In [2]:
```
import pandas as pd
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from sklearn.feature_extraction.text import CountVectorizer,TfidfVectorizer
from sklearn.preprocessing import LabelEncoder

stopwords = [ "a", "about", "above", "after", "again", "against", "all", "a
m", "an", "and", "any", "are", "as", "at", "be", "because", "been", "before"
, "being", "below", "between", "both", "but", "by", "could", "did", "do", "d
oes", "doing", "down", "during", "each", "few", "for", "from", "further", "h
ad", "has", "have", "having", "he", "he'd", "he'll", "he's", "her", "here",
"here's", "hers", "herself", "him", "himself", "his", "how", "how's", "i",
"i'd", "i'll", "i'm", "i've", "if", "in", "into", "is", "it", "it's", "its",
"itself", "let's", "me", "more", "most", "my", "myself", "nor", "of", "on",
"once", "only", "or", "other", "ought", "our", "ours", "ourselves", "out",
"over", "own", "same", "she", "she'd", "she'll", "she's", "should", "so", "s
ome", "such", "than", "that", "that's", "the", "their", "theirs", "them", "t
hemselves", "then", "there", "there's", "these", "they", "they'd", "they'll"
, "they're", "they've", "this", "those", "through", "to", "too", "under", "u
ntil", "up", "very", "was", "we", "we'd", "we'll", "we're", "we've", "were",
"what", "what's", "when", "when's", "where", "where's", "which", "while", "w
ho", "who's", "whom", "why", "why's", "with", "would", "you", "you'd", "yo
u'll", "you're", "you've", "your", "yours", "yourself", "yourselves" ]
```

In [4]:
```
datasets = pd.read_csv('/home/nihar/Desktop/SEM 7/ML/Lab/Lab4/spam1.csv')
print("\nData :\n",datasets)
print("\nData statistics\n",datasets.info())
```

```
Data :
        v1                                                 v2
0     spam  Free entry in 2 a wkly comp to win FA Cup fina...
1     spam  FreeMsg Hey there darling it's been 3 week's n...
2     spam  WINNER!! As a valued network customer you have...
3     spam  Had your mobile 11 months or more? U R entitle...
4     spam  SIX chances to win CASH! From 100 to 20,000 po...
..     ...                                                ...
508   spam  This is the 2nd time we have tried 2 contact u...
509    ham              Will  b going to esplanade fr home?
510    ham  Pity, * was in mood for that. So...any other s...
511    ham  The guy did some bitching but I acted like i'd...
512    ham                         Rofl. Its true to its name

[513 rows x 2 columns]
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 513 entries, 0 to 512
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   v1      513 non-null    object
 1   v2      513 non-null    object
dtypes: object(2)
memory usage: 8.1+ KB

Data statistics
 None
```

In [5]:
```
from sklearn.model_selection import train_test_split
X,Y = datasets['v2'],datasets['v1']
le = LabelEncoder()
Y = le.fit_transform(Y)
x_train,x_test,y_train,y_test = train_test_split(X,Y,test_size=0.20,random_s
tate=129)
print(x_train.shape,x_test.shape)
print(y_train.shape,x_test.shape)
```

```
(410,) (103,)
(410,) (103,)
```

In [7]:
```
cv = CountVectorizer(lowercase=True,stop_words=stopwords,ngram_range=(1,2))
xtrain = cv.fit_transform(x_train).toarray()
xtest = cv.transform(x_test).toarray()
print(xtrain.shape,xtest.shape)
```

```
(410, 5345) (103, 5345)
```

In [8]:
```
from sklearn import metrics
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score

import numpy as np
from sklearn.naive_bayes import MultinomialNB,GaussianNB
mnb = MultinomialNB()
mnb.fit(xtrain,y_train)
ypred = mnb.predict(xtest)
print("accuracy:", metrics.accuracy_score(y_test,ypred))
print("classification report:\n", metrics.classification_report(y_test,ypred
))
```

```
accuracy: 0.970873786407767
classification report:
               precision    recall  f1-score   support

           0       0.97      0.98      0.98        65
           1       0.97      0.95      0.96        38

    accuracy                           0.97       103
   macro avg       0.97      0.97      0.97       103
weighted avg       0.97      0.97      0.97       103
```

In [11]:
```
cv = TfidfVectorizer(lowercase=True,stop_words=stopwords,ngram_range=(1,2))
xtrain = cv.fit_transform(x_train).toarray()
xtest = cv.transform(x_test).toarray()
print(xtrain.shape,xtest.shape)
```

```
(410, 5345) (103, 5345)

/home/nihar/.local/lib/python3.6/site-packages/sklearn/feature_extraction/te
xt.py:386: UserWarning: Your stop_words may be inconsistent with your prepro
cessing. Tokenizing the stop words generated tokens ['let', 'll', 're', 'v
e'] not in stop_words.
  'stop_words.' % sorted(inconsistent))
```

In [12]:

```
mnb = MultinomialNB()
mnb.fit(xtrain,y_train)
ypred = mnb.predict(xtest)
print("accuracy:", metrics.accuracy_score(y_test,ypred))
print("classification report:\n", metrics.classification_report(y_test,ypred
))
```

```
accuracy: 0.9029126213592233
classification report:
               precision    recall  f1-score   support

           0       0.87      1.00      0.93        65
           1       1.00      0.74      0.85        38

    accuracy                           0.90       103
   macro avg       0.93      0.87      0.89       103
weighted avg       0.92      0.90      0.90       103
```