**Name: Jigar Siddhpura SAPID:** 60004200155

**DIV: C/C2 Branch:** Computer Engineering

ML - Experiment 8 - SVM

from google.colab import drive

drive.mount('/content/gdrive')

import numpy as np

import pandas as pd

from sklearn.preprocessing import StandardScaler

from sklearn.decomposition import PCA

from sklearn.svm import SVC

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

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

import matplotlib.pyplot as plt

from sklearn.preprocessing import KBinsDiscretizer

***LINEAR***

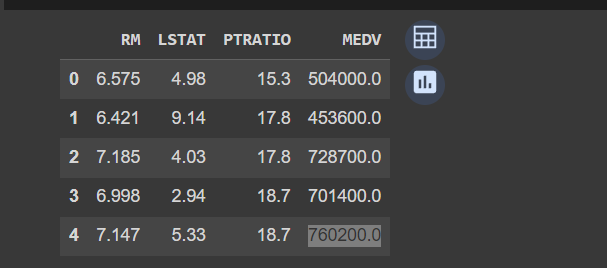
**DATASET**

# Load the dataset

dataset\_path = '/content/gdrive/MyDrive/ML/california.csv'

data = pd.read\_csv(dataset\_path)

data.head()



X = data.drop(columns=['medv'])  # Features

y = data['medv']  # Target variable

kbins = KBinsDiscretizer(n\_bins=3, encode='ordinal', strategy='quantile')

y\_discrete = kbins.fit\_transform(y.values.reshape(-1, 1))

scaler = StandardScaler()

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

pca = PCA(n\_components=2)

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

svm = SVC(kernel='linear')

svm.fit(X\_pca, y\_discrete)

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

h = .02

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

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

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

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

Z = Z.reshape(xx.shape)

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

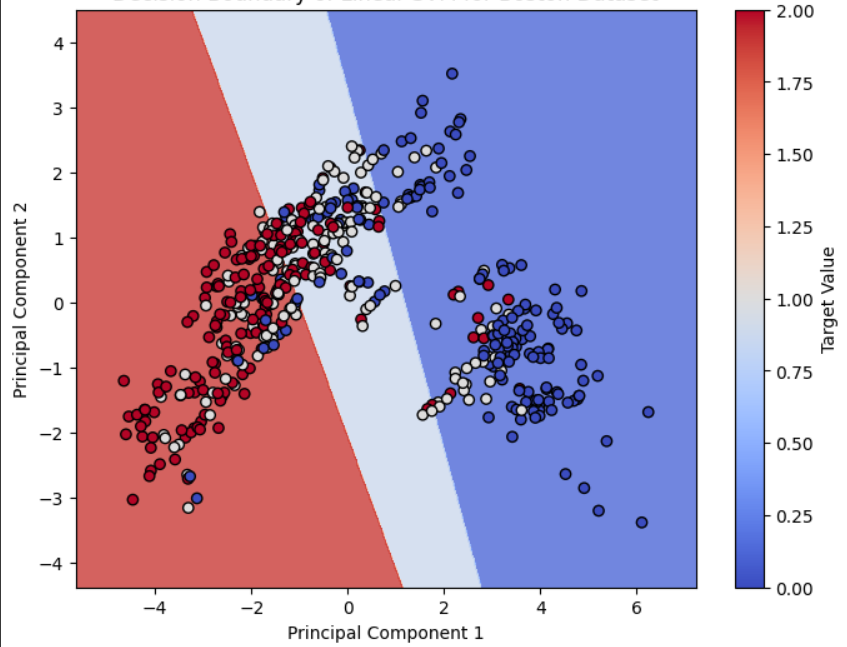
plt.scatter(X\_pca[:, 0], X\_pca[:, 1], c=y\_discrete, cmap='coolwarm', marker='o', edgecolors='k')

plt.xlabel('Principal Component 1')

plt.ylabel('Principal Component 2')

plt.colorbar(label='Target Value')

plt.show()



data = pd.read\_csv('/content/gdrive/MyDrive/ML/california.csv')

bins = [0, 453600.0, 760200.0, float('inf')]

labels = ['low', 'medium', 'high']

data['medv\_category'] = pd.cut(data['MEDV'], bins=bins, labels=labels)

X = data.drop(columns=['MEDV', 'medv\_category'])

y = data['medv\_category']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

kernels = ['linear', 'poly', 'rbf', 'sigmoid']

for kernel in kernels:

    svm = SVC(kernel=kernel)

    svm.fit(X\_train, y\_train)

    y\_pred = svm.predict(X\_test)

