Spam detection

September 25, 2021

[1]:
```
import numpy as np,pandas as pd
```

[3]:
```
import string
from nltk.corpus import stopwords
```

[7]:
```
data=pd.read_csv('G:/Simplilearn/Data Science with Python/Practice Project/
↪1574413540_lesson91/Lesson 9 -1/SpamCollection/SpamCollection',sep='\t'
                 ,names=['response','message'])
```

[8]:
```
data.head()
```

[8]:

	response	message
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…

[10]:
```
data.describe()
```

[10]:

	response	message
count	5572	5572
unique	2	5169
top	ham	Sorry, I'll call later
freq	4825	30

[12]:
```
data.groupby('response').describe
```

[12]:
```
<bound method GroupBy.describe of <pandas.core.groupby.generic.DataFrameGroupBy
object at 0x00000220CDBFF310>>
```

[13]:
```
data['length']=data['message'].apply(len)
```

[14]:
```
data.head()
```

[14]:

	response	message	length
0	ham	Go until jurong point, crazy.. Available only …	111

```
1      ham                      Ok lar… Joking wif u oni…      29
2     spam  Free entry in 2 a wkly comp to win FA Cup fina…     155
3      ham  U dun say so early hor… U c already then say…      49
4      ham  Nah I don't think he goes to usf, he lives aro…      61
```

[16]:
```
from string import punctuation
```

[17]:
```
stop_nltk = stopwords.words("english")
stop_punct = list(punctuation)
```

[18]:
```
stop_final = stop_nltk + stop_punct
```

[19]:
```
def del_stop(sent):
    return [term for term in sent if term not in stop_final]
```

[22]:
```
data['message'].head(5).apply(del_stop)
```

[22]:
```
0    [G,  , u, n, l,  , j, u, r, n, g,  , p, n,  , …
1    [O, k,  , l, r,  , J, k, n, g,  , w, f,  , u, …
2    [F, r, e, e,  , e, n, r,  , n,  , 2,  ,  , w, …
3    [U,  , u, n,  ,  ,  , e, r, l,  , h, r,  , U, …
4    [N, h,  , I,  , n,  , h, n, k,  , h, e,  , g, …
Name: message, dtype: object
```

[26]:
```
from sklearn.feature_extraction.text import CountVectorizer
```

[29]:
```
bag_of_words=CountVectorizer(analyzer=del_stop).fit(data['message'])
```

[30]:
```
print(len(bag_of_words.vocabulary_))
```

```
81
```

[31]:
```
message_bow=bag_of_words.transform(data['message'])
```

[38]:
```
from sklearn.feature_extraction.text import TfidfTransformer
```

[39]:
```
tfidf_transformer=TfidfTransformer().fit(message_bow)
```

[40]:
```
message_tfidf=tfidf_transformer.transform(message_bow)
```

[41]:
```
print(message_tfidf.shape)
```

```
(5572, 81)
```

[43]:
```
 from sklearn.naive_bayes import MultinomialNB
```

[45]:
```
spam_detect_model=MultinomialNB().fit(message_tfidf,data['response'])
```

[46]:
```
message=data['message'][4]
bow_for_message=bag_of_words.transform([message])
tfidf=tfidf_transformer.transform(bow_for_message)

print('predicted',spam_detect_model.predict(tfidf)[0])
print('expected',data.response[4])
```

```
predicted ham
expected ham
```

[ ]: