In [1]:

import pandas as pd import numpy as np import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn.svm import SVC, LinearSVC

from sklearn.neighbors import KNeighborsClassifier

from sklearn import metrics

from sklearn import preprocessing

## Loading the Dataset

First we load the dataset and ﬁnd out the number of columns, rows, NULL values, etc.

In [2]:

df = pd.read\_csv('emails.csv')

In [3]:

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 [4]:

df.head()

Out[4]:

No.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Email the | to | ect | and | for | of | a | you | hou ... connevey | jay | valued | lay | infrastructure |
| 0 Email 0 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 0 ... 0 | 0 | 0 | 0 | 0 |
| 1 Email 8 | 13 | 24 | 6 | 6 | 2 | 102 | 1 | 27 ... 0 | 0 | 0 | 0 | 0 |
| 2 Email 0 | 0 | 1 | 0 | 0 | 0 | 8 | 0 | 0 ... 0 | 0 | 0 | 0 | 0 |
| 3 Email 0 | 5 | 22 | 0 | 5 | 1 | 51 | 2 | 10 ... 0 | 0 | 0 | 0 | 0 |
| 4 Email 7 | 6 | 17 | 1 | 5 | 2 | 57 | 0 | 9 ... 0 | 0 | 0 | 0 | 0 |

1

2

3

4

5

5 rows × 3002 columns

In [5]:

df.dtypes

Out[5]: Email No. object the int64

to int64

ect int64

and int64

military int64

allowing int64

ff int64

dry int64

Prediction int64 Length: 3002, dtype: object

## Cleaning

In [6]:

df.drop(columns=['Email No.'], inplace=True)

In [7]:

df.isna().sum()

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Out[7]: | the | 0 | |  | | | | | | |
|  | to | 0 | |
|  | ect | 0 | |
|  | and | 0 | |
|  | for | 0 | |
|  | military | 0 | |
|  | allowing | 0 | |
|  | ff | 0 | |
|  | dry | 0 | |
|  | Prediction Length: 3001, | 0  dtype: int64 | |
| In [8]: | df.describe() |  | |
| Out[8]: |  | the | | to | ect | and | for | of | | a |
| count5172.000000 | | | 5172.000000 | 5172.000000 | | 5172.000000 5172.000000 5172.000000 5172.000000 | | | | |
| mean6.640565 | | 6.188128 | | 5.143852 | | 3.075599 | 3.124710 | 2.627030 | 55.517401 | |
| std11.745009 | | 9.534576 | | 14.101142 | | 6.045970 | 4.680522 | 6.229845 | 87.574172 | |
| min0.000000 | | 0.000000 | | 1.000000 | | 0.000000 | 0.000000 | 0.000000 | 0.000000 | |
| 25%0.000000 | | 1.000000 | | 1.000000 | | 0.000000 | 1.000000 | 0.000000 | 12.000000 | |
| 50%3.000000 | | 3.000000 | | 1.000000 | | 1.000000 | 2.000000 | 1.000000 | 28.000000 | |
| 75%8.000000 | | 7.000000 | | 4.000000 | | 3.000000 | 4.000000 | 2.000000 | 62.250000 | |
| max210.000000 | | 132.000000 | | 344.000000 | | 89.000000 | 47.000000 | 77.000000 | 1898.000000 | |

8 rows × 3001 columns

# Separating the features and the labels

In [9]:

X=df.iloc[:, :df.shape[1]-1]

y=df.iloc[:, -1] X.shape, y.shape

*#Independent Variables*

*#Dependent Variable*

Out[9]: ((5172, 3000), (5172,))

# Splitting the Dataset

Training and Test Set

In [10]:

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.15, random\_state

# Machine Learning models

The following 5 models are used:

1. K-Nearest Neighbors
2. Linear SVM
3. Polynomial SVM
4. RBF SVM
5. Sigmoid SVM

In [11]:

models = {

"K-Nearest Neighbors": KNeighborsClassifier(n\_neighbors=2), "Linear SVM":LinearSVC(random\_state=8, max\_iter=900000), "Polynomical SVM":SVC(kernel="poly", degree=2, random\_state=8), "RBF SVM":SVC(kernel="rbf", random\_state=8),

"Sigmoid SVM":SVC(kernel="sigmoid", random\_state=8)

}

## Fit and predict on each model

Each model is trained using the train set and predictions are made based on the test set. Accuracy scores are calculated for each model.

In [12]:

for model\_name, model in models.items(): y\_pred=model.fit(X\_train, y\_train).predict(X\_test)

print(f"Accuracy for {model\_name} model \t: {metrics.accuracy\_score(y\_test, y\_pred

Accuracy for K-Nearest Neighbors model : 0.8878865979381443 Accuracy for Linear SVM model : 0.9755154639175257 Accuracy for Polynomical SVM model : 0.7615979381443299 Accuracy for RBF SVM model : 0.8182989690721649 Accuracy for Sigmoid SVM model : 0.6237113402061856