```python
# Aditya Desai
# Roll no : A-16
# BE-A

import numpy as np
import pandas as pd

data = pd.read_csv("/home/kj-comp/ML/Email dataset/emails.csv")

data
```

```
      Email No.  the  to  ect  and  for  of    a  you  hou  ...
connevey  \
0        Email 1    0   0    1    0    0   0    2    0    0  ...
0
1        Email 2    8  13   24    6    6   2  102    1   27  ...
0
2        Email 3    0   0    1    0    0   0    8    0    0  ...
0
3        Email 4    0   5   22    0    5   1   51    2   10  ...
0
4        Email 5    7   6   17    1    5   2   57    0    9  ...
0
...          ...  ...  ..  ...  ...  ...  ..  ...  ...  ...  ...
...
5167  Email 5168    2   2    2    3    0   0   32    0    0  ...
0
5168  Email 5169   35  27   11    2    6   5  151    4    3  ...
0
5169  Email 5170    0   0    1    1    0   0   11    0    0  ...
0
5170  Email 5171    2   7    1    0    2   1   28    2    0  ...
0
5171  Email 5172   22  24    5    1    6   5  148    8    2  ...
0

      jay  valued  lay  infrastructure  military  allowing  ff  dry  \
0       0       0    0               0         0         0   0    0
1       0       0    0               0         0         0   1    0
2       0       0    0               0         0         0   0    0
3       0       0    0               0         0         0   0    0
4       0       0    0               0         0         0   1    0
...   ...     ...  ...             ...       ...       ...  ..  ...
5167    0       0    0               0         0         0   0    0
5168    0       0    0               0         0         0   1    0
5169    0       0    0               0         0         0   0    0
5170    0       0    0               0         0         0   1    0
5171    0       0    0               0         0         0   0    0

      Prediction
```

```
0              0
1              0
2              0
3              0
4              0
...          ...
5167           0
5168           0
5169           1
5170           1
5171           0

[5172 rows x 3002 columns]
```

```
data.columns
```

```
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)
```

```
data.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
```

```
data.isnull()
```

```
      Email No.    the     to    ect    and    for     of      a
you  \
0         False  False  False  False  False  False  False  False
False
1         False  False  False  False  False  False  False  False
False
2         False  False  False  False  False  False  False  False
False
3         False  False  False  False  False  False  False  False
False
4         False  False  False  False  False  False  False  False
False
...         ...    ...    ...    ...    ...    ...    ...    ...    ..
.
5167      False  False  False  False  False  False  False  False
False
5168      False  False  False  False  False  False  False  False
```

```
False
5169      False  False  False  False  False  False  False  False
False
5170      False  False  False  False  False  False  False  False
False
5171      False  False  False  False  False  False  False  False
False

        hou  ...  connevey    jay  valued    lay  infrastructure
military  \
0     False  ...     False  False   False  False           False
False
1     False  ...     False  False   False  False           False
False
2     False  ...     False  False   False  False           False
False
3     False  ...     False  False   False  False           False
False
4     False  ...     False  False   False  False           False
False
...     ...  ...       ...    ...     ...    ...             ...
...
5167  False  ...     False  False   False  False           False
False
5168  False  ...     False  False   False  False           False
False
5169  False  ...     False  False   False  False           False
False
5170  False  ...     False  False   False  False           False
False
5171  False  ...     False  False   False  False           False
False

      allowing     ff    dry  Prediction
0        False  False  False       False
1        False  False  False       False
2        False  False  False       False
3        False  False  False       False
4        False  False  False       False
...        ...    ...    ...         ...
5167     False  False  False       False
5168     False  False  False       False
5169     False  False  False       False
5170     False  False  False       False
5171     False  False  False       False

[5172 rows x 3002 columns]
```

```
data.isnull().sum()
```

```
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
```

```
data.head()
```

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

   valued  lay  infrastructure  military  allowing  ff  dry
Prediction
0       0    0               0         0         0   0    0
0
1       0    0               0         0         0   1    0
0
2       0    0               0         0         0   0    0
0
3       0    0               0         0         0   0    0
0
4       0    0               0         0         0   1    0
0

[5 rows x 3002 columns]
```

```
data.head(10)
```

```
  Email No.  the  to  ect  and  for  of    a  you  hou  ...  connevey
jay  \
0   Email 1    0   0    1    0    0   0    2    0    0  ...         0
0
1   Email 2    8  13   24    6    6   2  102    1   27  ...         0
```

