In [1]:
```
import nltk
```

In [2]:
```
dir(nltk)
```

Out[2]:
```
['AbstractLazySequence',
 'AffixTagger',
 'AlignedSent',
 'Alignment',
 'AnnotationTask',
 'ApplicationExpression',
 'Assignment',
 'BigramAssocMeasures',
 'BigramCollocationFinder',
 'BigramTagger',
 'BinaryMaxentFeatureEncoding',
 'BlanklineTokenizer',
 'BllipParser',
 'BottomUpChartParser',
 'BottomUpLeftCornerChartParser',
 'BottomUpProbabilisticChartParser',
 'Boxer',
 'BrillTagger',
 'BrillTaggerTrainer',
 'CFG',
 'CRFTagger',
 'CfgReadingCommand',
 'ChartParser',
 'ChunkParserI',
 'ChunkScore',
 'Cistem',
 'ClassifierBasedPOSTagger',
 'ClassifierBasedTagger',
 'ClassifierI',
 'ConcordanceIndex',
 'ConditionalExponentialClassifier',
 'ConditionalFreqDist',
```

```
    'ConditionalProbDist',
    'ConditionalProbDistI',
    'ConfusionMatrix',
    'ContextIndex',
    'ContextTagger',
    'ContingencyMeasures',
    'CoreNLPDependencyParser',
    'CoreNLPParser',
    'Counter',
    'CrossValidationProbDist',
    'DRS',
    'DecisionTreeClassifier',
    'DefaultTagger',
    'DependencyEvaluator',
    'DependencyGrammar',
    'DependencyGraph',
    'DependencyProduction',
    'DictionaryConditionalProbDist',
    'DictionaryProbDist',
    'DiscourseTester',
    'DrtExpression',
    'DrtGlueReadingCommand',
    'ELEProbDist',
    'EarleyChartParser',
    'Expression',
    'FStructure',
    'FeatDict',
    'FeatList',
    'FeatStruct',
    'FeatStructReader',
    'Feature',
    'FeatureBottomUpChartParser',
    'FeatureBottomUpLeftCornerChartParser',
    'FeatureChartParser',
    'FeatureEarleyChartParser',
    'FeatureIncrementalBottomUpChartParser',
    'FeatureIncrementalBottomUpLeftCornerChartParser',
    'FeatureIncrementalChartParser',
    'FeatureIncrementalTopDownChartParser',
```

```
    'FeatureTopDownChartParser',
    'FreqDist',
    'HTTPPasswordMgrWithDefaultRealm',
    'HeldoutProbDist',
    'HiddenMarkovModelTagger',
    'HiddenMarkovModelTrainer',
    'HunposTagger',
    'IBMModel',
    'IBMModel1',
    'IBMModel2',
    'IBMModel3',
    'IBMModel4',
    'IBMModel5',
    'ISRIStemmer',
    'ImmutableMultiParentedTree',
    'ImmutableParentedTree',
    'ImmutableProbabilisticMixIn',
    'ImmutableProbabilisticTree',
    'ImmutableTree',
    'IncrementalBottomUpChartParser',
    'IncrementalBottomUpLeftCornerChartParser',
    'IncrementalChartParser',
    'IncrementalLeftCornerChartParser',
    'IncrementalTopDownChartParser',
    'Index',
    'InsideChartParser',
    'JSONTaggedDecoder',
    'JSONTaggedEncoder',
    'KneserNeyProbDist',
    'LancasterStemmer',
    'LaplaceProbDist',
    'LazyConcatenation',
    'LazyEnumerate',
    'LazyIteratorList',
    'LazyMap',
    'LazySubsequence',
    'LazyZip',
    'LeftCornerChartParser',
    'LidstoneProbDist',
```

```
    'LineTokenizer',
    'LogicalExpressionException',
    'LongestChartParser',
    'MLEProbDist',
    'MWETokenizer',
    'Mace',
    'MaceCommand',
    'MaltParser',
    'MaxentClassifier',
    'Model',
    'MultiClassifierI',
    'MultiParentedTree',
    'MutableProbDist',
    'NaiveBayesClassifier',
    'NaiveBayesDependencyScorer',
    'NgramAssocMeasures',
    'NgramTagger',
    'NonprojectiveDependencyParser',
    'Nonterminal',
    'OrderedDict',
    'PCFG',
    'Paice',
    'ParallelProverBuilder',
    'ParallelProverBuilderCommand',
