NLP_Spam Detection

October 19, 2020

```
[20]: #import required libraries
      import pandas as pd
      import string
      from nltk.corpus import stopwords
```

```
[6]: #Get the spam data collection
     df_spamData = pd.read_csv("SpamCollection", sep='\t',␣
      ↪names=['response','message'])
```

```
[9]: df_spamData.describe()
```

```
[9]:        response                 message
     count      5572                    5572
     unique        2                    5169
     top         ham  Sorry, I'll call later
     freq       4825                      30
```

```
[10]: df_spamData.head()
```

```
[10]:   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…
```

```
[11]: #view response
      df_spamData.groupby('response').describe()
```

```
[11]:          message                                                       \
                count unique                                             top
      response
      ham        4825   4516                           Sorry, I'll call later
      spam        747    653  Please call our customer service representativ…


               freq
```

```
response
ham        30
spam        4
```

[12]:
```
#Verify length of the messages and also add it as a new column
df_spamData['lenght'] = df_spamData['message'].apply(len)
```

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

[14]:
```
  response                                           message  lenght
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
```

[17]:
```
#define a function to get rid of stopwords present in the messages
def message_text_process(mess):
    # trim none-alpha
    no_punctuation = [char for char in mess if char not in string.punctuation]
    # now form sentence
    no_punctuation = ''.join(no_punctuation)
    # now remove stop (reserved) words
    return [word for word in no_punctuation.split() if word.lower() not in␣
 ↪stopwords.words('english')]
```

[21]:
```
df_spamData['message'].head(5).apply(message_text_process)
```

[21]:
```
0    [Go, jurong, point, crazy, Available, bugis, n…
1                        [Ok, lar, Joking, wif, u, oni]
2    [Free, entry, 2, wkly, comp, win, FA, Cup, fin…
3        [U, dun, say, early, hor, U, c, already, say]
4    [Nah, dont, think, goes, usf, lives, around, t…
Name: message, dtype: object
```

[41]:
```
#start text processing with vectorizer
from sklearn.feature_extraction.text import CountVectorizer
```

[42]:
```
#use bag of words by applying the function and fit the data into it
bag_of_words_transformer = CountVectorizer(analyzer=message_text_process).
 ↪fit(df_spamData['message'])
```

[40]:
```
#print length of bag of words stored in the vocabulary_ attribute
len(bag_of_words_transformer.vocabulary_)
```

[40]:
```
11425
```

[43]:

```
message_bagofwords = bag_of_words_transformer.transform(df_spamData['message'])
```

[44]:

```
#apply tfidf transformer and fit the bag of words into it (transformed version)
from sklearn.feature_extraction.text import TfidfTransformer
tfidf_transformer = TfidfTransformer().fit(message_bagofwords)
message_tfidf = tfidf_transformer.transform(message_bagofwords)
```

[45]:

```
#print shape of the tfidf
message_tfidf.shape
```

[45]:

```
(5572, 11425)
```

[51]:

```
#choose naive Bayes model to detect the spam and fit the tfidf data into it
from sklearn.naive_bayes import MultinomialNB
spam_detect_model = MultinomialNB().fit(message_tfidf,df_spamData['response'])
```

[57]:

```
#check model for the predicted and expected value say for message#2 and␣
 ↪message#5
message = df_spamData['message'][5]
bag_of_words_for_message = bag_of_words_transformer.transform([message])
tfidf = tfidf_transformer.transform(bag_of_words_for_message)
```

[58]:

```
print('predicted', spam_detect_model.predict(tfidf)[0])
print('expected',df_spamData.response[5])
```

```
predicted ham
expected spam
```

[59]:

```
df_spamData['message'][5]
```

[59]:

```
"FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like
some fun you up for it still? Tb ok! XxX std chgs to send, £1.50 to rcv"
```