```
0
2   Email 3    0   0    1    0    0   0    8    0    0  ...          0
0
3   Email 4    0   5   22    0    5   1   51    2   10  ...          0
0
4   Email 5    7   6   17    1    5   2   57    0    9  ...          0
0
5   Email 6    4   5    1    4    2   3   45    1    0  ...          0
0
6   Email 7    5   3    1    3    2   1   37    0    0  ...          0
0
7   Email 8    0   2    2    3    1   2   21    6    0  ...          0
0
8   Email 9    2   2    3    0    0   1   18    0    0  ...          0
0
9  Email 10    4   4   35    0    1   0   49    1   16  ...          0
0

   valued  lay  infrastructure  military  allowing  ff  dry
Prediction
0       0    0               0         0         0   0    0
0
1       0    0               0         0         0   1    0
0
2       0    0               0         0         0   0    0
0
3       0    0               0         0         0   0    0
0
4       0    0               0         0         0   1    0
0
5       0    0               0         0         0   0    0
1
6       0    0               0         0         0   0    0
0
7       0    0               0         0         0   1    0
1
8       0    0               0         0         0   0    0
0
9       0    0               0         0         0   0    0
0

[10 rows x 3002 columns]

data.tail()

       Email No.  the  to  ect  and  for  of    a  you  hou  ...
connevey  \
5167  Email 5168    2   2    2    3    0   0   32    0    0  ...
0
5168  Email 5169   35  27   11    2    6   5  151    4    3  ...
```

```
0
5169  Email 5170    0   0    1    1    0   0   11    0    0  ...
0
5170  Email 5171    2   7    1    0    2   1   28    2    0  ...
0
5171  Email 5172   22  24    5    1    6   5  148    8    2  ...
0

      jay  valued  lay  infrastructure  military  allowing  ff  dry  \
5167    0       0    0               0         0         0   0    0
5168    0       0    0               0         0         0   1    0
5169    0       0    0               0         0         0   0    0
5170    0       0    0               0         0         0   1    0
5171    0       0    0               0         0         0   0    0

      Prediction
5167           0
5168           0
5169           1
5170           1
5171           0

[5 rows x 3002 columns]
```

```
data.drop(columns = ['Email No.'], inplace = True)
data.tail()
```

```
      the  to  ect  and  for  of    a  you  hou  in  ...  connevey
jay  \
5167    2   2    2    3    0   0   32    0    0   5  ...         0
0
5168   35  27   11    2    6   5  151    4    3  23  ...         0
0
5169    0   0    1    1    0   0   11    0    0   1  ...         0
0
5170    2   7    1    0    2   1   28    2    0   8  ...         0
0
5171   22  24    5    1    6   5  148    8    2  23  ...         0
0

      valued  lay  infrastructure  military  allowing  ff  dry
Prediction
5167       0    0               0         0         0   0    0
0
5168       0    0               0         0         0   1    0
0
5169       0    0               0         0         0   0    0
1
5170       0    0               0         0         0   1    0
1
```

```
5171       0    0                0         0         0   0    0
0

[5 rows x 3001 columns]
```

```
data.isnull().any().value_counts()
```

```
False    3001
dtype: int64
```

```
#Separating Features & Labels

x = data.iloc[:, :data.shape[1] -1]
y = data.iloc[:, -1]
x.shape, y.shape
```

```
((5172, 3000), (5172,))
```

```
# Splitting  into Train and Test Dataset
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.15)

# Machine Learning Models

from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC, LinearSVC
from sklearn.neural_network import MLPClassifier

from sklearn.neighbors import KNeighborsClassifier  #KNN
models = {"Logistic Regression": LogisticRegression(solver='lbfgs',
max_iter = 2000),
          "Linear SVM": LinearSVC(max_iter = 3000),
          "Polynomial SVM": SVC(kernel = "poly", degree = 2),
          "RBF SVM": SVC(kernel = "rbf"),
          "Sigmoid SVM": SVC(kernel = "sigmoid"),
          "Multi-layer Perception Classification":
MLPClassifier(hidden_layer_sizes = [20,20]),
          "K-Nearest Neighbors": KNeighborsClassifier(n_neighbors =
20)
         }

# Predict Accuracy Score for Each Model
from sklearn.metrics import accuracy_score
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 is :
{accuracy_score(y_test, y_pred)}")
```

```
Accuracy for Logistic Regression model is : 0.9716494845360825
```

```
/home/kj-comp/anaconda3/lib/python3.7/site-packages/sklearn/svm/
_base.py:947: ConvergenceWarning: Liblinear failed to converge,
increase the number of iterations.
  "the number of iterations.", ConvergenceWarning)

Accuracy for Linear SVM model is : 0.9536082474226805
Accuracy for Polynomial SVM model is : 0.7525773195876289
Accuracy for RBF SVM model is : 0.8054123711340206
Accuracy for Sigmoid SVM model is : 0.6082474226804123
Accuracy for Multi-layer Perception Classification model is :
0.9884020618556701
Accuracy for K-Nearest Neighbors model is : 0.8698453608247423

# Accuracy in percentage is:-
# Logistic Regression model: 97%
# Linear SVM model: 95%
# Polynomial SVM model: 75%
# RBF SVM model: 80%
# Sigmoid SVM model: 61%
# Multi-layer Perception Classification model: 98%
# K-Nearest Neighbors model: 86%
```