    'ParentedTree',
    'ParserI',
    'PerceptronTagger',
    'PhraseTable',
    'PorterStemmer',
    'PositiveNaiveBayesClassifier',
    'ProbDistI',
    'ProbabilisticDependencyGrammar',
    'ProbabilisticMixIn',
    'ProbabilisticNonprojectiveParser',
    'ProbabilisticProduction',
    'ProbabilisticProjectiveDependencyParser',
    'ProbabilisticTree',
    'Production',
    'ProjectiveDependencyParser',
```

```
'Prover9',
'Prover9Command',
'ProxyBasicAuthHandler',
'ProxyDigestAuthHandler',
'ProxyHandler',
'PunktSentenceTokenizer',
'QuadgramAssocMeasures',
'QuadgramCollocationFinder',
'RSLPStemmer',
'RTEFeatureExtractor',
'RUS_PICKLE',
'RandomChartParser',
'RangeFeature',
'ReadingCommand',
'RecursiveDescentParser',
'RegexpChunkParser',
'RegexpParser',
'RegexpStemmer',
'RegexpTagger',
'RegexpTokenizer',
'ReppTokenizer',
'ResolutionProver',
'ResolutionProverCommand',
'SExprTokenizer',
'SLASH',
'Senna',
'SennaChunkTagger',
'SennaNERTagger',
'SennaTagger',
'SequentialBackoffTagger',
'ShiftReduceParser',
'SimpleGoodTuringProbDist',
'SklearnClassifier',
'SlashFeature',
'SnowballStemmer',
'SpaceTokenizer',
'StackDecoder',
'StanfordNERTagger',
'StanfordPOSTagger',
```

```
'StanfordSegmenter',
'StanfordTagger',
'StemmerI',
'SteppingChartParser',
'SteppingRecursiveDescentParser',
'SteppingShiftReduceParser',
'SyllableTokenizer',
'TYPE',
'TabTokenizer',
'TableauProver',
'TableauProverCommand',
'TaggerI',
'TestGrammar',
'Text',
'TextCat',
'TextCollection',
'TextTilingTokenizer',
'TnT',
'TokenSearcher',
'ToktokTokenizer',
'TopDownChartParser',
'TransitionParser',
'Tree',
'TreebankWordTokenizer',
'Trie',
'TrigramAssocMeasures',
'TrigramCollocationFinder',
'TrigramTagger',
'TweetTokenizer',
'TypedMaxentFeatureEncoding',
'Undefined',
'UniformProbDist',
'UnigramTagger',
'UnsortedChartParser',
'Valuation',
'Variable',
'ViterbiParser',
'WekaClassifier',
'WhitespaceTokenizer',
```

```
'WittenBellProbDist',
'WordNetLemmatizer',
'WordPunctTokenizer',
'__author__',
'__author_email__',
'__builtins__',
'__cached__',
'__classifiers__',
'__copyright__',
'__doc__',
'__file__',
'__keywords__',
'__license__',
'__loader__',
'__longdescr__',
'__maintainer__',
'__maintainer_email__',
'__name__',
'__package__',
'__path__',
'__spec__',
'__url__',
'__version__',
'absolute_import',
'accuracy',
'add_logs',
'agreement',
'align',
'alignment_error_rate',
'aline',
'api',
'app',
'apply_features',
'approxrand',
'arity',
'association',
'bigrams',
'binary_distance',
'binary_search_file',
```

```
'binding_ops',
'bisect',
'blankline_tokenize',
'bleu',
'bleu_score',
'bllip',
'boolean_ops',
'boxer',
'bracket_parse',
'breadth_first',
'brill',
'brill_trainer',
'build_opener',
'call_megam',
'casual',
'casual_tokenize',
'ccg',
'chain',
'chart',
'chat',
'choose',
'chunk',
'cistem',
'class_types',
'classify',
'clause',
'clean_html',
'clean_url',
'cluster',
'collections',
'collocations',
'combinations',
'compat',
'config_java',
'config_megam',
'config_weka',
'conflicts',
'confusionmatrix',
'conllstr2tree',
```

```
    'conlltags2tree',
    'corenlp',
    'corpus',
    'crf',
    'custom_distance',
    'data',
    'decisiontree',
    'decorator',
    'decorators',
    'defaultdict',
    'demo',
    'dependencygraph',
    'deque',
    'discourse',
    'distance',
    'download',
    'download_gui',
    'download_shell',
    'downloader',
    'draw',
    'drt',
    'earleychart',
    'edit_distance',
    'edit_distance_align',
    'elementtree_indent',
    'entropy',
    'equality_preds',
    'evaluate',
    'evaluate_sents',
    'everygrams',
    'extract_rels',
    'extract_test_sentences',
    'f_measure',
    'featstruct',
    'featurechart',
    'filestring',
    'find',
    'flatten',
    'fractional_presence',
```

```
'getproxies',
'ghd',
'glue',
'grammar',
'guess_encoding',
'help',
'hmm',
'hunpos',
'ibm1',
'ibm2',
'ibm3',
'ibm4',
'ibm5',
'ibm_model',
'ieerstr2tree',
'improved_close_quote_regex',
'improved_open_quote_regex',
'improved_open_single_quote_regex',
'improved_punct_regex',
'in_idle',
'induce_pcfg',
'inference',
'infile',
'inspect',
'install_opener',
'internals',
'interpret_sents',
'interval_distance',
'invert_dict',
'invert_graph',
'is_rel',
'islice',
'isri',
'jaccard_distance',
'json_tags',
'jsontags',
'lancaster',
'lazyimport',
'lfg',
```

```
    'line_tokenize',
    'linearlogic',
    'lm',
    'load',
    'load_parser',
    'locale',
    'log_likelihood',
    'logic',
    'mace',
    'malt',
    'map_tag',
    'mapping',
    'masi_distance',
    'maxent',
    'megam',
    'memoize',
    'meteor',
    'meteor_score',
    'metrics',
    'misc',
    'mwe',
    'naivebayes',
    'ne_chunk',
    'ne_chunk_sents',
    'ngrams',
    'nonprojectivedependencyparser',
    'nonterminals',
    'numpy',
    'os',
    'pad_sequence',
    'paice',
    'parse',
    'parse_sents',
    'pchart',
    'perceptron',
    'pk',
    'porter',
    'pos_tag',
    'pos_tag_sents',
```

```
'positivenaivebayes',
'pprint',
'pr',
'precision',
'presence',
'print_function',
'print_string',
'probability',
'projectivedependencyparser',
'prover9',
'punkt',
'py25',
'py26',
'py27',
'pydoc',
'python_2_unicode_compatible',
'raise_unorderable_types',
'ranks_from_scores',
'ranks_from_sequence',
're',
're_show',
'read_grammar',
'read_logic',
'read_valuation',
'recall',
'recursivedescent',
'regexp',
'regexp_span_tokenize',
'regexp_tokenize',
'register_tag',
'relextract',
'repp',
'resolution',
'ribes',
'ribes_score',
'root_semrep',
'rslp',
'rte_classifier',
'rte_classify',
```

```
    'rte_features',
    'rtuple',
    'scikitlearn',
    'scores',
    'segmentation',
    'sem',
    'senna',
    'sent_tokenize',
    'sequential',
    'set2rel',
    'set_proxy',
    'sexpr',
    'sexpr_tokenize',
    'shiftreduce',
    'simple',
    'sinica_parse',
    'skipgrams',
    'skolemize',
    'slice_bounds',
    'snowball',
    'sonority_sequencing',
    'spearman',
    'spearman_correlation',
    'stack_decoder',
    'stanford',
    'stanford_segmenter',
    'stem',
    'str2tuple',
    'string_span_tokenize',
    'string_types',
    'subprocess',
    'subsumes',
    'sum_logs',
    'sys',
    'tableau',
    'tadm',
    'tag',
    'tagset_mapping',
    'tagstr2tree',
```

```
'tbl',
'text',
'text_type',
'textcat',
'texttiling',
'textwrap',
'tkinter',
'tnt',
'tokenize',
'tokenwrap',
'toktok',
'toolbox',
'total_ordering',
'transitionparser',
'transitive_closure',
'translate',
'tree',
'tree2conllstr',
'tree2conlltags',
'treebank',
'treetransforms',
'trigrams',
'tuple2str',
'types',
'unify',
'unique_list',
'untag',
'usage',
'util',
'version_file',
'version_info',
'viterbi',
'weka',
'windowdiff',
'word_tokenize',
'wordnet',
'wordpunct_tokenize',
'wsd']
```

