importing libraries

In [26]:
```
import nltk
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings("ignore")
```

Data gathering

In [9]:
```
df=pd.read_table("SMSSpamCollection", header=None, encoding='utf-8')
df.head()
```

Out[9]:

	0	1
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 [10]:
```
df.info()
```

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

In [18]:
```
# check class distribution
classes = df[0]
df[0].value_counts()
```

Out[18]:
```
ham     4825
spam     747
Name: 0, dtype: int64
```

Pre-processing of data

Preprocessing the data is an essential step in natural language process. In the following cells, we will convert our class labels to binary values using the LabelEncoder from sklearn, replace email addresses, URLs, phone numbers, and other symbols by using regular expressions, remove stop words, and extract word stems.

In [21]:
```
# converting class label to  binary values,0=ham,1=spam
from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
Y=encoder.fit_transform(classes)
```

```
print(Y[:20])
```

```
[0 0 1 0 0 1 0 0 1 1 0 1 1 0 0 1 0 0 0 1]
```

In [23]:
```
# now store SMS message data
text_messages= df[1]

print(text_messages[:15])
```

```
0     Go until jurong point, crazy.. Available only ...
1                         Ok lar... Joking wif u oni...
2     Free entry in 2 a wkly comp to win FA Cup fina...
3     U dun say so early hor... U c already then say...
4     Nah I don't think he goes to usf, he lives aro...
5     FreeMsg Hey there darling it's been 3 week's n...
6     Even my brother is not like to speak with me. ...
7     As per your request 'Melle Melle (Oru Minnamin...
8     WINNER!! As a valued network customer you have...
9     Had your mobile 11 months or more? U R entitle...
10    I'm gonna be home soon and i don't want to tal...
11    SIX chances to win CASH! From 100 to 20,000 po...
12    URGENT! You have won a 1 week FREE membership ...
13    I've been searching for the right words to tha...
14                  I HAVE A DATE ON SUNDAY WITH WILL!!
Name: 1, dtype: object
```

In [ ]:
```
# Regular Expressions

we want to switch some things like if have email address in text messages
```

In [30]:
```
# replace email address with "email"
processed = text_messages.str.replace(r'^.+@[^\.].*\.[a-z]{2,}$',"emailaddress")

# replae URL with "webaddress"
processed = processed.str.replace(r'^http\://[a-zA-Z0-9\-\.]+\.[a-zA-Z]{2-3}(/\S*)?$',

# Replace money symbols with 'moneysymb' (£ can by typed with ALT key + 156)
processed = processed.str.replace(r'£|\$', 'moneysymb')

# Replace 10 digit phone numbers (formats include paranthesis, spaces, no spaces, dash
processed = processed.str.replace(r'^\(?[\d]{3}\)?[\s-]?[\d]{3}[\s-]?[\d]{4}$',
                                  'phonenumbr')

# Replace numbers with 'numbr'
processed = processed.str.replace(r'\d+(\.\d+)?', 'numbr')

# Remove punctuation
processed = processed.str.replace(r'[^\w\d\s]', ' ')

# Replace whitespace between terms with a single space
processed = processed.str.replace(r'\s+', ' ')

# Remove leading and trailing whitespace
processed = processed.str.replace(r'^\s+|\s+?$', '')
```

In [31]:
```
# change word with lower case
processed = processed.str.lower()
processed
```

Out[31]:
```
0       go until jurong point crazy available only in ...
1                           ok lar joking wif u oni
2       free entry in numbr a wkly comp to win fa cup ...
3             u dun say so early hor u c already then say
4       nah i don t think he goes to usf he lives arou...
                           ...
5567    this is the numbrnd time we have tried numbr c...
5568                 will ü b going to esplanade fr home
5569    pity was in mood for that so any other suggest...
5570    the guy did some bitching but i acted like i d...
5571                        rofl its true to its name
Name: 1, Length: 5572, dtype: object
```

In [32]:
```
# remove stopwords from the text
from nltk.corpus import stopwords
```

In [33]:
```
stopwords  = set(stopwords.words("english"))

processed = processed.apply(lambda x : " ".join(i for i in x.split() if i not in stopw
```

In [34]:
```
processed
```

Out[34]:
```
0       go jurong point crazy available bugis n great ...
1                           ok lar joking wif u oni
2       free entry numbr wkly comp win fa cup final tk...
3                   u dun say early hor u c already say
4                 nah think goes usf lives around though
                           ...
5567    numbrnd time tried numbr contact u u moneysymb...
5568                        ü b going esplanade fr home
5569                          pity mood suggestions
5570    guy bitching acted like interested buying some...
5571                                 rofl true name
Name: 1, Length: 5572, dtype: object
```

In [36]:
```
# lets use stemming

ps = nltk.PorterStemmer()

processed = processed.apply(lambda x : " ".join(ps.stem(i) for i in x.split()))
```

In [38]:
```
processed.head(20)
```

Out[38]:
```
0     go jurong point crazi avail bugi n great world...
1                                 ok lar joke wif u oni
2     free entri numbr wkli comp win fa cup final tk...
3                   u dun say earli hor u c alreadi say
4                  nah think goe usf live around though
5     freemsg hey darl numbr week word back like fun...
6         even brother like speak treat like aid patent
7     per request mell mell oru minnaminungint nurun...
8     winner valu network custom select receivea mon...
9     mobil numbr month u r entitl updat latest colo...
10    gonna home soon want talk stuff anymor tonight...
11    six chanc win cash numbr numbr numbr pound txt...
12    urgent numbr week free membership moneysymbnum...
13    search right word thank breather promis wont t...
14                                          date sunday
15    xxxmobilemovieclub use credit click wap link n...
16                                           oh k watch
17    eh u rememb numbr spell name ye v naughti make...
18                           fine way u feel way gota b
19    england v macedonia dont miss goal team news t...
Name: 1, dtype: object
```

Feature generating

Feature engineering is the process of using domain knowledge of the data to create features for machine learning algorithms. In this project, the words in each text message will be our features. For this purpose, it will be necessary to tokenize each word. We will use the 1500 most common words as features.

