In [21]:
```
import pandas as pd
import string
from  nltk.corpus import stopwords
import numpy as np
```

In [43]:
```
#Get the spam data collection
spam = pd.read_csv('SpamCollection',sep='\t',names=['response','message'])
```

In [44]:
```
spam.head()
```

Out[44]:

| | 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... |

In [45]:
```
spam.describe()
```

Out[45]:

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

In [47]:
```
#view response
spam[['response']].head()
```

Out[47]:

| | response |
|---|---|
| 0 | ham |
| 1 | ham |
| 2 | spam |
| 3 | ham |
| 4 | ham |

In [48]:
```
spam.groupby('response').describe()
```

Out[48]:

| | message | | | |
|---|---|---|---|---|
| | count | unique | top | freq |
| response | | | | |
| ham | 4825 | 4516 | Sorry, I'll call later | 30 |
| spam | 747 | 653 | Please call our customer service representativ... | 4 |

In [55]:
```
#Verify length of the messages and also add it as a new column
spam['length'] = spam['message'].apply(len)
```

In [56]:
```
spam.head()
```

Out[56]:

| | 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 |

In [57]:
```
#define a function to get rid of stopwords present in the messages
def text_message(mess):
    no_pun= [char for char in mess if char not in string.punctuation]
    no_pun= ''.join(no_pun)
    return [word for word in no_pun.split() if word.lower() not in stopwords.words('english')]
```

In [60]:
```
spam['message'].head(5).apply(text_message)
```

Out[60]:
```
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
```

In [61]:
```
#start text processing with vectorizer
from sklearn.feature_extraction.text import CountVectorizer
```

In [82]:
```
#use bag of words by applying the function and fit the data into it
bage_of_word_transformer = CountVectorizer(analyzer=text_message).fit(spam['message'])
```

In [83]:
```
#print length of bag of words stored in the vocabulary_ attribute
print(len(bage_of_word_transformer.vocabulary_))
```
```
11425
```

In [84]:
```
message_bagof_word = bage_of_word_transformer.transform(spam['message'])
```

In [85]:
```
#apply tfidf transformer and fit the bag of words into it (transformed version)
from sklearn.feature_extraction.text import TfidfTransformer
tfidfTransformer = TfidfTransformer().fit(message_bagof_word)
```

In [71]:
```
#print shape of the tfidf
tfidf_message= tfidfTransformer.transform(message_bagof_word)
print(tfidf_message.shape)
```
```
(5572, 11425)
```

In [79]:
```
#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(tfidf_message,spam['response'])
```

In [91]:
```
#check model for the predicted and expected value say for message#2 and message#5
message = spam['message'][4]
bage_of_word_for_messge = bage_of_word_transformer.transform([message])
tfidf = tfidfTransformer.transform(bage_of_word_for_messge)
```

In [93]:
```
print ('predicted',spam_detect_model.predict(tfidf)[0])
print('expected',spam.response[4])
```
```
predicted ham
expected ham
```

In [ ]:
```

```