Read a text data

In [3]:
```
import pandas as pd
import numpy as np
```

In [10]:
```
df=open("SMSSpamCollection.tsv").read(500)
```

In [11]:
```
print(df)
```

```
ham     I've been searching for the right words to thank you for this b
reather. I promise i wont take your help for granted and will fulfil my
promise. You have been wonderful and a blessing at all times.
spam    Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2
005. Text FA to 87121 to receive entry question(std txt rate)T&C's appl
y 08452810075over18's
ham     Nah I don't think he goes to usf, he lives around here though
ham     Even my brother is not like to speak with me. They treat me lik
e aid
```

In [6]:
```
df[0:500]
```

Out[6]:
```
"ham\tI've been searching for the right words to thank you for this bre
ather. I promise i wont take your help for granted and will fulfil my p
romise. You have been wonderful and a blessing at all times.\nspam\tFre
e entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text F
A to 87121 to receive entry question(std txt rate)T&C's apply 084528100
75over18's\nham\tNah I don't think he goes to usf, he lives around here
though\nham\tEven my brother is not like to speak with me. They treat m
e like aid"
```

In [7]:
```
parsedata=df.replace("\t","\n").split('\n')
```

In [8]:
```
parsedata[0:5]
```

Out[8]:
```
['ham',
 "I've been searching for the right words to thank you for this breathe
r. I promise i wont take your help for granted and will fulfil my promi
```

```
se. You have been wonderful and a blessing at all times.",
 'spam',
 "Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. T
ext FA to 87121 to receive entry question(std txt rate)T&C's apply 0845
2810075over18's",
 'ham']
```

Seperating all the labels and the text.

In [9]:
```
labellist=parsedata[0::2]
```

In [10]:
```
textlist=parsedata[1::2]
```

In [ ]:
```
```

In [11]:
```
print(labellist[0:5])
print(textlist[0:5])
```

```
['ham', 'spam', 'ham', 'ham', 'ham']
["I've been searching for the right words to thank you for this breathe
r. I promise i wont take your help for granted and will fulfil my promi
se. You have been wonderful and a blessing at all times.", "Free entry
in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 871
21 to receive entry question(std txt rate)T&C's apply 08452810075over1
8's", "Nah I don't think he goes to usf, he lives around here though",
'Even my brother is not like to speak with me. They treat me like aids
patent.', 'I HAVE A DATE ON SUNDAY WITH WILL!!']
```

Reading the datset

In [12]:
```
df1=pd.read_csv("SMSSpamCollection.tsv",sep="\t",header=None)
```

In [13]:
```
df1.head()
```

Out[13]:

	0	1
0	ham	I've been searching for the right words to tha...
1	spam	Free entry in 2 a wkly comp to win FA Cup fina...
2	ham	Nah I don't think he goes to usf, he lives aro...
3	ham	Even my brother is not like to speak with me. ...
4	ham	I HAVE A DATE ON SUNDAY WITH WILL!!

In [14]:
```
df1.columns
```

Out[14]:
```
Int64Index([0, 1], dtype='int64')
```

In [15]:
```
df1.columns=["label",'body_text']  #assigning columns names
```

In [16]:
```
df1.head()
```

Out[16]:

	label	body_text
0	ham	I've been searching for the right words to tha...
1	spam	Free entry in 2 a wkly comp to win FA Cup fina...
2	ham	Nah I don't think he goes to usf, he lives aro...
3	ham	Even my brother is not like to speak with me. ...
4	ham	I HAVE A DATE ON SUNDAY WITH WILL!!

In [17]:
```
 df1.shape  #checking the shape of the data
```

Out[17]:
```
(5568, 2)
```

In [18]:
```
df1.isnull().sum() #checking the number of null values
```

Out[18]:
```
label        0
body_text    0
dtype: int64
```

In [19]:
```
print(len(df1))     #checking the count of each label
print(len(df1[df1['label']=='spam']))
print(len(df1[df1['label']=='ham']))
```

```
5568
746
4822
```

Removing Punctuations

In [20]:
```
import string
string.punctuation
```

Out[20]:
```
'!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~'
```

In [21]:
```
def remove_punc(text):
    text_nopunc="".join([char for char in text if char not in string.pu
nctuation])
    return text_nopunc

df1['body_text_clean']=df1['body_text'].apply(lambda x:remove_punc(x))
df1.head()
```

Out[21]:

	label	body_text	body_text_clean
0	ham	I've been searching for the right words to tha...	Ive been searching for the right words to than...
1	spam	Free entry in 2 a wkly comp to win FA Cup fina...	Free entry in 2 a wkly comp to win FA Cup fina...
2	ham	Nah I don't think he goes to usf, he lives aro...	Nah I dont think he goes to usf he lives aroun...
3	ham	Even my brother is not like to speak with me. ...	Even my brother is not like to speak with me T...
4	ham	I HAVE A DATE ON SUNDAY WITH WILL!!	I HAVE A DATE ON SUNDAY WITH WILL

Tokenization

In [23]:
```
import re
```

In [24]:
```
def tokenize(text):
    tokens=re.split('\W+',text)
    return tokens

df1['body_text_tokenized']=df1['body_text_clean'].apply(lambda x:tokeni
ze(x.lower()))
df1.head()
```

Out[24]:

	label	body_text	body_text_clean	body_text_tokenized
0	ham	I've been searching for the right words to tha...	Ive been searching for the right words to than...	[ive, been, searching, for, the, right, words,...
1	spam	Free entry in 2 a wkly comp to win FA Cup fina...	Free entry in 2 a wkly comp to win FA Cup fina...	[free, entry, in, 2, a, wkly, comp, to, win, f...
2	ham	Nah I don't think he goes to usf, he lives aro...	Nah I dont think he goes to usf he lives aroun...	[nah, i, dont, think, he, goes, to, usf, he, l...
3	ham	Even my brother is not like to speak with me. ...	Even my brother is not like to speak with me T...	[even, my, brother, is, not, like, to, speak, ...
4	ham	I HAVE A DATE ON SUNDAY WITH WILL!!	I HAVE A DATE ON SUNDAY WITH WILL	[i, have, a, date, on, sunday, with, will]

Removing Stopwords

In [25]:
```
import nltk
nltk.download('stopwords')
```

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

Out[25]: True

In [26]:
```
import nltk
stopword=nltk.corpus.stopwords.words('english')
print(stopword)
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "y
ou're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'your
selves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers',
'herself', 'it', "it's", 'its', 'itself', 'they', 'them', 'their', 'the
irs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "th
at'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'bee
n', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doin
g', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'unti
l', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'betw
een', 'into', 'through', 'during', 'before', 'after', 'above', 'below',
'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under',
'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where',
'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'oth
er', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so',
'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', "don't",
'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ai
n', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn',
"doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'is
n', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn',
"needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't",
'weren', "weren't", 'won', "won't", 'wouldn', "wouldn't"]
```

In [27]:
```
def remove_stopwords(tokenized_list):
    text=[word for word in tokenized_list if word not in stopword]
    return text

df1['body_text_nostop']=df1['body_text_tokenized'].apply(lambda x:remov
e_stopwords(x))
df1.head()
```

Out[27]:

label	body_text	body_text_clean	body_text_tokenized	body_text_nostop

	label	body_text	body_text_clean	body_text_tokenized	body_text_nostop
0	ham	I've been searching for the right words to tha...	Ive been searching for the right words to than...	[ive, been, searching, for, the, right, words,...	[ive, searching, right, words, thank, breather...
1	spam	Free entry in 2 a wkly comp to win FA Cup fina...	Free entry in 2 a wkly comp to win FA Cup fina...	[free, entry, in, 2, a, wkly, comp, to, win, f...	[free, entry, 2, wkly, comp, win, fa, cup, fin...
2	ham	Nah I don't think he goes to usf, he lives aro...	Nah I dont think he goes to usf he lives aroun...	[nah, i, dont, think, he, goes, to, usf, he, l...	[nah, dont, think, goes, usf, lives, around, t...
3	ham	Even my brother is not like to speak with me. ...	Even my brother is not like to speak with me T...	[even, my, brother, is, not, like, to, speak, ...	[even, brother, like, speak, treat, like, aids...
4	ham	I HAVE A DATE ON SUNDAY WITH WILL!!	I HAVE A DATE ON SUNDAY WITH WILL	[i, have, a, date, on, sunday, with, will]	[date, sunday]

In [28]:
```
df2=df1.copy()
```

In [29]:
```
print(df2)
```

```
     label                                          body_text  \
0      ham  I've been searching for the right words to tha...
1     spam  Free entry in 2 a wkly comp to win FA Cup fina...
2      ham  Nah I don't think he goes to usf, he lives aro...
3      ham  Even my brother is not like to speak with me. ...
4      ham                I HAVE A DATE ON SUNDAY WITH WILL!!
...    ...                                                ...
5563  spam  This is the 2nd time we have tried 2 contact u...
5564   ham              Will ü b going to esplanade fr home?
5565   ham  Pity, * was in mood for that. So...any other s...
5566   ham  The guy did some bitching but I acted like i'd...
5567   ham                        Rofl. Its true to its name

                                     body_text_clean  \
0     Ive been searching for the right words to than...
1     Free entry in 2 a wkly comp to win FA Cup fina...
```

```
2     Nah I dont think he goes to usf he lives aroun...
3     Even my brother is not like to speak with me T...
4                     I HAVE A DATE ON SUNDAY WITH WILL
...                                                 ...
5563  This is the 2nd time we have tried 2 contact u...
5564                 Will ü b going to esplanade fr home
5565  Pity  was in mood for that Soany other suggest...
5566  The guy did some bitching but I acted like id ...
5567                           Rofl Its true to its name

                                      body_text_tokenized  \
0     [ive, been, searching, for, the, right, words,...
1     [free, entry, in, 2, a, wkly, comp, to, win, f...
2     [nah, i, dont, think, he, goes, to, usf, he, l...
3     [even, my, brother, is, not, like, to, speak, ...
4            [i, have, a, date, on, sunday, with, will]
...                                                 ...
5563  [this, is, the, 2nd, time, we, have, tried, 2,...
5564       [will, ü, b, going, to, esplanade, fr, home]
5565  [pity, was, in, mood, for, that, soany, other,...
5566  [the, guy, did, some, bitching, but, i, acted,...
5567                    [rofl, its, true, to, its, name]

                                        body_text_nostop
0     [ive, searching, right, words, thank, breather...
1     [free, entry, 2, wkly, comp, win, fa, cup, fin...
2     [nah, dont, think, goes, usf, lives, around, t...
3     [even, brother, like, speak, treat, like, aids...
4                                       [date, sunday]
...                                                 ...
5563  [2nd, time, tried, 2, contact, u, u, 750, poun...
5564                  [ü, b, going, esplanade, fr, home]
5565                    [pity, mood, soany, suggestions]
5566  [guy, bitching, acted, like, id, interested, b...
5567                                 [rofl, true, name]

[5568 rows x 5 columns]
```

In [ ]:

In [30]:
```
ps=nltk.PorterStemmer()  #using Porter Stemmer
```

In [31]:
```
def stemming(tokenized_text):
    text=[ps.stem(word) for word in tokenized_text]
    return text

df1['body_text_stemmed']=df1['body_text_nostop'].apply(lambda x: stemmi
ng(x))
```

