RIPHAH
INTERNATIONAL
UNIVERSITY

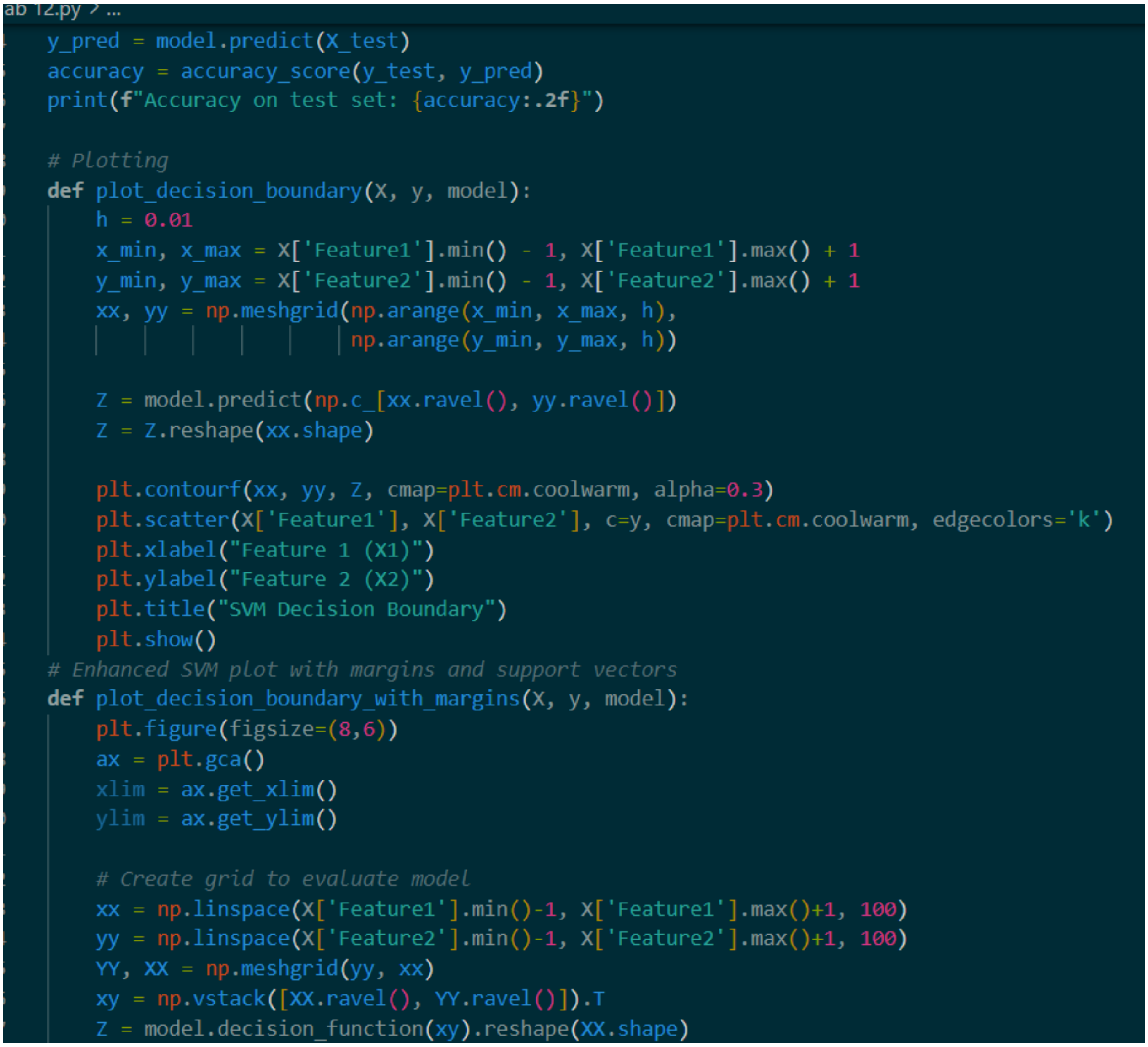
Artificial Intelligence Lab 12

Tasks Name: AYIZA WAQAR

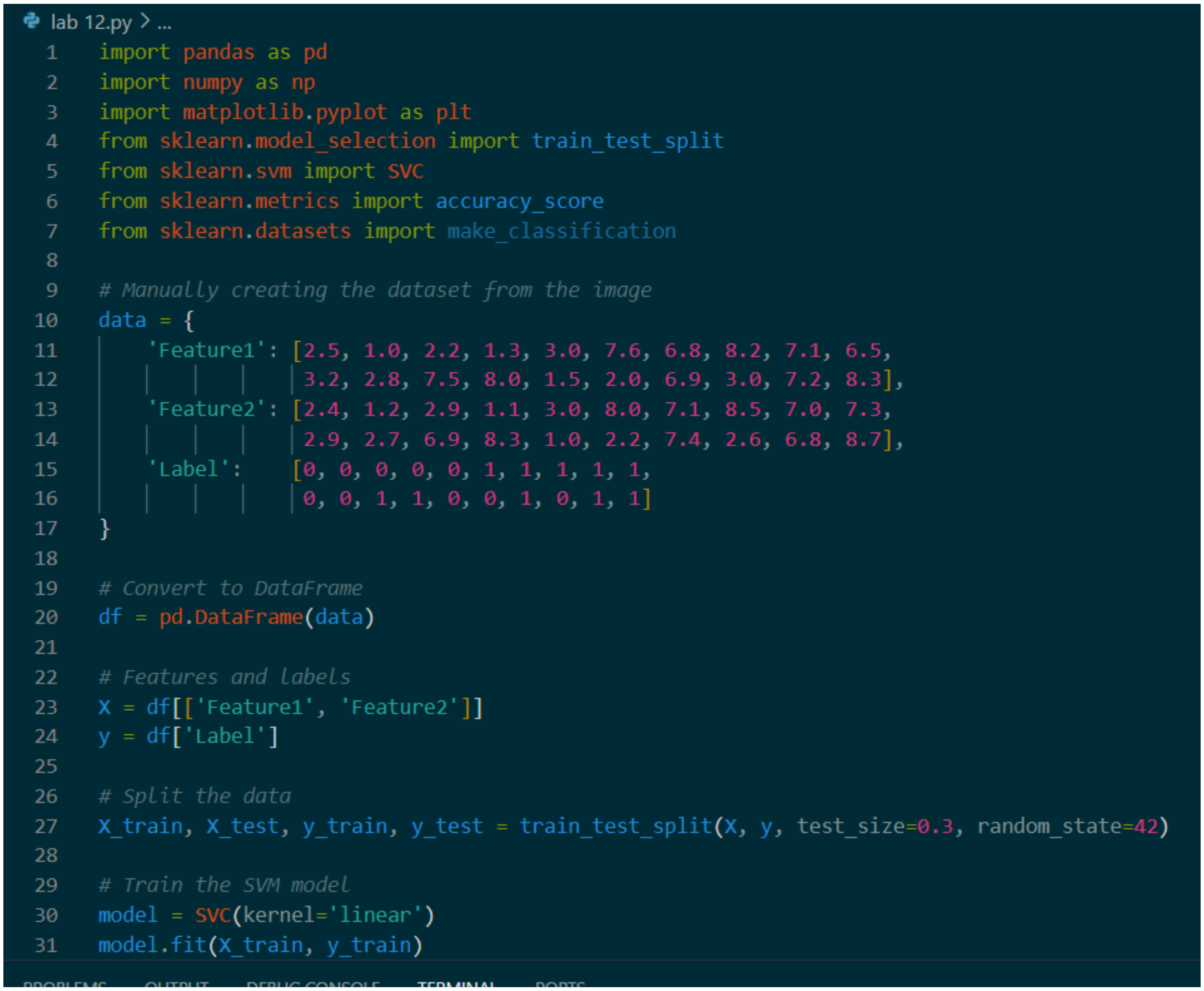
SapID: 44529

Batch: BSCS-6 th semester

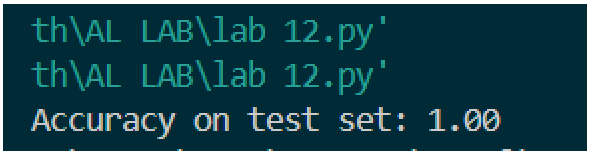
Lab Instructor: Mam Ayesha Akram



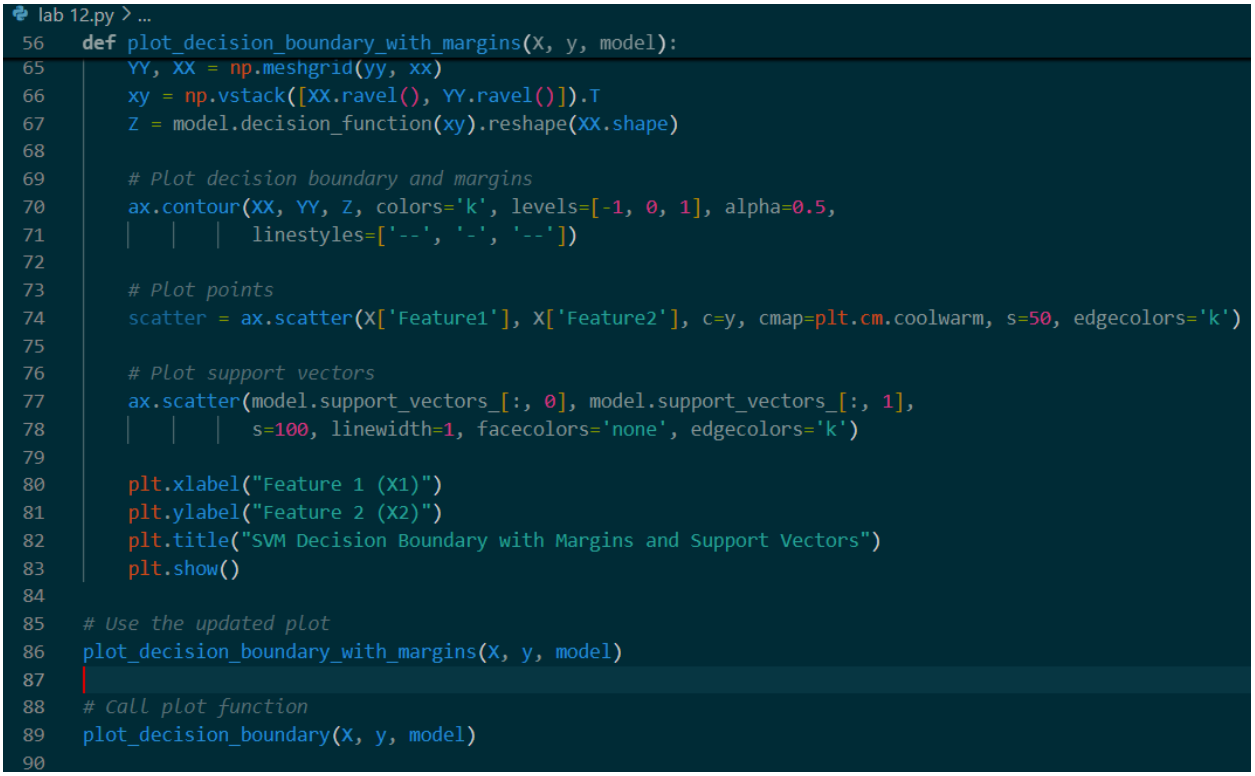
```python
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy on test set: {accuracy:.2f}")

# Plotting
def plot_decision_boundary(X, y, model):
    h = 0.01
    x_min, x_max = X['Feature1'].min() - 1, X['Feature1'].max() + 1
    y_min, y_max = X['Feature2'].min() - 1, X['Feature2'].max() + 1
    xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
                         np.arange(y_min, y_max, h))

    Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)

    plt.contourf(xx, yy, Z, cmap=plt.cm.coolwarm, alpha=0.3)
    plt.scatter(X['Feature1'], X['Feature2'], c=y, cmap=plt.cm.coolwarm, edgecolors='k')
    plt.xlabel("Feature 1 (X1)")
    plt.ylabel("Feature 2 (X2)")
    plt.title("SVM Decision Boundary")
    plt.show()
# Enhanced SVM plot with margins and support vectors
def plot_decision_boundary_with_margins(X, y, model):
    plt.figure(figsize=(8,6))
    ax = plt.gca()
    xlim = ax.get_xlim()
    ylim = ax.get_ylim()

    # Create grid to evaluate model
    xx = np.linspace(X['Feature1'].min()-1, X['Feature1'].max()+1, 100)
    yy = np.linspace(X['Feature2'].min()-1, X['Feature2'].max()+1, 100)
    YY, XX = np.meshgrid(yy, xx)
    xy = np.vstack([XX.ravel(), YY.ravel()]).T
    Z = model.decision_function(xy).reshape(XX.shape)
```



