Import Libraries & Load Data

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
df = pd.read_csv('spambase.csv', header=None)
```

In [3]:
```
df.head()
```

Out[3]:

	0	1	2	3	4	5	6	7	8	9	...	48	49	50	51	52
0	0.00	0.64	0.64	0.0	0.32	0.00	0.00	0.00	0.00	0.00	...	0.00	0.000	0.0	0.778	0.000
1	0.21	0.28	0.50	0.0	0.14	0.28	0.21	0.07	0.00	0.94	...	0.00	0.132	0.0	0.372	0.180
2	0.06	0.00	0.71	0.0	1.23	0.19	0.19	0.12	0.64	0.25	...	0.01	0.143	0.0	0.276	0.184
3	0.00	0.00	0.00	0.0	0.63	0.00	0.31	0.63	0.31	0.63	...	0.00	0.137	0.0	0.137	0.000
4	0.00	0.00	0.00	0.0	0.63	0.00	0.31	0.63	0.31	0.63	...	0.00	0.135	0.0	0.135	0.000

5 rows × 58 columns

Data Exploration

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4601 entries, 0 to 4600
Data columns (total 58 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   0       4601 non-null   float64
 1   1       4601 non-null   float64
 2   2       4601 non-null   float64
 3   3       4601 non-null   float64
 4   4       4601 non-null   float64
 5   5       4601 non-null   float64
 6   6       4601 non-null   float64
 7   7       4601 non-null   float64
 8   8       4601 non-null   float64
 9   9       4601 non-null   float64
 10  10      4601 non-null   float64
 11  11      4601 non-null   float64
 12  12      4601 non-null   float64
 13  13      4601 non-null   float64
 14  14      4601 non-null   float64
 15  15      4601 non-null   float64
 16  16      4601 non-null   float64
 17  17      4601 non-null   float64
 18  18      4601 non-null   float64
 19  19      4601 non-null   float64
 20  20      4601 non-null   float64
 21  21      4601 non-null   float64
 22  22      4601 non-null   float64
 23  23      4601 non-null   float64
 24  24      4601 non-null   float64
 25  25      4601 non-null   float64
 26  26      4601 non-null   float64
 27  27      4601 non-null   float64
 28  28      4601 non-null   float64
 29  29      4601 non-null   float64
 30  30      4601 non-null   float64
 31  31      4601 non-null   float64
 32  32      4601 non-null   float64
 33  33      4601 non-null   float64
 34  34      4601 non-null   float64
 35  35      4601 non-null   float64
 36  36      4601 non-null   float64
 37  37      4601 non-null   float64
 38  38      4601 non-null   float64
 39  39      4601 non-null   float64
 40  40      4601 non-null   float64
 41  41      4601 non-null   float64
 42  42      4601 non-null   float64
 43  43      4601 non-null   float64
 44  44      4601 non-null   float64
 45  45      4601 non-null   float64
 46  46      4601 non-null   float64
 47  47      4601 non-null   float64
 48  48      4601 non-null   float64
```

```
 49  49      4601 non-null   float64
 50  50      4601 non-null   float64
 51  51      4601 non-null   float64
 52  52      4601 non-null   float64
 53  53      4601 non-null   float64
 54  54      4601 non-null   float64
 55  55      4601 non-null   int64
 56  56      4601 non-null   int64
 57  57      4601 non-null   int64
dtypes: float64(55), int64(3)
memory usage: 2.0 MB
```

In [5]:
```
df.describe()
```

Out[5]:

	0	1	2	3	4	5	
count	4601.000000	4601.000000	4601.000000	4601.000000	4601.000000	4601.000000	4601.00
mean	0.104553	0.213015	0.280656	0.065425	0.312223	0.095901	0.11
std	0.305358	1.290575	0.504143	1.395151	0.672513	0.273824	0.39
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00
25%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00
50%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00
75%	0.000000	0.000000	0.420000	0.000000	0.380000	0.000000	0.00
max	4.540000	14.280000	5.100000	42.810000	10.000000	5.880000	7.27

8 rows × 58 columns

Feature Engineering

In [6]:
```
X = df.iloc[:,:57]
```

In [7]:
```
y = pd.DataFrame(df.iloc[:,-1])
```

In [8]:
```
X.shape, y.shape
```

Out[8]: ((4601, 57), (4601, 1))

In [9]:
```
from sklearn.model_selection import train_test_split
```

In [10]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, str
```

In [11]:
```
from sklearn.preprocessing import StandardScaler
```

In [12]:
```
scaler = StandardScaler()
```

In [13]:
```
X_train_scaled = scaler.fit_transform(X_train)
```

In [14]:
```
X_test_scaled = scaler.transform(X_test)
```

Model & Evaluation

In [15]:
```
from sklearn.naive_bayes import BernoulliNB, GaussianNB
from sklearn.metrics import accuracy_score, classification_report, confusion_
```

In [16]:
```
bayes_bern = BernoulliNB(binarize=0.01)
bayes_bern.fit(X_train_scaled, y_train)
y_pred = bayes_bern.predict(X_test_scaled)
```

```
C:\Users\15516\anaconda3\lib\site-packages\sklearn\utils\validation.py:760:
DataConversionWarning: A column-vector y was passed when a 1d array was exp
ected. Please change the shape of y to (n_samples, ), for example using rav
el().
  y = column_or_1d(y, warn=True)
```

In [17]:
```
accuracy_score(y_test, y_pred), confusion_matrix(y_test, y_pred)
```

Out[17]:
```
(0.9196234612599565,
 array([[803,  34],
        [ 77, 467]], dtype=int64))
```

In [18]:
```
bayes_gaus = GaussianNB()
bayes_gaus.fit(X_train_scaled, y_train)
y_pred = bayes_gaus.predict(X_test_scaled)
```

```
C:\Users\15516\anaconda3\lib\site-packages\sklearn\naive_bayes.py:206: Data
ConversionWarning: A column-vector y was passed when a 1d array was expecte
d. Please change the shape of y to (n_samples, ), for example using ravel
().
  y = column_or_1d(y, warn=True)
```

In [19]:
```
accuracy_score(y_test, y_pred), confusion_matrix(y_test, y_pred)
```

Out[19]:
```
(0.8088341781317886,
 array([[605, 232],
        [ 32, 512]], dtype=int64))
```

In [ ]:
```

```