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

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

    class\_report = classification\_report(y\_test, y\_pred)

    print(f"Kernel: {kernel}")

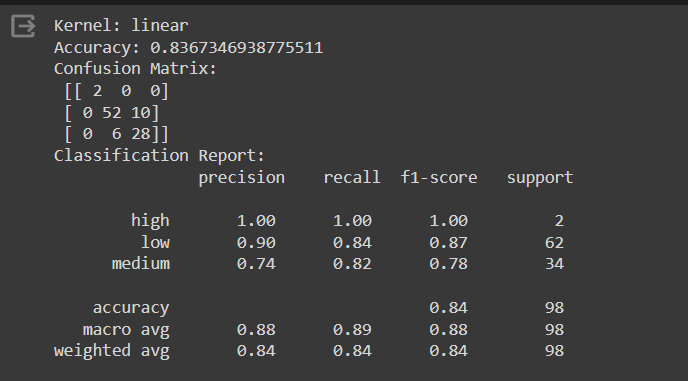
    print("Accuracy:", accuracy)

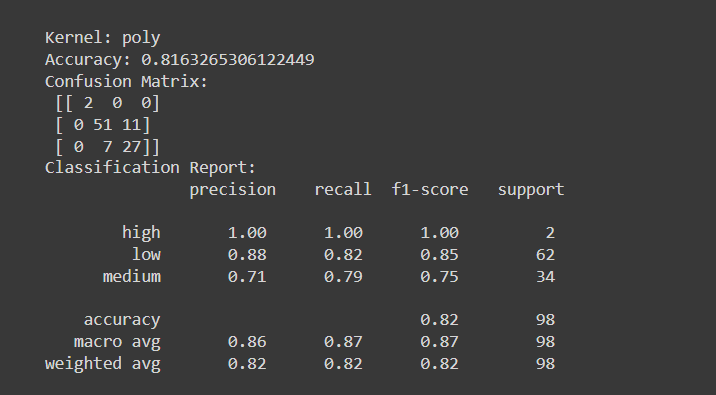
    print("Confusion Matrix:\n", confusion\_mat)

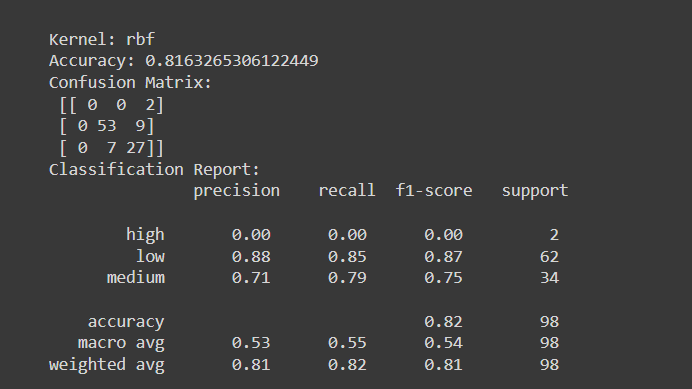
    print("Classification Report:\n", class\_report)

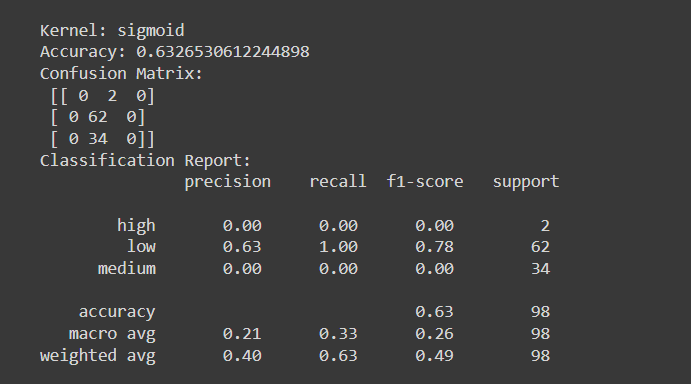
print()

**OUTPUT**







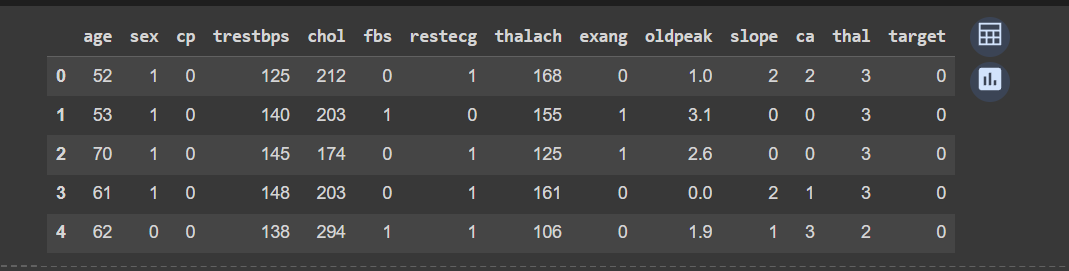


***NON - LINEAR***

**DATASET**

df = pd.read\_csv('/content/gdrive/MyDrive/ML/heart.csv')

df.head()



dataset\_url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/processed.cleveland.data'

names = ['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg',

         'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target']

data = pd.read\_csv(dataset\_url, names=names)

data.replace('?', np.nan, inplace=True)

data.dropna(inplace=True)

X = data.drop(columns=['target'])

y = data['target']

scaler = StandardScaler()

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

pca = PCA(n\_components=2)

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

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_pca, y, test\_size=0.2, random\_state=42)

svm = SVC(kernel='rbf', gamma='scale')

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

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

h = .02  # step size in the mesh

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

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

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

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

Z = Z.reshape(xx.shape)

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

plt.scatter(X\_pca[:, 0], X\_pca[:, 1], c=y, cmap='coolwarm', marker='o', edgecolors='k')

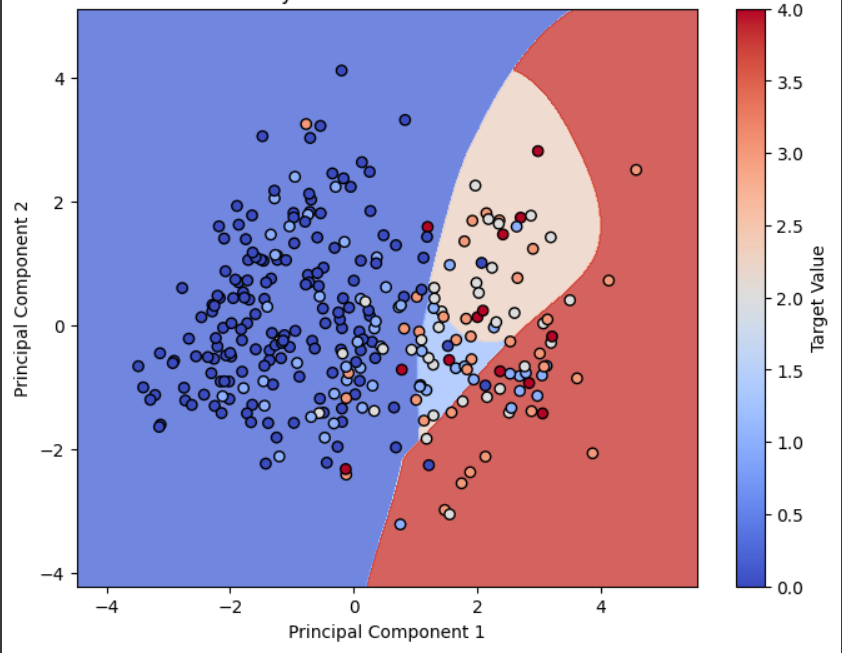
plt.title('Decision Boundary of Non-linear SVM for Heart Disease')

plt.xlabel('Principal Component 1')

plt.ylabel('Principal Component 2')

plt.colorbar(label='Target Value')

plt.show()



dataset\_url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/processed.cleveland.data'

names = ['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg',

         'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target']

data = pd.read\_csv(dataset\_url, names=names)

data.replace('?', np.nan, inplace=True)

data.dropna(inplace=True)

X = data.drop(columns=['target'])

y = data['target']

scaler = StandardScaler()

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

pca = PCA(n\_components=2)

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

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_pca, y, test\_size=0.2, random\_state=42)

kernels = ['linear', 'poly', 'rbf', 'sigmoid']

results = {}

for kernel in kernels:

    svm = SVC(kernel=kernel, gamma='scale')

    svm.fit(X\_train, y\_train)

    y\_pred = svm.predict(X\_test)

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

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

    report = classification\_report(y\_test, y\_pred)

    results[kernel] = {'accuracy': accuracy, 'confusion\_matrix': cm, 'classification\_report': report}

for kernel, result in results.items():

    print(f"Kernel: {kernel}")

    print(f"Accuracy: {result['accuracy']}")

    print("Confusion Matrix:")

    print(result['confusion\_matrix'])

    print("Classification Report:")

    print(result['classification\_report'])

    print()

**OUTPUT**

