%pip install seaborn

import numpy as np

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

from sklearn.neighbors import KNeighborsClassifier

from sklearn.datasets import load\_digits

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

import matplotlib.pyplot as plt

import seaborn as sns

# 1. Load the Dataset

digits = load\_digits()

X = digits.data

y = digits.target

# 2. Visualize a Few Samples (Optional)

fig, axes = plt.subplots(2, 5, figsize=(10, 5))

for i, ax in enumerate(axes.ravel()):

ax.imshow(digits.images[i], cmap=plt.cm.gray\_r)

ax.set\_title(f"Label: {digits.target[i]}")

ax.axis('off')

plt.tight\_layout()

plt.show()

# 3. Split the Data into Training and Testing Sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# 4. Initialize the KNN Classifier

# You can experiment with different values of 'n\_neighbors' (k)

knn = KNeighborsClassifier(n\_neighbors=5)

# 5. Train the KNN Classifier

knn.fit(X\_train, y\_train)

# 6. Make Predictions on the Test Set

y\_pred = knn.predict(X\_test)

# 7. Evaluate the Model

accuracy = accuracy\_score(y\_test, y\_pred)

print(f"Accuracy of KNN: {accuracy:.4f}")

print("\nClassification Report:")

print(classification\_report(y\_test, y\_pred))

# 8. Confusion Matrix (Optional but Recommended for Detailed Evaluation)

cm = confusion\_matrix(y\_test, y\_pred)

plt.figure(figsize=(8, 6))

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',

xticklabels=digits.target\_names, yticklabels=digits.target\_names)

plt.xlabel('Predicted Label')

plt.ylabel('True Label')

plt.title('Confusion Matrix')

plt.show()

# 9. Making Predictions on New, Unseen Digits (Example)

# Assuming you have a new digit represented as a 1D array (8x8 flattened)

# new\_digit = ... (load or create your new digit data)

# new\_digit\_reshaped = new\_digit.reshape(1, -1) # Reshape to match the model's input format

# prediction = knn.predict(new\_digit\_reshaped)

# print(f"\nPrediction for the new digit: {prediction[0]}")