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

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy\_score

from sklearn.datasets import load\_iris

from sklearn.datasets import make\_blobs

from sklearn.cluster import KMeans

import matplotlib.pyplot as plt

import numpy as np

# Generate random data for demonstration

np.random.seed(42)

X = 2 \* np.random.rand(100, 1)

y = 4 + 3 \* X + np.random.randn(100, 1)

# 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)

# Create a Linear Regression model

model = LinearRegression()

# Train the model

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

# Make predictions on the test set

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

# Evaluate the model

mse = mean\_squared\_error(y\_test, y\_pred)

print(f'Mean Squared Error: {mse}')

iris = load\_iris()

X, y = iris.data, iris.target

# 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)

# Create a K-Nearest Neighbors classifier

knn = KNeighborsClassifier(n\_neighbors=3)

# Train the model

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

# Make predictions on the test set

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

# Evaluate the model

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

print(f'Accuracy: {accuracy}')

X, \_ = make\_blobs(n\_samples=300, centers=4, random\_state=42)

# Create a K-Means clustering model

kmeans = KMeans(n\_clusters=4)

# Fit the model to the data

kmeans.fit(X)

# Get cluster assignments and plot the results

labels = kmeans.labels\_

plt.scatter(X[:, 0], X[:, 1], c=labels, cmap='viridis')

plt.scatter(kmeans.cluster\_centers\_[:, 0], kmeans.cluster\_centers\_[:, 1], marker='X', s=200, color='red')

plt.show()