* **TASK 1**

# Step 1: Import necessary libraries

import pandas as pd

import numpy as np

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

from sklearn.svm import SVC

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

# Step 2: Load the Dataset

# For this example, let's assume the datasets are stored in CSV files

# Iris dataset

iris\_data = pd.read\_csv('/content/Iris.csv')

# Heart Disease dataset

heart\_data = pd.read\_csv('/content/heart.csv')

# Step 3: Data Preprocessing

# Iris dataset

X\_iris = iris\_data.drop(columns=['Species'])  # Replace 'species' with target column name in Iris dataset

y\_iris = iris\_data['Species']

# Heart Disease dataset

X\_heart = heart\_data.drop(columns=['target'])  # Replace 'target' with the target column name in Heart Disease dataset

y\_heart = heart\_data['target']

# Step 4: Train-Test Split (50-50 split)

X\_train\_iris, X\_test\_iris, y\_train\_iris, y\_test\_iris = train\_test\_split(X\_iris, y\_iris, test\_size=0.5, random\_state=42)

X\_train\_heart, X\_test\_heart, y\_train\_heart, y\_test\_heart = train\_test\_split(X\_heart, y\_heart, test\_size=0.5, random\_state=42)

# Step 5: Train SVM Classifier

# For Iris dataset

svm\_iris = SVC(kernel='linear', random\_state=42)  # You can try other kernels like 'rbf', 'poly', etc.

svm\_iris.fit(X\_train\_iris, y\_train\_iris)

# For Heart Disease dataset

svm\_heart = SVC(kernel='linear', random\_state=42)

svm\_heart.fit(X\_train\_heart, y\_train\_heart)

# Step 6: Evaluate Model

# Iris dataset evaluation

y\_pred\_iris = svm\_iris.predict(X\_test\_iris)

print("Iris Dataset - Classification Report:\n", classification\_report(y\_test\_iris, y\_pred\_iris))

print("Iris Dataset - Confusion Matrix:\n", confusion\_matrix(y\_test\_iris, y\_pred\_iris))

print("Iris Dataset - Accuracy:", accuracy\_score(y\_test\_iris, y\_pred\_iris))

# Heart Disease dataset evaluation

y\_pred\_heart = svm\_heart.predict(X\_test\_heart)

print("Heart Disease Dataset - Classification Report:\n", classification\_report(y\_test\_heart, y\_pred\_heart))

print("Heart Disease Dataset - Confusion Matrix:\n", confusion\_matrix(y\_test\_heart, y\_pred\_heart))

print("Heart Disease Dataset - Accuracy:", accuracy\_score(y\_test\_heart, y\_pred\_heart))

**//OUTPUT:**

Iris Dataset - Classification Report:

precision recall f1-score support

Iris-setosa 1.00 1.00 1.00 29

Iris-versicolor 1.00 1.00 1.00 23

Iris-virginica 1.00 1.00 1.00 23

accuracy 1.00 75

macro avg 1.00 1.00 1.00 75

weighted avg 1.00 1.00 1.00 75

Iris Dataset - Confusion Matrix:

[[29 0 0]

[ 0 23 0]

[ 0 0 23]]

Iris Dataset - Accuracy: 1.0

Heart Disease Dataset - Classification Report:

precision recall f1-score support

0 0.85 0.74 0.79 254

1 0.77 0.87 0.82 259

accuracy 0.81 513

macro avg 0.81 0.80 0.80 513

weighted avg 0.81 0.81 0.80 513

Heart Disease Dataset - Confusion Matrix:

[[187 67]

[ 33 226]]

Heart Disease Dataset - Accuracy: 0.8050682261208577

* **TASK2**

from sklearn.decomposition import PCA

import matplotlib.pyplot as plt

import seaborn as sns

# Function to apply PCA and plot the results

def apply\_pca\_and\_plot(X, y, dataset\_name, n\_components=2):

    # Step 1: Apply PCA

    pca = PCA(n\_components=n\_components)

    X\_pca = pca.fit\_transform(X)

    # Step 2: Plot the explained variance

    explained\_variance = pca.explained\_variance\_ratio\_

    print(f"{dataset\_name} Dataset - Explained Variance by Principal Components: {explained\_variance}")

    # Step 3: Visualize the Principal Components (only if n\_components is 2 for 2D visualization)

    if n\_components == 2:

        plt.figure(figsize=(8, 6))

        sns.scatterplot(x=X\_pca[:, 0], y=X\_pca[:, 1], hue=y, palette='viridis', s=60)

        plt.title(f"{dataset\_name} Dataset - PCA (2 Components)")

        plt.xlabel("Principal Component 1")

        plt.ylabel("Principal Component 2")

        plt.legend(title="Class")

        plt.show()

    return X\_pca

# Step 4: Apply PCA to Iris and Heart Disease datasets

print("Applying PCA on Iris Dataset")

X\_pca\_iris = apply\_pca\_and\_plot(X\_iris, y\_iris, "Iris")

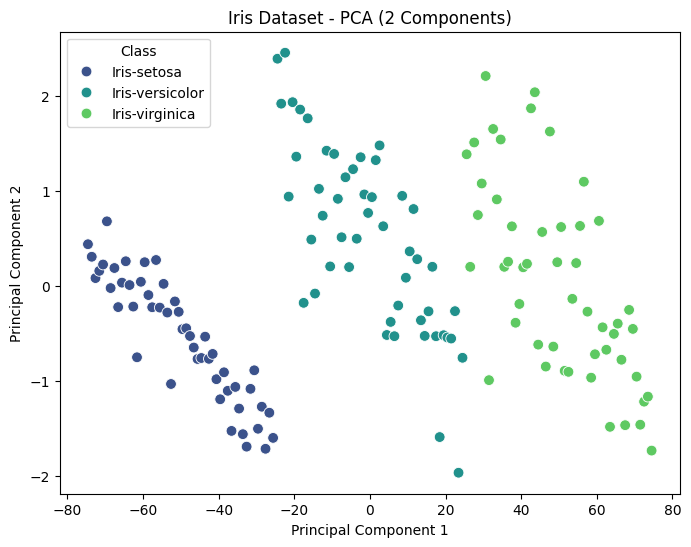
print("Applying PCA on Heart Disease Dataset")

X\_pca\_heart = apply\_pca\_and\_plot(X\_heart, y\_heart, "Heart Disease")

**//OUTPUT:**

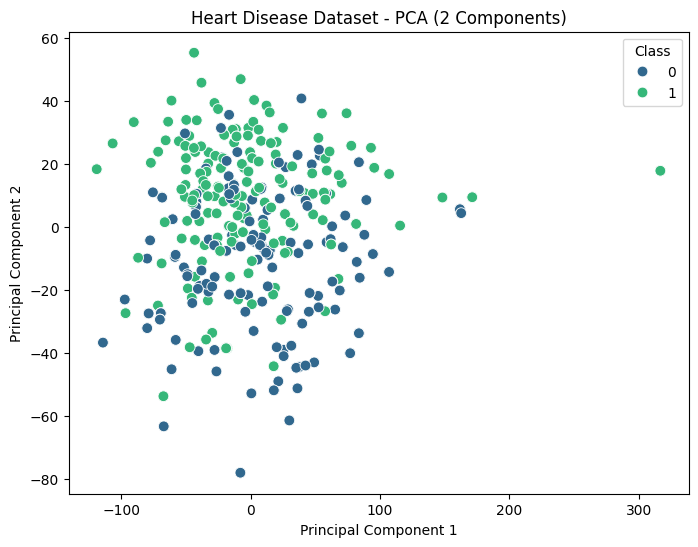
Applying PCA on Iris Dataset

Iris Dataset - Explained Variance by Principal Components: [9.99319101e-01 5.14403271e-04]



Applying PCA on Heart Disease Dataset

Heart Disease Dataset - Explained Variance by Principal Components: [0.74530931 0.15199174]



* **TASK 3**

# Apply PCA to get individual components

pca = PCA(n\_components=2)

X\_pca\_iris = pca.fit\_transform(X\_iris)

X\_pca\_heart = pca.fit\_transform(X\_heart)

# Split 1st and 2nd PCA components for 50-50 training and testing

X\_iris\_pca1 = X\_pca\_iris[:, 0].reshape(-1, 1)  # Only 1st component for Iris

X\_iris\_pca2 = X\_pca\_iris[:, 1].reshape(-1, 1)  # Only 2nd component for Iris

X\_heart\_pca1 = X\_pca\_heart[:, 0].reshape(-1, 1)  # Only 1st component for Heart Disease

X\_heart\_pca2 = X\_pca\_heart[:, 1].reshape(-1, 1)  # Only 2nd component for Heart Disease

# Split data for each component (50-50 train-test split)

X\_train\_iris\_pca1, X\_test\_iris\_pca1, y\_train\_iris, y\_test\_iris = train\_test\_split(X\_iris\_pca1, y\_iris, test\_size=0.5, random\_state=42)

X\_train\_iris\_pca2, X\_test\_iris\_pca2, y\_train\_iris, y\_test\_iris = train\_test\_split(X\_iris\_pca2, y\_iris, test\_size=0.5, random\_state=42)

X\_train\_heart\_pca1, X\_test\_heart\_pca1, y\_train\_heart, y\_test\_heart = train\_test\_split(X\_heart\_pca1, y\_heart, test\_size=0.5, random\_state=42)

X\_train\_heart\_pca2, X\_test\_heart\_pca2, y\_train\_heart, y\_test\_heart = train\_test\_split(X\_heart\_pca2, y\_heart, test\_size=0.5, random\_state=42)

# Train SVM classifier and evaluate for each dataset component

# 1st PCA Component - Iris

svm\_iris\_pca1 = SVC(kernel='linear', random\_state=42)

svm\_iris\_pca1.fit(X\_train\_iris\_pca1, y\_train\_iris)

y\_pred\_iris\_pca1 = svm\_iris\_pca1.predict(X\_test\_iris\_pca1)

print("Iris Dataset - 1st PCA Component - Classification Report:\n", classification\_report(y\_test\_iris, y\_pred\_iris\_pca1))

print("Iris Dataset - 1st PCA Component - Accuracy:", accuracy\_score(y\_test\_iris, y\_pred\_iris\_pca1))

# 2nd PCA Component - Iris

svm\_iris\_pca2 = SVC(kernel='linear', random\_state=42)

svm\_iris\_pca2.fit(X\_train\_iris\_pca2, y\_train\_iris)

y\_pred\_iris\_pca2 = svm\_iris\_pca2.predict(X\_test\_iris\_pca2)

print("Iris Dataset - 2nd PCA Component - Classification Report:\n", classification\_report(y\_test\_iris, y\_pred\_iris\_pca2))

print("Iris Dataset - 2nd PCA Component - Accuracy:", accuracy\_score(y\_test\_iris, y\_pred\_iris\_pca2))

# 1st PCA Component - Heart Disease

svm\_heart\_pca1 = SVC(kernel='linear', random\_state=42)

svm\_heart\_pca1.fit(X\_train\_heart\_pca1, y\_train\_heart)

y\_pred\_heart\_pca1 = svm\_heart\_pca1.predict(X\_test\_heart\_pca1)

print("Heart Disease Dataset - 1st PCA Component - Classification Report:\n", classification\_report(y\_test\_heart, y\_pred\_heart\_pca1))

print("Heart Disease Dataset - 1st PCA Component - Accuracy:", accuracy\_score(y\_test\_heart, y\_pred\_heart\_pca1))

# 2nd PCA Component - Heart Disease

svm\_heart\_pca2 = SVC(kernel='linear', random\_state=42)

svm\_heart\_pca2.fit(X\_train\_heart\_pca2, y\_train\_heart)

y\_pred\_heart\_pca2 = svm\_heart\_pca2.predict(X\_test\_heart\_pca2)

print("Heart Disease Dataset - 2nd PCA Component - Classification Report:\n", classification\_report(y\_test\_heart, y\_pred\_heart\_pca2))

print("Heart Disease Dataset - 2nd PCA Component - Accuracy:", accuracy\_score(y\_test\_heart, y\_pred\_heart\_pca2))

**//OUTPUT:**

Iris Dataset - 1st PCA Component - Classification Report:

precision recall f1-score support

Iris-setosa 1.00 1.00 1.00 29

Iris-versicolor 1.00 1.00 1.00 23

Iris-virginica 1.00 1.00 1.00 23

accuracy 1.00 75

macro avg 1.00 1.00 1.00 75

weighted avg 1.00 1.00 1.00 75

Iris Dataset - 1st PCA Component - Accuracy: 1.0

Iris Dataset - 2nd PCA Component - Classification Report:

precision recall f1-score support

Iris-setosa 0.54 0.52 0.53 29

Iris-versicolor 0.59 0.57 0.58 23

Iris-virginica 0.24 0.26 0.25 23

accuracy 0.45 75

macro avg 0.46 0.45 0.45 75

weighted avg 0.46 0.45 0.46 75

Iris Dataset - 2nd PCA Component - Accuracy: 0.4533333333333333

Heart Disease Dataset - 1st PCA Component - Classification Report:

precision recall f1-score support

0 0.55 0.37 0.44 254

1 0.53 0.70 0.61 259

accuracy 0.54 513

macro avg 0.54 0.54 0.52 513

weighted avg 0.54 0.54 0.52 513

Heart Disease Dataset - 1st PCA Component - Accuracy: 0.5380116959064327

Heart Disease Dataset - 2nd PCA Component - Classification Report:

precision recall f1-score support

0 0.72 0.66 0.69 254

1 0.69 0.75 0.72 259

accuracy 0.70 513

macro avg 0.70 0.70 0.70 513

weighted avg 0.70 0.70 0.70 513

Heart Disease Dataset - 2nd PCA Component - Accuracy: 0.7037037037037037