In [46]:
```
from nltk.tokenize import word_tokenize

# creating bag of words

all_words = []

for message in processed:
    words = word_tokenize(message)
    for w in words:
        all_words.append(w)
all_words= nltk.FreqDist(all_words)
```

In [49]:
```
# print total number of words and the 15 most common words

print("Number of Words",len(all_words))
print("Most common words",all_words.most_common(20))
```

```
Number of Words 6579
Most common words [('numbr', 2648), ('u', 1207), ('call', 674), ('go', 456), ('get',
451), ('ur', 391), ('gt', 318), ('lt', 316), ('come', 304), ('moneysymbnumbr', 303),
('ok', 293), ('free', 284), ('day', 276), ('know', 275), ('love', 266), ('like', 26
1), ('got', 252), ('time', 252), ('good', 248), ('want', 247)]
```

In [65]:
```
# lets use most common 1500 words as features
word_features = list(all_words.keys())[:1500]
```

In [66]:
```
len(word_features)
```

Out[66]:
```
1500
```

In [ ]:
```
word_features
```

In [78]:
```
# The find_features function will determine which of the 1500 word features are contai
def find_features(message):
    words = word_tokenize(message)
    features = {}
    for word in word_features:
        features[word] = (word in words)

    return features

# Lets see an example!
features = find_features(processed[0])
for key, value in features.items():
    if value == True:
        print (key)
```

```
go
jurong
point
crazi
avail
bugi
n
great
world
la
e
buffet
cine
got
amor
wat
```

In [79]:
```
processed[0]
```

Out[79]:
```
'go jurong point crazi avail bugi n great world la e buffet cine got amor wat'
```

In [82]:
```
# Now lets do it for all the messages
messages = zip(processed, Y)
```

In [84]:
```
# call find_features function for each SMS message
featuresets = [(find_features(text), label) for (text, label) in messages]
```

In [86]:
```
# split the featureset into trainin and testing datasets using sklearn

from sklearn import model_selection


training,testing = model_selection.train_test_split(featuresets,test_size=0.25)
```

In [87]:
```
print(len(training))
```

```
4179
```

In [88]:
```
print(len(testing))
```

```
1393
```

Model evaluation

Now that we have our dataset, we can start building algorithms! Let's start with a simple linear support vector classifier, then expand to other algorithms. We'll need to import each algorithm we plan on using from sklearn. We also need to import some performance metrics, such as accuracy_score and classification_report.

In [94]:
```
from nltk.classify.scikitlearn import SklearnClassifier
from sklearn.svm import SVC

model = SklearnClassifier(SVC(kernel="linear"))

#train the model
model.train(training)

# test on testing datasets
accuracy=nltk.classify.accuracy(model,testing)*100
print("SVC Accuracy",accuracy)
```

```
SVC Accuracy 97.84637473079684
```

In [104…
```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC

from sklearn.metrics import classification_report,accuracy_score,confusion_matrix
```

In [108…
```
# define models to train

names = ["K Nearest Neighbors", "Decision Tree", "Random Forest", "Logistic Regressior
         "Naive Bayes", "SVM Linear"]
classifiers = [KNeighborsClassifier(),
    DecisionTreeClassifier(),
    RandomForestClassifier(),
    LogisticRegression(),
    MultinomialNB(),
    SVC(kernel = 'linear')]

models=zip(names,classifiers)

for name,model in models:
    nltk_model = SklearnClassifier(model)
    nltk_model.train(training)

    accuracy = nltk.classify.accuracy(nltk_model,testing)*100
    print("{} Accuracy:{}".format(name,accuracy))
```

```
K Nearest Neighbors Accuracy:92.46231155778895
Decision Tree Accuracy:96.91313711414213
Random Forest Accuracy:97.63101220387652
Logistic Regression Accuracy:97.91816223977028
Naive Bayes Accuracy:98.20531227566404
SVM Linear Accuracy:97.84637473079684
```

In [110…
```
# make class label prediction for testing set
txt_features, labels = zip(*testing)

prediction = nltk_model.classify_many(txt_features)
```

In [111…

```
# print a confusion matrix and a classification report
print(classification_report(labels, prediction))

pd.DataFrame(
    confusion_matrix(labels, prediction),
    index = [['actual', 'actual'], ['ham', 'spam']],
    columns = [['predicted', 'predicted'], ['ham', 'spam']])
```

```
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      1190
           1       0.98      0.87      0.92       203

    accuracy                           0.98      1393
   macro avg       0.98      0.93      0.95      1393
weighted avg       0.98      0.98      0.98      1393
```

Out[111]:

		predicted	
		ham	**spam**
actual	**ham**	1186	4
	spam	26	177

In [ ]: