```
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

# Load dataset
df = pd.read_csv("emails.csv")

df.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]
```

```
df.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
```

```python
X = df.iloc[:,1:3001]  # word frequency features
X
```

```
      the  to  ect  and  for  of    a  you  hou  in  ...  enhancements  
\
0       0   0    1    0    0   0    2    0    0   0  ...             0  

1       8  13   24    6    6   2  102    1   27  18  ...             0  

2       0   0    1    0    0   0    8    0    0   4  ...             0  

3       0   5   22    0    5   1   51    2   10   1  ...             0  

4       7   6   17    1    5   2   57    0    9   3  ...             0  

...   ...  ..  ...  ...  ...  ..  ...  ...  ...  ..  ...           ...  

5167    2   2    2    3    0   0   32    0    0   5  ...             0  

5168   35  27   11    2    6   5  151    4    3  23  ...             0  

5169    0   0    1    1    0   0   11    0    0   1  ...             0  

5170    2   7    1    0    2   1   28    2    0   8  ...             0  

5171   22  24    5    1    6   5  148    8    2  23  ...             0  


      connevey  jay  valued  lay  infrastructure  military  allowing  
ff  dry
0            0    0       0    0               0         0         0  
0    0
1            0    0       0    0               0         0         0  
1    0
2            0    0       0    0               0         0         0  
0    0
3            0    0       0    0               0         0         0  
0    0
4            0    0       0    0               0         0         0  
1    0
...        ...  ...     ...  ...             ...       ...       ...  
..  ...
5167         0    0       0    0               0         0         0  
0    0
5168         0    0       0    0               0         0         0  
1    0
5169         0    0       0    0               0         0         0  
0    0
```

```
5170         0    0       0    0               0         0         0
1    0
5171         0    0       0    0               0         0         0
0    0

[5172 rows x 3000 columns]
```

```
Y = df.iloc[:,-1].values # 1 = spam, 0 = not spam
Y
```

```
array([0, 0, 0, ..., 1, 1, 0], shape=(5172,))
```

```
# Split data
X_train, X_test, y_train, y_test = train_test_split(X, Y,
test_size=0.25, random_state=42)
```

```
from sklearn.metrics import classification_report, confusion_matrix
```

```
# -------- Support Vector Machine --------
svc = SVC(C=1.0, kernel='rbf', gamma='auto')
svc.fit(X_train, y_train)
svc_pred = svc.predict(X_test)
```

```
SVM Accuracy: 0.8932714617169374
SVM Classification Report:
               precision    recall  f1-score   support

           0       0.90      0.96      0.93       913
           1       0.87      0.74      0.80       380

    accuracy                           0.89      1293
   macro avg       0.89      0.85      0.87      1293
weighted avg       0.89      0.89      0.89      1293

SVM Confusion Matrix:
 [[872  41]
 [ 97 283]]
```

```
print("SVM Accuracy:", accuracy_score(y_test, svc_pred))
print("SVM Classification Report:\n", classification_report(y_test,
svc_pred))
print("SVM Confusion Matrix:\n", confusion_matrix(y_test, svc_pred))
```

```
SVM Accuracy: 0.8932714617169374
SVM Classification Report:
               precision    recall  f1-score   support

           0       0.90      0.96      0.93       913
           1       0.87      0.74      0.80       380

    accuracy                           0.89      1293
```

```
   macro avg       0.89      0.85      0.87      1293
weighted avg       0.89      0.89      0.89      1293

SVM Confusion Matrix:
 [[872  41]
 [ 97 283]]
```

```
# -------- K-Nearest Neighbors --------
knn = KNeighborsClassifier(n_neighbors=7)
knn.fit(X_train, y_train)
knn_pred = knn.predict(X_test)

print("KNN Accuracy:", knn.score(X_test, y_test))
print("KNN Classification Report:\n", classification_report(y_test,
knn_pred))
print("KNN Confusion Matrix:\n", confusion_matrix(y_test, knn_pred))
```

```
KNN Accuracy: 0.8685990338164251
KNN Classification Report:
               precision    recall  f1-score   support

           0       0.94      0.87      0.90       739
           1       0.73      0.86      0.79       296

    accuracy                           0.87      1035
   macro avg       0.83      0.87      0.85      1035
weighted avg       0.88      0.87      0.87      1035

KNN Confusion Matrix:
 [[645  94]
 [ 42 254]]
```