In [32]:
```
df1.head()
```

Out[32]:

	label	body_text	body_text_clean	body_text_tokenized	body_text_nostop	body_text_stemmed
0	ham	I've been searching for the right words to tha...	Ive been searching for the right words to than...	[ive, been, searching, for, the, right, words,...	[ive, searching, right, words, thank, breather...	[ive, search, right, word, thank, breather, pr...
1	spam	Free entry in 2 a wkly comp to win FA Cup fina...	Free entry in 2 a wkly comp to win FA Cup fina...	[free, entry, in, 2, a, wkly, comp, to, win, f...	[free, entry, 2, wkly, comp, win, fa, cup, fin...	[free, entri, 2, wkli, comp, win, fa, cup, fin...
2	ham	Nah I don't think he goes to usf, he lives aro...	Nah I dont think he goes to usf he lives aroun...	[nah, i, dont, think, he, goes, to, usf, he, l...	[nah, dont, think, goes, usf, lives, around, t...	[nah, dont, think, goe, usf, live, around, tho...
3	ham	Even my brother is not like to speak with me. ...	Even my brother is not like to speak with me T...	[even, my, brother, is, not, like, to, speak, ...	[even, brother, like, speak, treat, like, aids...	[even, brother, like, speak, treat, like, aid,...
4	ham	I HAVE A DATE ON SUNDAY WITH WILL!!	I HAVE A DATE ON SUNDAY WITH WILL	[i, have, a, date, on, sunday, with, will]	[date, sunday]	[date, sunday]

Lemmatization

In [33]:
```
wn=nltk.WordNetLemmatizer()
nltk.download('wordnet')
```

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

Out[33]: True

In [34]:
```
def lemmatizing(tokenized_text):
    text=[wn.lemmatize(word) for word in tokenized_text]
    return text
df2['body_text_lemmatized'] = df2['body_text_nostop'].apply(lambda x:le
mmatizing(x))
```

In [35]:
```
df2.head()
```

Out[35]:

	label	body_text	body_text_clean	body_text_tokenized	body_text_nostop	body_text_lemmatize
0	ham	I've been searching for the right words to tha...	Ive been searching for the right words to than...	[ive, been, searching, for, the, right, words,...	[ive, searching, right, words, thank, breather...	[ive, searching, righ word, thank, breather,
1	spam	Free entry in 2 a wkly comp to win FA Cup fina...	Free entry in 2 a wkly comp to win FA Cup fina...	[free, entry, in, 2, a, wkly, comp, to, win, f...	[free, entry, 2, wkly, comp, win, fa, cup, fin...	[free, entry, 2, wkl comp, win, fa, cup, fin

	label	body_text	body_text_clean	body_text_tokenized	body_text_nostop	body_text_lemmatize
2	ham	Nah I don't think he goes to usf, he lives aro...	Nah I dont think he goes to usf he lives aroun...	[nah, i, dont, think, he, goes, to, usf, he, l...	[nah, dont, think, goes, usf, lives, around, t...	[nah, dont, think, g usf, life, around, thoug
3	ham	Even my brother is not like to speak with me. ...	Even my brother is not like to speak with me T...	[even, my, brother, is, not, like, to, speak, ...	[even, brother, like, speak, treat, like, aids...	[even, brother, lik speak, treat, like, aid,
4	ham	I HAVE A DATE ON SUNDAY WITH WILL!!	I HAVE A DATE ON SUNDAY WITH WILL	[i, have, a, date, on, sunday, with, will]	[date, sunday]	[date, sunda

