In [15]:
```
### Text Preprocessing

import string
from nltk.corpus import stopwords

import nltk
nltk.download('stopwords')

def textPreprocessing(data):
    #Removal of Punctuations
    remove_pun = [ c for c in data if c not in string.punctuation ]
    sentences = ''.join(remove_pun)
    #Converting Sentences to Words
    words = sentences.split(" ")
    #Removal of Stopwords
    vocabulary = [ word for word in words if word not in stopwords.words('english') ]
    #Return Vocabulary
    return vocabulary
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\SW20407278\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

In [ ]:
```
```

In [16]:
```
import numpy as np
import pandas as pd
```

In [17]:
```
df = pd.read_csv('C:/Users/SW20407278/Desktop/Final AI/Hands-On/NLP_Spam_Classificatic
df.head()
```

Out[17]:

	label	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 [18]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 2 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   label    5572 non-null   object
 1   message  5572 non-null   object
dtypes: object(2)
memory usage: 87.2+ KB
```

In [19]:
```
## Creation of BOW (Bag of Words)
from sklearn.feature_extraction.text import CountVectorizer
```

```
wordVector = CountVectorizer(analyzer=textPreprocessing) #Text Preprocessing and BOW
finalWordVector = wordVector.fit(df['message'])
```

In [ ]:
```
## Vocabulary
finalWordVector.vocabulary_
```

In [20]:
```
### Creation of Bag of Words

bow = finalWordVector.transform(df['message'])
```

In [21]:
```
bow
```

Out[21]:
```
<5572x11619 sparse matrix of type '<class 'numpy.int64'>'
        with 57067 stored elements in Compressed Sparse Row format>
```

In [22]:
```
### Applyin TF-IDF on BOW

from sklearn.feature_extraction.text import TfidfTransformer

### Calculation of TF and IDF
tfidfObject = TfidfTransformer().fit(bow)
```

In [23]:
```
### Transformation of data
final_feature = tfidfObject.transform(bow)
```

In [24]:
```
final_feature
```

Out[24]:
```
<5572x11619 sparse matrix of type '<class 'numpy.float64'>'
        with 57067 stored elements in Compressed Sparse Row format>
```

In [25]:
```
### Model Building

from sklearn.naive_bayes import MultinomialNB
model = MultinomialNB()
model.fit(final_feature, df['label'])
```

Out[25]:
```
MultinomialNB()
```

In [26]:
```
model.score(final_feature,df['label'])
```

Out[26]:
```
0.9791816223977028
```

In [27]:
```
inputSMS = input("Enter SMS Content: ")
preprocessText = textPreprocessing(inputSMS)
vector = finalWordVector.transform(preprocessText)
finalFeature = tfidfObject.transform(vector)
pred = model.predict(finalFeature)[0]

print("Given SMS is ",pred)
```

```
Enter SMS Content: Lottery Win !!!
Given SMS is  ham
```

In [ ]:

In [ ]: