In [3]:
```
import numpy as np
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
from sklearn.naive_bayes import BernoulliNB
from sklearn.feature_extraction.text import CountVectorizer
```

In [9]:
```
df=pd.read_csv("C:/Users/HP/Downloads/spam.csv",encoding="latin-1")
```

In [10]:
```
#visualizing dataset
df.head(n=10)
```

Out[10]:

	class	message	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN
5	spam	FreeMsg Hey there darling it's been 3 week's n...	NaN	NaN	NaN
6	ham	Even my brother is not like to speak with me. ...	NaN	NaN	NaN
7	ham	As per your request 'Melle Melle (Oru Minnamin...	NaN	NaN	NaN
8	spam	WINNER!! As a valued network customer you have...	NaN	NaN	NaN
9	spam	Had your mobile 11 months or more? U R entitle...	NaN	NaN	NaN

In [11]:
```
df.shape
```

Out[11]:
```
(5572, 5)
```

In [12]:
```
#to check whether target attribute is binary or not
np.unique(df['class'])
```

Out[12]:
```
array(['ham', 'spam'], dtype=object)
```

In [13]:
```
np.unique(df['message'])
```

Out[13]:
```
array([' &lt;#&gt;  in mca. But not conform.',
       ' &lt;#&gt;  mins but i had to stop somewhere first.',
       ' &lt;DECIMAL&gt; m but its not a common car here so its better to buy from china
or asia. Or if i find it less expensive. I.ll holla',
       ..., 'ÌÏ thk of wat to eat tonight.', 'ÌÏ v ma fan...',
       'ÌÏ wait 4 me in sch i finish ard 5..'], dtype=object)
```

In [14]:

```
#creating sparse matrix

x=df["message"].values
y=df["class"].values

#create count vectorizer object
cv=CountVectorizer()

x=cv.fit_transform(x)
v=x.toarray()

print(v)
```

```
[[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]]
```

In [15]:

```
first_col=df.pop('message')
df.insert(0,'message',first_col)
df
```

Out[15]:

	message	class	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	Go until jurong point, crazy.. Available only ...	ham	NaN	NaN	NaN
1	Ok lar... Joking wif u oni...	ham	NaN	NaN	NaN
2	Free entry in 2 a wkly comp to win FA Cup fina...	spam	NaN	NaN	NaN
3	U dun say so early hor... U c already then say...	ham	NaN	NaN	NaN
4	Nah I don't think he goes to usf, he lives aro...	ham	NaN	NaN	NaN
...	...	...	...	...	...
5567	This is the 2nd time we have tried 2 contact u...	spam	NaN	NaN	NaN
5568	Will Ì_ b going to esplanade fr home?	ham	NaN	NaN	NaN
5569	Pity, * was in mood for that. So...any other s...	ham	NaN	NaN	NaN
5570	The guy did some bitching but I acted like i'd...	ham	NaN	NaN	NaN
5571	Rofl. Its true to its name	ham	NaN	NaN	NaN

5572 rows × 5 columns

In [30]:

```
#splitting train + 3:1

train_x=x[:4180]
train_y=y[:4180]

test_x=x[4180:]
test_y=y[4180:]
```

In [31]:
```
bnb=BernoulliNB(binarize=0.0)
model=bnb.fit(train_x,train_y)

y_pred_train=bnb.predict(train_x)
y_pred_test=bnb.predict(test_x)
```

In [32]:
```
#training score
print(bnb.score(train_x,train_y)*100)

#testing score
print(bnb.score(test_x,test_y)*100)
```

```
98.70813397129187
98.20402298850574
```

In [33]:
```
from sklearn.metrics import classification_report
print(classification_report(train_y,y_pred_train))
```

```
              precision    recall  f1-score   support

         ham       0.99      1.00      0.99      3615
        spam       0.99      0.91      0.95       565

    accuracy                           0.99      4180
   macro avg       0.99      0.95      0.97      4180
weighted avg       0.99      0.99      0.99      4180
```

In [34]:
```
from sklearn.metrics import classification_report
print(classification_report(test_y,y_pred_test))
```

```
              precision    recall  f1-score   support

         ham       0.98      1.00      0.99      1210
        spam       0.99      0.87      0.93       182

    accuracy                           0.98      1392
   macro avg       0.99      0.93      0.96      1392
weighted avg       0.98      0.98      0.98      1392
```