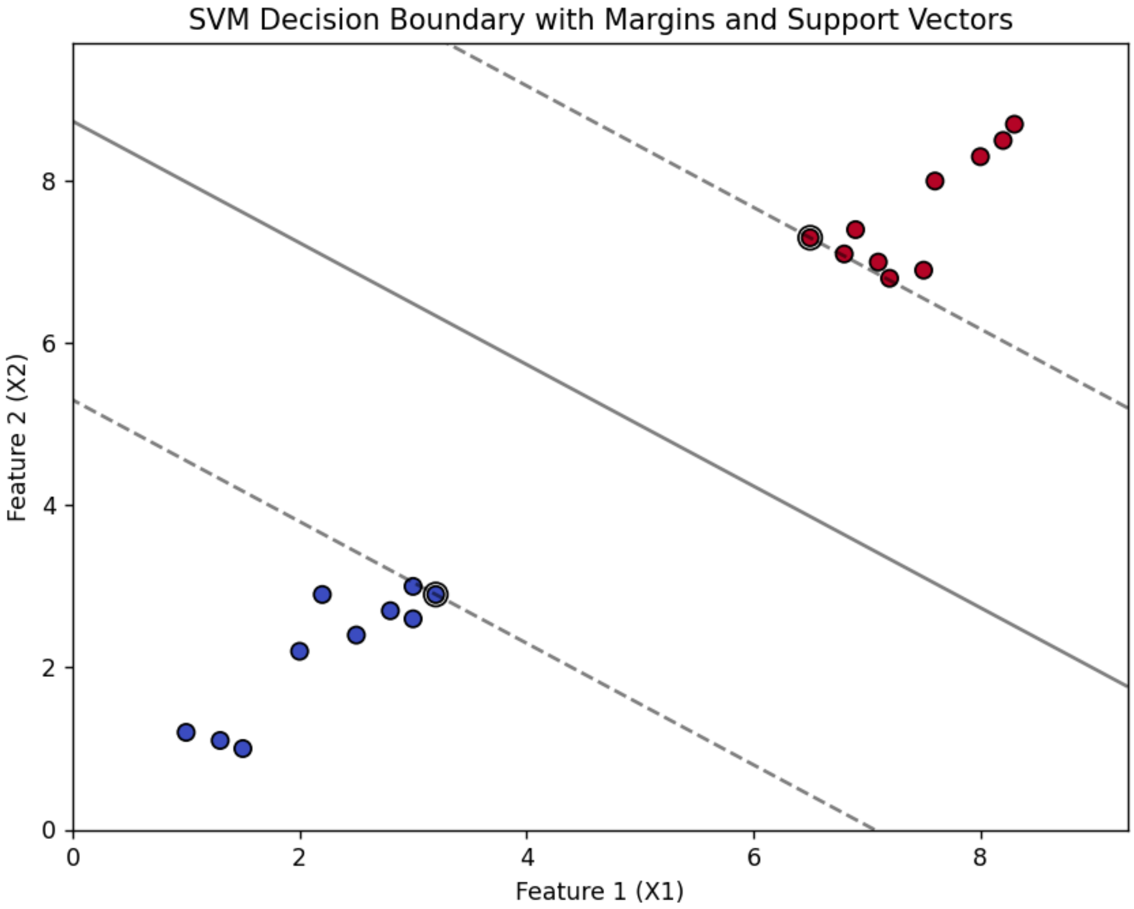
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from sklearn.datasets import make_classification

# Manually creating the dataset from the image
data = {
    'Feature1': [2.5, 1.0, 2.2, 1.3, 3.0, 7.6, 6.8, 8.2, 7.1, 6.5,
                 3.2, 2.8, 7.5, 8.0, 1.5, 2.0, 6.9, 3.0, 7.2, 8.3],
    'Feature2': [2.4, 1.2, 2.9, 1.1, 3.0, 8.0, 7.1, 8.5, 7.0, 7.3,
                 2.9, 2.7, 6.9, 8.3, 1.0, 2.2, 7.4, 2.6, 6.8, 8.7],
    'Label':    [0, 0, 0, 0, 0, 1, 1, 1, 1, 1,
                 0, 0, 1, 1, 0, 0, 1, 0, 1, 1]
}

# Convert to DataFrame
df = pd.DataFrame(data)

# Features and labels
X = df[['Feature1', 'Feature2']]
y = df['Label']

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Train the SVM model
model = SVC(kernel='linear')
model.fit(X_train, y_train)
```



```
th\AL LAB\lab 12.py'
th\AL LAB\lab 12.py'
Accuracy on test set: 1.00
```



```
def plot_decision_boundary_with_margins(X, y, model):
    YY, XX = np.meshgrid(yy, xx)
    xy = np.vstack([XX.ravel(), YY.ravel()]).T
    Z = model.decision_function(xy).reshape(XX.shape)

    # Plot decision boundary and margins
    ax.contour(XX, YY, Z, colors='k', levels=[-1, 0, 1], alpha=0.5,
               linestyles=['--', '-', '--'])

    # Plot points
    scatter = ax.scatter(X['Feature1'], X['Feature2'], c=y, cmap=plt.cm.coolwarm, s=50, edgecolors='k')

    # Plot support vectors
    ax.scatter(model.support_vectors_[:, 0], model.support_vectors_[:, 1],
               s=100, linewidth=1, facecolors='none', edgecolors='k')

    plt.xlabel("Feature 1 (X1)")
    plt.ylabel("Feature 2 (X2)")
    plt.title("SVM Decision Boundary with Margins and Support Vectors")
    plt.show()

# Use the updated plot
plot_decision_boundary_with_margins(X, y, model)

# Call plot function
plot_decision_boundary(X, y, model)
```


SVM Decision Boundary with Margins and Support Vectors
Feature 2 (X2)
Feature 1 (X1)