Assignment

1. SVM Classifier

Consider the following instances

* x1 = 10, x2 = 4, x3 = 7
* x1 = 9, x2 = 5, x3 = 8
* x1 = 3, x2 = 2, x3 = 2
* x1 = 6, x2 = 3, x3 = 4
* Write the decision boundary/hyperplane and Calculate the decision boundary function using SVM Classifier with given weights and bias w1 = 1 w2 = -0.5,w3 = 0.6,bias = -8 for each instance and predict class.
* Find the Margin value for above features.
* What is the Objective function SVM with respect to subject constraints.
* Analyze constraints with subject to SVM classifier.
* Summarize types of SVM kernels and write the kernel function for each type
* Role of Support vectors in SVM

1. Compare Hyperplane, Margin and support vectors
2. Applications, strengths and weakness of SVM
3. Run the following code and Write the observed points (at least 5) and output from the code

# Import necessary libraries

import numpy as np

import matplotlib.pyplot as plt

from sklearn import datasets

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

from sklearn.svm import SVC

from sklearn.metrics import accuracy\_score

# Generate a synthetic dataset

X, y = datasets.make\_classification(n\_samples=100, n\_features=2, n\_informative=2, n\_redundant=0, random\_state=42)

# Split the dataset into training and testing sets

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

# Create an SVM classifier with a linear kernel

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

# Train the SVM classifier on the training data

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

# Extract the weight vector and bias

w = svm\_classifier.coef\_[0]

b = svm\_classifier.intercept\_[0]

# Extract support vectors

support\_vectors = svm\_classifier.support\_vectors\_

# Calculate the margin

margin = 2 / np.linalg.norm(w)

# Make predictions on the test data

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

# Calculate the accuracy of the classifier

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

print("Accuracy:", accuracy)

# Visualize the decision boundary and support vectors

plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Paired)

plt.scatter(support\_vectors[:, 0], support\_vectors[:, 1], s=100, linewidth=1, facecolors='none', edgecolors='k')

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], 50), np.linspace(ylim[0], ylim[1], 50))

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=['--', '-', '--'])

plt.title("SVM Decision Boundary with Support Vectors")

plt.xlabel("Feature 1")

plt.ylabel("Feature 2")

plt.show()

# Print the extracted parameters

print("Weight vector (w):", w)

print("Bias (b):", b)

print("Support Vectors:")

print(support\_vectors)

print("Margin:", margin)