In [36]:
```
from sklearn.feature_extraction.text import CountVectorizer

cv=CountVectorizer()
x_counts=cv.fit_transform(df2['body_text'])
print(x_counts.shape)
print(x_counts)
```

```
(5568, 8710)
  (0, 8169)     1
  (0, 1474)     2
  (0, 6723)     1
  (0, 3321)     3
  (0, 7671)     1
  (0, 6513)     1
  (0, 8537)     1
  (0, 7803)     1
  (0, 7658)     1
  (0, 8665)     2
  (0, 7715)     1
  (0, 1705)     1
  (0, 6143)     2
```

```
(0, 8527)     1
(0, 7534)     1
(0, 8671)     1
(0, 3840)     1
(0, 3642)     1
(0, 1096)     2
(0, 8457)     1
(0, 3432)     1
(0, 5251)     1
(0, 3792)     1
(0, 8524)     1
(0, 1581)     1
:     :
(5566, 3371)  1
(5566, 4112)  1
(5566, 3803)  1
(5566, 4639)  1
(5566, 1450)  1
(5566, 8368)  1
(5566, 5364)  1
(5566, 2604)  1
(5566, 4243)  1
(5566, 8117)  1
(5566, 1793)  1
(5566, 7096)  1
(5566, 2903)  1
(5566, 3487)  1
(5566, 7086)  1
(5566, 1801)  1
(5566, 3707)  1
(5566, 4186)  1
(5566, 914)   1
(5566, 1560)  1
(5567, 7803)  1
(5567, 5273)  1
(5567, 4251)  2
(5567, 7935)  1
(5567, 6545)  1
```

In [88]:
```
##data=pd.DataFrame(x_counts.toarray())
##print(data)
```

```
      0     1     2     3     4     5     6     7     8     9     ...
8700  \
0        0     0     0     0     0     0     0     0     0     0  ...
   0
1        0     0     0     0     0     0     0     0     0     0  ...
   0
2        0     0     0     0     0     0     0     0     0     0  ...
   0
3        0     0     0     0     0     0     0     0     0     0  ...
   0
4        0     0     0     0     0     0     0     0     0     0  ...
   0
...    ...   ...   ...   ...   ...   ...   ...   ...   ...   ...  ...
...
5563     0     0     0     0     0     0     0     0     0     0  ...
   0
5564     0     0     0     0     0     0     0     0     0     0  ...
   0
5565     0     0     0     0     0     0     0     0     0     0  ...
   0
5566     0     0     0     0     0     0     0     0     0     0  ...
   0
5567     0     0     0     0     0     0     0     0     0     0  ...
   0


      8701  8702  8703  8704  8705  8706  8707  8708  8709
0        0     0     0     0     0     0     0     0     0
1        0     0     0     0     0     0     0     0     0
2        0     0     0     0     0     0     0     0     0
3        0     0     0     0     0     0     0     0     0
4        0     0     0     0     0     0     0     0     0
...    ...   ...   ...   ...   ...   ...   ...   ...   ...
5563     0     0     0     0     0     0     0     0     0
5564     0     0     0     0     0     0     0     0     0
5565     0     0     0     0     0     0     0     0     0
5566     0     0     0     0     0     0     0     0     0
5567     0     0     0     0     0     0     0     0     0
```

```
[5568 rows x 8710 columns]
```

Seperating the dependent and independent variable

In [38]:
```
x=x_counts.toarray()
y=df2.iloc[:,0].values
print(y)
```

```
['ham' 'spam' 'ham' ... 'ham' 'ham' 'ham']
```

Encoding the dependent variable

In [41]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
y=le.fit_transform(y)
print(y)
```

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

Splitting the Data into train and test

In [42]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y, test_size=0.20 , ra
ndom_state = 0)
```

Training the model

In [43]:
```
from sklearn.naive_bayes import GaussianNB
classifier=GaussianNB()
classifier.fit(x_train,y_train)
```

Out[43]:
```
GaussianNB(priors=None, var_smoothing=1e-09)
```

Predicting the value of test set

In [44]:
```
y_pred=classifier.predict(x_test)
```

Confusion Matrix

In [47]:
```
from sklearn.metrics import confusion_matrix,accuracy_score
cm=confusion_matrix(y_test,y_pred)
```

In [48]:
```
print(cm)
print(accuracy_score(y_pred,y_test))
```

```
[[867  88]
 [ 11 148]]
0.9111310592459605
```

In [ ]:

In [ ]:

In [ ]: