Importing Libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

Loading the dataset

In [2]:
```
df=pd.read_csv(r"C:\Users\Shubham\Desktop\Data Science\Data Science Class\Stats and
df.head()
```

Out[2]:

	Unnamed: 0	label	text	label_num
0	605	ham	Subject: enron methanol ; meter # : 988291\r\n...	0
1	2349	ham	Subject: hpl nom for january 9 , 2001\r\n(see...	0
2	3624	ham	Subject: neon retreat\r\nho ho ho , we ' re ar...	0
3	4685	spam	Subject: photoshop , windows , office . cheap ...	1
4	2030	ham	Subject: re : indian springs\r\nthis deal is t...	0

In [3]:
```
df=df.iloc[:,1:3]
```

In [4]:
```
df.head()
```

Out[4]:

	label	text
0	ham	Subject: enron methanol ; meter # : 988291\r\n...
1	ham	Subject: hpl nom for january 9 , 2001\r\n(see...
2	ham	Subject: neon retreat\r\nho ho ho , we ' re ar...
3	spam	Subject: photoshop , windows , office . cheap ...
4	ham	Subject: re : indian springs\r\nthis deal is t...

Check for the null values

In [5]:
```
df.isna().sum()
```

Out[5]:
```
label    0
text     0
dtype: int64
```

In [6]:
```
df.label.value_counts()
```

Out[6]:
```
ham     3672
spam    1499
Name: label, dtype: int64
```

Balancing the Data

In [7]:
```
ham = df[df['label']=='ham']
spam = df[df['label']=='spam']
```

In [8]:
```
spam = spam.sample(ham.shape[0], replace=True)
```

In [9]:
```
print(ham.shape, spam.shape)
```

```
(3672, 2) (3672, 2)
```

In [10]:
```
df = ham.append(spam, ignore_index=True)
df.shape
```

Out[10]:
```
(7344, 2)
```

In [11]:
```
df.label.value_counts()
```

Out[11]:
```
ham     3672
spam    3672
Name: label, dtype: int64
```

Importing The Nural language toolkit

In [12]:
```
import re
import nltk
nltk.download('stopwords')
```

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

Out[12]:
```
True
```

preprocess a corpus of text data using NLTK's tools for natural language processing.

In [16]:
```
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
ps = PorterStemmer()
corpus = []

for i in range(0, len(df)):
    review = re.sub('[^a-zA-Z]',' ', df['text'][i])
    review = review.lower()
    review = review.split()
    review = [ps.stem(word) for word in review if not word in stopwords.words('eng
    review = ' '.join(review)
    corpus.append(review)
```

Bag of words

In [17]:
```
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer()
x = cv.fit_transform(corpus).toarray()
```

In [18]:
```
x.shape
```

Out[18]:
```
(7344, 36280)
```

In [19]:
```
x
```

Out[19]:
```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]], dtype=int64)
```

In [21]:
```
df.head()
```

Out[21]:

	label	text
0	ham	Subject: enron methanol ; meter # : 988291\r\n...
1	ham	Subject: hpl nom for january 9 , 2001\r\n(see...
2	ham	Subject: neon retreat\r\nho ho ho , we ' re ar...
3	ham	Subject: re : indian springs\r\nthis deal is t...
4	ham	Subject: ehronline web address change\r\nthis ...

Encoding Concept

In [22]:
```
df['label'] = df['label'].astype('category')
df['label'] = df['label'].cat.codes
```

In [23]:
```
df.shape
```

Out[23]:
```
(7344, 2)
```

split the data into training and test

In [25]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, df['label'], test_size=0.25
```

Building Niave Bayes Theorem

In [26]:
```
from sklearn.naive_bayes import MultinomialNB
nbmodel = MultinomialNB().fit(x_train, y_train)
```

Predict the data

In [27]:
```
y_pred_train = nbmodel.predict(x_train)
y_pred_test = nbmodel.predict(x_test)
```

In [28]:
```
y_pred_test
```

Out[28]:
```
array([0, 0, 0, ..., 1, 1, 1], dtype=int8)
```

Evaluate the model

In [29]:
```
from sklearn.metrics import classification_report, accuracy_score, confusion_matri
```

In [30]:
```
print(confusion_matrix(y_train, y_pred_train))
print()
print(confusion_matrix(y_test, y_pred_test))
```

```
[[2761   30]
 [  69 2648]]

[[864  17]
 [ 25 930]]
```

In [31]:
```
print(classification_report(y_train, y_pred_train))
print()
print(classification_report(y_test, y_pred_test))
```

```
              precision    recall  f1-score   support

           0       0.98      0.99      0.98      2791
           1       0.99      0.97      0.98      2717

    accuracy                           0.98      5508
   macro avg       0.98      0.98      0.98      5508
weighted avg       0.98      0.98      0.98      5508


              precision    recall  f1-score   support

           0       0.97      0.98      0.98       881
           1       0.98      0.97      0.98       955

    accuracy                           0.98      1836
   macro avg       0.98      0.98      0.98      1836
weighted avg       0.98      0.98      0.98      1836
```

In [32]:
```
print(accuracy_score(y_train, y_pred_train))
print()
print(accuracy_score(y_test, y_pred_test))
```

```
0.9820261437908496

0.9771241830065359
```

In [ ]: