification_report
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
ham	0.96	1.00	0.98	965
spam	1.00	0.74	0.85	150
accuracy			0.97	1115
macro avg	0.98	0.87	0.92	1115
weighted avg	0.97	0.97	0.96	1115

```
In [ ]: #STEP 10:
```

```
In [19]: #modify ngram_range=(1,2) and perform 7 to 9
from sklearn.feature_extraction.text import TfidfVectorizer
df2 = TfidfVectorizer(use_idf=True, analyzer = process_text, ngram_range=(1,2), m
df2
```

```
Out[19]: TfidfVectorizer(analyzer=<function process_text at 0x000002D087C039D0>,
ngram_range=(1, 2), stop_words='english')
```

```
In [20]: b = df2.fit_transform(X_train)
b1= df2.transform(X_test)
```

```
In [21]: #create multinomialNB model
from sklearn.naive_bayes import MultinomialNB
clf = MultinomialNB()
clf.fit(b,y_train)
```

```
Out[21]: MultinomialNB()
```

```
In [22]: #predict labels on the test set
y1_pred = clf.predict(b1)
y1_pred
```

```
Out[22]: array(['ham', 'ham', 'ham', ..., 'ham', 'ham', 'spam'], dtype='<U4')
```

```
In [23]: #print confusion matrix
confusion_matrix(y_test,y1_pred)
```

```
Out[23]: array([[965,  0],
[ 39, 111]], dtype=int64)
```

```
In [24]: #print classification_report
print(classification_report(y_test,y1_pred))
```

	precision	recall	f1-score	support
ham	0.96	1.00	0.98	965
spam	1.00	0.74	0.85	150
accuracy			0.97	1115
macro avg	0.98	0.87	0.92	1115
weighted avg	0.97	0.97	0.96	1115