Machine Learning Laboratory – Assignment 2

- **NAME :- ANURAG AVINASH SHEVALE**
- **CLASS :- BE COMP I**
- **ROLL NO :- 20**

In [1]:
```
#Name :- Anurag Avinash Shevale
#Roll No :- 20
#Class :- BE Comp I
```

In [2]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from sklearn.neighbors import KNeighborsClassifier
```

```
/home/admin1/anaconda3/lib/python3.9/site-packages/scipy/__init__.p
y:146: UserWarning: A NumPy version >=1.16.5 and <1.23.0 is require
d for this version of SciPy (detected version 1.26.0
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxver
sion}"
```

In [3]:
```
data = pd.read_csv("/home/admin1/Desktop/Anurag/emails.csv")
```

In [4]:
```
data.head(10)
```

Out[4]:

	Email No.	the	to	ect	and	for	of	a	you	hou	...	connevey	jay	valued	lay	infrastr
0	Email 1	0	0	1	0	0	0	2	0	0	...	0	0	0	0	
1	Email 2	8	13	24	6	6	2	102	1	27	...	0	0	0	0	
2	Email 3	0	0	1	0	0	0	8	0	0	...	0	0	0	0	
3	Email 4	0	5	22	0	5	1	51	2	10	...	0	0	0	0	
4	Email 5	7	6	17	1	5	2	57	0	9	...	0	0	0	0	
5	Email 6	4	5	1	4	2	3	45	1	0	...	0	0	0	0	
6	Email 7	5	3	1	3	2	1	37	0	0	...	0	0	0	0	
7	Email 8	0	2	2	3	1	2	21	6	0	...	0	0	0	0	
8	Email 9	2	2	3	0	0	1	18	0	0	...	0	0	0	0	
9	Email 10	4	4	35	0	1	0	49	1	16	...	0	0	0	0	

10 rows × 3002 columns

In [5]:
```
data.tail(10)
```

Out[5]:

	Email No.	the	to	ect	and	for	of	a	you	hou	...	connevey	jay	valued	lay	infra
5162	Email 5163	2	3	1	2	1	2	32	0	0	...	0	0	0	0	
5163	Email 5164	0	0	1	0	0	0	1	0	0	...	0	0	0	0	
5164	Email 5165	21	18	3	1	6	4	106	1	2	...	0	0	0	0	
5165	Email 5166	1	0	1	0	3	1	12	1	0	...	0	0	0	1	
5166	Email 5167	1	0	1	1	0	0	4	0	0	...	0	0	0	0	
5167	Email 5168	2	2	2	3	0	0	32	0	0	...	0	0	0	0	
5168	Email 5169	35	27	11	2	6	5	151	4	3	...	0	0	0	0	
5169	Email 5170	0	0	1	1	0	0	11	0	0	...	0	0	0	0	
5170	Email 5171	2	7	1	0	2	1	28	2	0	...	0	0	0	0	
5171	Email 5172	22	24	5	1	6	5	148	8	2	...	0	0	0	0	

10 rows × 3002 columns

In [7]:
```
data.isnull().sum()
```

Out[7]:
```
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]:
```
data.describe()
```

Out[9]:

	the	to	ect	and	for	of	
count	5172.000000	5172.000000	5172.000000	5172.000000	5172.000000	5172.000000	5172.0
mean	6.640565	6.188128	5.143852	3.075599	3.124710	2.627030	55.5
std	11.745009	9.534576	14.101142	6.045970	4.680522	6.229845	87.5
min	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.0
25%	0.000000	1.000000	1.000000	0.000000	1.000000	0.000000	12.0
50%	3.000000	3.000000	1.000000	1.000000	2.000000	1.000000	28.0
75%	8.000000	7.000000	4.000000	3.000000	4.000000	2.000000	62.2
max	210.000000	132.000000	344.000000	89.000000	47.000000	77.000000	1898.0

8 rows × 3001 columns

In [10]:
```
#Selecting number of columns from the dataset
X = data.iloc[:,1:3001]
X
```

Out[10]:

	the	to	ect	and	for	of	a	you	hou	in	...	enhancements	connevey	jay	valu
0	0	0	1	0	0	0	2	0	0	0	...	0	0	0	
1	8	13	24	6	6	2	102	1	27	18	...	0	0	0	
2	0	0	1	0	0	0	8	0	0	4	...	0	0	0	
3	0	5	22	0	5	1	51	2	10	1	...	0	0	0	
4	7	6	17	1	5	2	57	0	9	3	...	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
5167	2	2	2	3	0	0	32	0	0	5	...	0	0	0	
5168	35	27	11	2	6	5	151	4	3	23	...	0	0	0	
5169	0	0	1	1	0	0	11	0	0	1	...	0	0	0	
5170	2	7	1	0	2	1	28	2	0	8	...	0	0	0	
5171	22	24	5	1	6	5	148	8	2	23	...	0	0	0	

5172 rows × 3000 columns

In [11]:
```
Y = data.iloc[:,-1].values
Y
```

Out[11]:
```
array([0, 0, 0, ..., 1, 1, 0])
```

In [12]:
```
#Perform the train test model from SKLearn Library
train_x,test_x,train_y,test_y = train_test_split(X,Y,test_size = 0.5)
```

In [13]:
```
svc = SVC(C=1.0,kernel='rbf',gamma='auto')
# C here is the regularization parameter. Here, L2 penalty is used(de
# As C increases, model overfits.
# Kernel here is the radial basis function kernel.
# gamma (only used for rbf kernel) : As gamma increases, model overfi
svc.fit(train_x,train_y)
y_pred2 = svc.predict(test_x)
print("Accuracy Score for SVC : ", accuracy_score(y_pred2,test_y))
```

```
Accuracy Score for SVC :  0.8797370456303171
```

In [14]:
```
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size =
```

In [15]:
```
knn = KNeighborsClassifier(n_neighbors=7)
```

In [16]:
```
knn.fit(X_train, y_train)
```

Out[16]:
```
KNeighborsClassifier(n_neighbors=7)
```

In [17]:
```
print(knn.predict(X_test))
```

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

In [18]:
```
print(knn.score(X_test, y_test))
```

```
0.8685990338164251
```

In [ ]: