# MACHINE LEARNING

**LAB WORK 6**

**NAME: CHETHAN Y**

**USN: 22BTRAD011**

**Implementing SVM**

**Code:**

# Import necessary libraries

import numpy as np

import matplotlib.pyplot as plt

from sklearn.datasets import make\_classification

from sklearn.svm import SVC

# Create a small synthetic dataset

X, y = make\_classification(n\_samples=100, n\_features=2, n\_classes=2, n\_clusters\_per\_class=1, n\_redundant=0, random\_state=42)

# Create an SVM classifier

svm\_classifier = SVC(kernel='linear')

svm\_classifier.fit(X, y)

# Visualize the decision boundary

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

# Plot data points

plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Paired, edgecolors='k', marker='o', s=100)

# Plot the decision boundary

ax = plt.gca()

xlim = ax.get\_xlim()

ylim = ax.get\_ylim()

# Create grid to evaluate model

xx, yy = np.meshgrid(np.linspace(xlim[0], xlim[1], 100), np.linspace(ylim[0], ylim[1], 100))

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

# Plot decision boundary and margins

Z = Z.reshape(xx.shape)

plt.contour(xx, yy, Z, colors='k', levels=[-1, 0, 1], alpha=0.5, linestyles=['--', '-', '--'])

# Highlight the support vectors

plt.scatter(svm\_classifier.support\_vectors\_[:, 0], svm\_classifier.support\_vectors\_[:, 1], s=200, facecolors='none', edgecolors='k')

plt.title('SVM Decision Boundary')

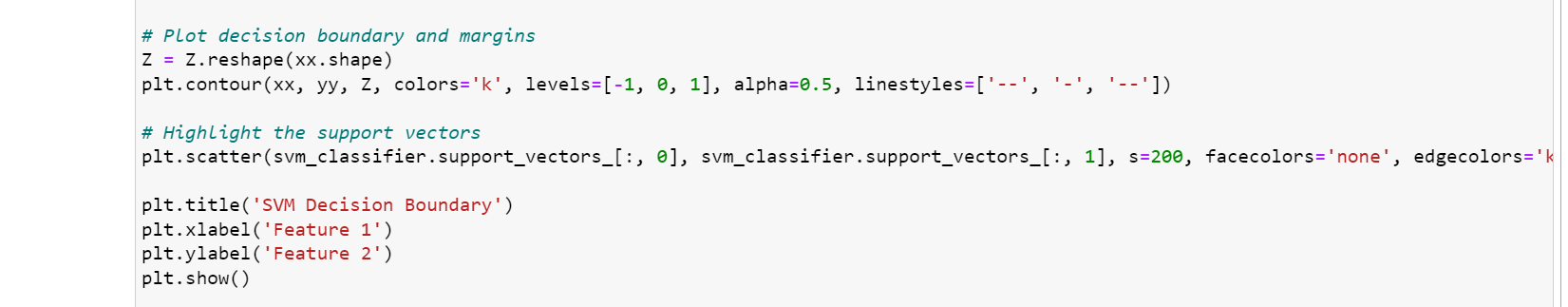
plt.xlabel('Feature 1')

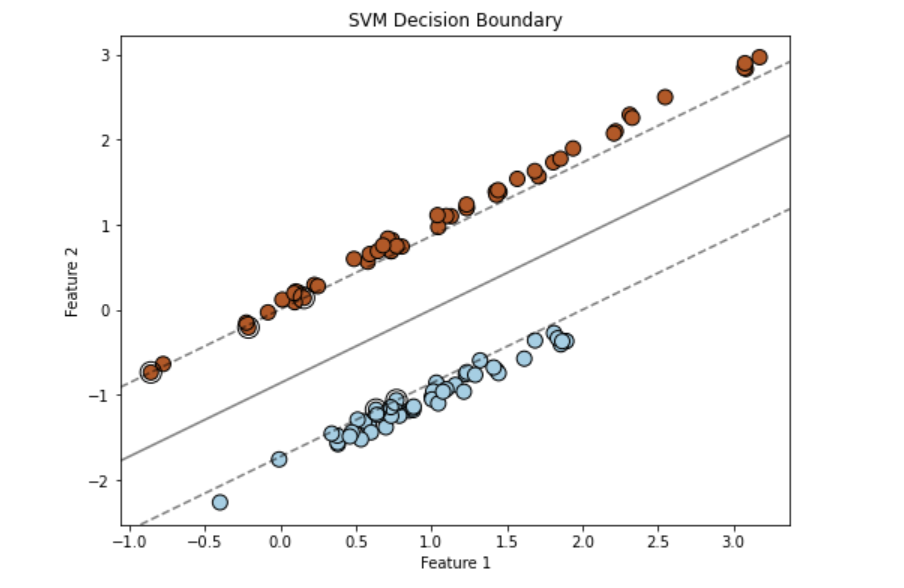
plt.ylabel('Feature 2')

plt.show()

**Output:**







**GitHub Link: https://github.com/chethan1n1/machine-learning**