# Import necessary libraries

import numpy as np

import pandas as pd

from sklearn.datasets import load\_iris

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

from sklearn.preprocessing import StandardScaler

from sklearn.neighbors import KNeighborsClassifier

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

import matplotlib.pyplot as plt

# Load the Iris dataset

iris = load\_iris()

X = iris.data

y = iris.target

# Split the dataset into training and test sets

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

# Standardize the features

scaler = StandardScaler()

X\_train\_scaled = scaler.fit\_transform(X\_train)

X\_test\_scaled = scaler.transform(X\_test)

# Initialize the K-Nearest Neighbors classifier

knn = KNeighborsClassifier(n\_neighbors=5)

# Train the model

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

# Make predictions

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

# Evaluate the model

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

print(f'Accuracy: {accuracy:.2f}')

print('Classification Report:')

print(classification\_report(y\_test, y\_pred, target\_names=iris.target\_names))

# Optionally, plot decision boundaries for the first two features

def plot\_decision\_boundaries(X, y, model):

h = .02 # step size in the mesh

x\_min, x\_max = X[:, 0].min() - 1, X[:, 0].max() + 1

y\_min, y\_max = X[:, 1].min() - 1, X[:, 1].max() + 1

xx, yy = np.meshgrid(np.arange(x\_min, x\_max, h),

np.arange(y\_min, y\_max, h))

Z = model.predict(np.c\_[xx.ravel(), yy.ravel()])

Z = Z.reshape(xx.shape)

plt.contourf(xx, yy, Z, alpha=0.8)

plt.scatter(X[:, 0], X[:, 1], c=y, edgecolor='k', marker='o')

plt.xlabel('Feature 1')

plt.ylabel('Feature 2')

plt.title('Decision Boundaries')

plt.show()

# Plot decision boundaries for the first two features

plot\_decision\_boundaries(X\_train\_scaled[:, :2], y\_train, knn)