ass2ml

[3]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report
```

[4]:
```
df=pd.read_csv("emails.csv")
```

[6]:
```
# Data preprocessing
X = df.drop(columns=['Email No.', 'Prediction'])
y = df['Prediction']
```

[7]:
```
# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[10]:
```
vectorizer = TfidfVectorizer()
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)
```

[12]:
```
knn = KNeighborsClassifier()
knn.fit(X_train, y_train)
y_pred_knn = knn.predict(X_test)
```

[13]:
```
# SVM Classifier
svm = SVC()
svm.fit(X_train, y_train)
y_pred_svm = svm.predict(X_test)
```

[14]:
```
# Evaluation
knn_accuracy = accuracy_score(y_test, y_pred_knn)
svm_accuracy = accuracy_score(y_test, y_pred_svm)
```

[15]:
```
# Evaluation
knn_accuracy = accuracy_score(y_test, y_pred_knn)
```

```
svm_accuracy = accuracy_score(y_test, y_pred_svm)
```

[16]:
```
print("KNN Classification Report: \n", classification_report(y_test,␣
 ↪y_pred_knn))
print("SVM Classification Report: \n", classification_report(y_test,␣
 ↪y_pred_svm))
```

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

           0       0.93      0.87      0.90       739
           1       0.73      0.83      0.78       296

    accuracy                           0.86      1035
   macro avg       0.83      0.85      0.84      1035
weighted avg       0.87      0.86      0.87      1035

SVM Classification Report:
               precision    recall  f1-score   support

           0       0.80      0.98      0.88       739
           1       0.91      0.40      0.56       296

    accuracy                           0.82      1035
   macro avg       0.86      0.69      0.72      1035
weighted avg       0.83      0.82      0.79      1035
```

[ ]:
```
```

[20]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report,␣
 ↪confusion_matrix

# Load the dataset
emails_df = pd.read_csv('emails.csv')

# Features (X) are all columns except the last one
X = emails_df.iloc[:, :-1]  # All columns except the last one
y = emails_df['Prediction']  # The last column is the label

# Identify non-numeric columns
non_numeric_columns = X.select_dtypes(include=['object']).columns
```

```
# Drop non-numeric columns
X = X.drop(columns=non_numeric_columns)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# K-Nearest Neighbors Classifier
knn = KNeighborsClassifier()
knn.fit(X_train, y_train)

# Support Vector Machine Classifier
svm = SVC()
svm.fit(X_train, y_train)

# Model Evaluation
y_pred_knn = knn.predict(X_test)
y_pred_svm = svm.predict(X_test)

# Evaluate KNN
print("KNN Accuracy: ", accuracy_score(y_test, y_pred_knn))
print("KNN Classification Report: \n", classification_report(y_test,␣
 ↪y_pred_knn))

# Evaluate SVM
print("SVM Accuracy: ", accuracy_score(y_test, y_pred_svm))
print("SVM Classification Report: \n", classification_report(y_test,␣
 ↪y_pred_svm))

# Confusion matrices
print("KNN Confusion Matrix: \n", confusion_matrix(y_test, y_pred_knn))
print("SVM Confusion Matrix: \n", confusion_matrix(y_test, y_pred_svm))
```

```
KNN Accuracy:  0.8628019323671497
KNN Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.87      0.90       739
           1       0.73      0.83      0.78       296

    accuracy                           0.86      1035
   macro avg       0.83      0.85      0.84      1035
weighted avg       0.87      0.86      0.87      1035

SVM Accuracy:  0.8173913043478261
SVM Classification Report:
              precision    recall  f1-score   support
```

```
              0       0.80      0.98      0.88       739
              1       0.91      0.40      0.56       296

       accuracy                           0.82      1035
      macro avg       0.86      0.69      0.72      1035
   weighted avg       0.83      0.82      0.79      1035

   KNN Confusion Matrix:
    [[646  93]
    [ 49 247]]
   SVM Confusion Matrix:
    [[727  12]
    [177 119]]
```

[ ]: