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

In [3]:
```
df = pd.read_csv('emails.csv')
```

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

Out[4]:

	Email No.	the	to	ect	and	for	of	a	you	hou	...	connevey	jay	valued	lay	infrastructure	milit
0	Email 1	0	0	1	0	0	0	2	0	0	...	0	0	0	0	0	
1	Email 2	8	13	24	6	6	2	102	1	27	...	0	0	0	0	0	
2	Email 3	0	0	1	0	0	0	8	0	0	...	0	0	0	0	0	
3	Email 4	0	5	22	0	5	1	51	2	10	...	0	0	0	0	0	
4	Email 5	7	6	17	1	5	2	57	0	9	...	0	0	0	0	0	

5 rows × 3002 columns

In [5]:
```
df.columns
```

Out[5]:
```
Index(['Email No.', 'the', 'to', 'ect', 'and', 'for', 'of', 'a', 'you', 'hou',
       ...
       'connevey', 'jay', 'valued', 'lay', 'infrastructure', 'military',
       'allowing', 'ff', 'dry', 'Prediction'],
      dtype='object', length=3002)
```

In [6]:
```
df.shape
```

Out[6]:
```
(5172, 3002)
```

In [7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5172 entries, 0 to 5171
Columns: 3002 entries, Email No. to Prediction
dtypes: int64(3001), object(1)
memory usage: 118.5+ MB
```

In [8]:
```
df.isnull().sum()
```

Out[8]:
```
Email No.     0
the           0
to            0
ect           0
and           0
              ..
military      0
allowing      0
ff            0
dry           0
Prediction    0
Length: 3002, dtype: int64
```

In [9]:
```
df.isnull().sum().sum()
```

Out[9]:
```
0
```

In [10]:
```
df.drop(['Email No.'], axis=1, inplace=True)
```

In [11]:
```
x = df.drop(['Prediction'], axis=1)
y = df['Prediction']
```

In [13]:
```
from sklearn.preprocessing import scale
x = scale(x)
```

In [14]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state
=42)
```

In [17]:
```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=7)
```

In [18]:
```
knn.fit(x_train, y_train)
```

Out[18]:
```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=7, p=2,
                     weights='uniform')
```

In [19]:
```
y_pred = knn.predict(x_test)
```

In [20]:
```
y_pred
```

Out[20]:
```
array([0, 0, 1, ..., 1, 1, 1], dtype=int64)
```

In [21]:
```
from sklearn import metrics
accuracy = metrics.accuracy_score(y_test, y_pred)
accuracy
```

Out[21]:
```
0.8009020618556701
```

In [22]:
```
cm = metrics.confusion_matrix(y_test, y_pred)
cm
```

Out[22]:
```
array([[804, 293],
       [ 16, 439]], dtype=int64)
```

In [25]:
```
from sklearn.svm import SVC
model = SVC(C=1)
```

In [26]:
```
model.fit(x_train, y_train)
```

Out[26]:
```
SVC(C=1, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

In [27]:
```
y_pred_sv = model.predict(x_test)
```

In [28]:
```
accuracy_sv = metrics.accuracy_score(y_test, y_pred)
accuracy_sv
```

Out[28]:
```
0.8009020618556701
```

In [29]:
```
cm_sv = metrics.confusion_matrix(y_test, y_pred)
cm_sv
```

Out[29]:
```
array([[804, 293],
       [ 16, 439]], dtype=int64)
```