RIPHAH INTERNATIONAL UNIVERSITY, ISLAMABAD

Lab#12

Bachelors of Computer Science – 6th Semester

Course: Artificial Intelligence

Submitted to: Ms. Ayesha

Submitted by: Tabinda Hassan

SAP-46374

Date of Submission: 29-04-2025

Lab Tasks

Question 01:

SVM Algorithm:

ID	Feature 1 (X1)	Feature 2 (X2)	Label (Y)
1	2.5	2.4	0
2	1.0	1.2	0
3	2.2	2.9	0
4	1.3	1.1	0
5	3.0	3.0	0
6	7.6	8.0	1
7	6.8	7.1	1
8	8.2	8.5	1
9	7.1	6.5	1
10	6.5	7.0	1
11	3.2	2.9	0
12	2.8	2.7	0
13	7.5	6.9	1
14	8.0	8.3	1
15	1.5	1.0	0
16	2.0	2.2	0
17	6.9	7.4	1
18	7.2	6.8	1
19	3.0	2.6	0
20	8.3	8.7	1

- Load the dataset into Python from a .csv file.
- Split it into training and testing sets (70% train, 30% test).
- Train an SVM model using kernel='linear'.
- Print the accuracy on the test set.
- Plot the data and the decision boundary.

```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

# Load data (adjust path if needed)
df = pd.read_csv('SVM.csv')

# Print columns to confirm names (debugging tip)
print("Column names:", df.columns)

# Ensure correct column names
# You may need to adjust these if your CSV uses 'X1', 'X2', 'Y' etc.
X = df[['Feature 1 (X1)', 'Feature 2 (X2)']].values
y = df['Label (Y)'].values

# Train-test split (70% train, 30% test)
X_train, X_test, y_train, y_test = train_test_split( *arrays: X, y, test_size=0.3, random_state=42)

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

# Predict and calculate accuracy
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Test Accuracy: {accuracy * 100:.2f}%")
```

```python

# Plotting decision boundary
def plot_decision_boundary(X, y, model):  1 usage
    plt.figure(figsize=(8, 6))
    plt.scatter(X[:, 0], X[:, 1], c=y, cmap='bwr', s=30, edgecolors='k')

    # Create mesh grid
    ax = plt.gca()
    xlim = ax.get_xlim()
    ylim = ax.get_ylim()
    xx, yy = np.meshgrid( *xi: np.linspace(xlim[0], xlim[1], num: 100),
                          np.linspace(ylim[0], ylim[1], num: 100))
    xy = np.vstack([xx.ravel(), yy.ravel()]).T
    Z = model.decision_function(xy).reshape(xx.shape)

    # Plot decision boundary and margins
    plt.contour( *args: xx, yy, Z, colors='k', levels=[0], alpha=0.5, linestyles=['-'])
    plt.contour( *args: xx, yy, Z, colors='grey', levels=[-1, 1], alpha=0.3, linestyles=['--'])

    plt.xlabel("Feature 1 (X1)")
    plt.ylabel("Feature 2 (X2)")
    plt.title("SVM Decision Boundary")
    plt.show()

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