# Simple code to demonstrate supervised vs unsupervised learning

import numpy as np

import matplotlib.pyplot as plt

from sklearn.datasets import load\_digits

from sklearn.cluster import KMeans

from sklearn.neighbors import KNeighborsClassifier

from sklearn.decomposition import PCA

from sklearn.preprocessing import StandardScaler

# STEP 1: Load a simple image dataset (handwritten digits)

digits = load\_digits()

X = digits.data # The image data

y = digits.target # The labels (digits 0-9)

# Display some sample images to show what we're working with

plt.figure(figsize=(10, 3))

for i in range(10):

plt.subplot(1, 10, i + 1)

plt.imshow(digits.images[i], cmap='gray')

plt.title(f'Digit: {y[i]}')

plt.axis('off')

plt.suptitle('Sample Images (Like Teachable Machine Training Data)')

plt.tight\_layout()

plt.show()

# Standardize the data (important for clustering)

scaler = StandardScaler()

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

# STEP 2: SUPERVISED LEARNING

# This is like what Teachable Machine does

print("\n--- SUPERVISED LEARNING (LIKE TEACHABLE MACHINE) ---")

print("Here, we tell the algorithm what categories to learn.")

# Train a simple classifier (using labels)

classifier = KNeighborsClassifier(n\_neighbors=5)

classifier.fit(X\_scaled, y) # We use the labels here!

# Test on a few examples

test\_indices = np.random.choice(range(len(X)), 5)

predicted = classifier.predict([X\_scaled[i] for i in test\_indices])

# Show prediction results

plt.figure(figsize=(10, 2))

for i, idx in enumerate(test\_indices):

plt.subplot(1, 5, i + 1)

plt.imshow(digits.images[idx], cmap='gray')

plt.title(f'Predicted: {predicted[i]}\nActual: {y[idx]}')

plt.axis('off')

plt.suptitle('Supervised Learning Results (Using Labels)')

plt.tight\_layout()

plt.show()

# STEP 3: UNSUPERVISED LEARNING

# Here we don't use the labels - the algorithm finds patterns on its own

print("\n--- UNSUPERVISED LEARNING ---")

print("Here, the algorithm finds patterns without any labels.")

# Use K-means clustering to find groups (without using labels)

kmeans = KMeans(n\_clusters=10, random\_state=42)

cluster\_labels = kmeans.fit\_predict(X\_scaled) # No labels used here!

# Visualize the results in 2D (using PCA for dimensionality reduction)

pca = PCA(n\_components=2)

X\_2d = pca.fit\_transform(X\_scaled)

# Create side-by-side comparison

plt.figure(figsize=(15, 6))

# Plot 1: Unsupervised Clusters

plt.subplot(1, 2, 1)

plt.scatter(X\_2d[:, 0], X\_2d[:, 1], c=cluster\_labels, cmap='viridis', alpha=0.8, s=30)

plt.title('Unsupervised Learning Results\n(Computer Found These Groups On Its Own)')

plt.xlabel('Feature 1')

plt.ylabel('Feature 2')

plt.colorbar(label='Cluster ID')

# Plot 2: True Categories (Supervised)

plt.subplot(1, 2, 2)

plt.scatter(X\_2d[:, 0], X\_2d[:, 1], c=y, cmap='viridis', alpha=0.8, s=30)

plt.title('Supervised Learning Categories\n(Human-Defined Labels)')

plt.xlabel('Feature 1')

plt.ylabel('Feature 2')

plt.colorbar(label='Digit (0-9)')

plt.tight\_layout()

plt.show()

# STEP 4: Compare the clusters with actual digits

# Let's see if unsupervised learning discovered the actual digit categories

print("\n--- COMPARING SUPERVISED VS UNSUPERVISED RESULTS ---")

# Show what's in each cluster

plt.figure(figsize=(12, 8))

for cluster\_id in range(5): # Show first 5 clusters only

# Get images from this cluster

indices = np.where(cluster\_labels == cluster\_id)[0][:8] # Get up to 8 images

# Display images from this cluster

for i, idx in enumerate(indices):

plt.subplot(5, 8, cluster\_id\*8 + i + 1)

plt.imshow(digits.images[idx], cmap='gray')

plt.title(f'Digit: {y[idx]}')

plt.axis('off')

# Add cluster label

plt.subplot(5, 8, cluster\_id\*8 + 1)

plt.ylabel(f'Cluster {cluster\_id}', rotation=0, labelpad=40, fontsize=12)

plt.suptitle('What Did Unsupervised Learning Discover?\nImages in Each Cluster (First 5 Clusters Shown)')

plt.tight\_layout()

plt.show()