MANJARA CHARITABLE TRUST

RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI

DEPARTMENT OF COMPUTER ENGINEERING

Academic Year: 2023-2024 **Semester: VII**

Student Name:___________________________________ Roll No:

Subject: **Machine Learning** Class/Div: **BE/A**

Experiment No.:_____

Title of Experiment:

Experiment Evaluation:

Experiment No.	Performance/ Coding (04)	Punctuality (03)	Question-Answer (03)	Total (10)
3				

Staff Signature

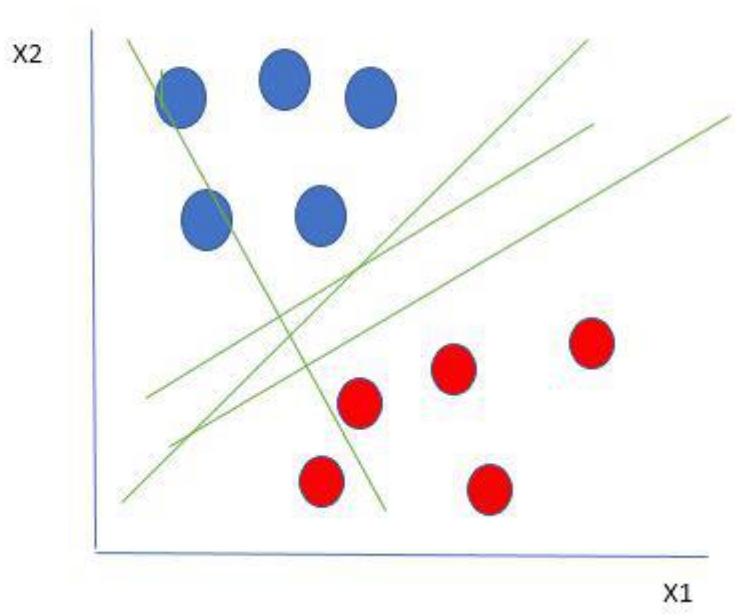
Experiment No.7

Aim:- To implement SVM

Theory:-

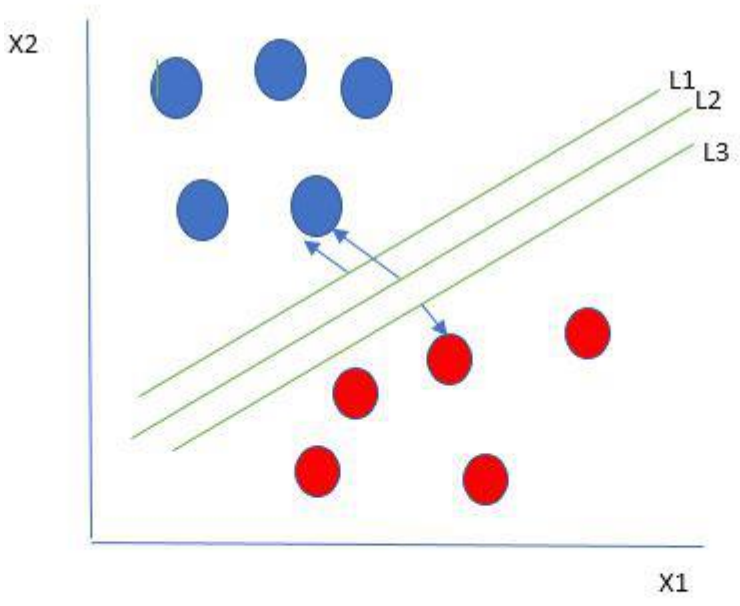
Support Vector Machine (SVM) is a supervised machine learning algorithm used for both classification and regression. Though we say regression problems as well it's best suited for classification. The main objective of the SVM algorithm is to find the optimal hyperplane in an N-dimensional space that can separate the data points in different classes in the feature space. The hyperplane tries that the margin between the closest points of different classes should be as maximum as possible. The dimension of the hyperplane depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a 2-D plane. It becomes difficult to imagine when the number of features exceeds three.

Let's consider two independent variables x1, x2, and one dependent variable which is either a blue circle or a red circle.

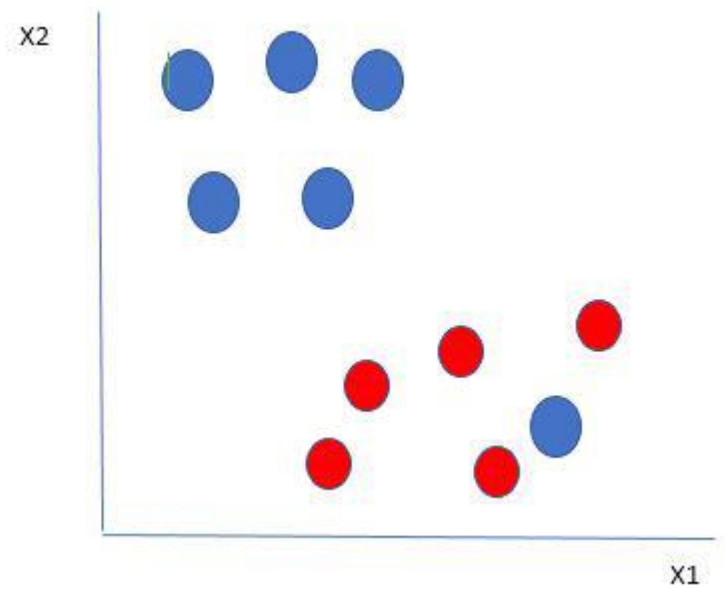


How does SVM work?

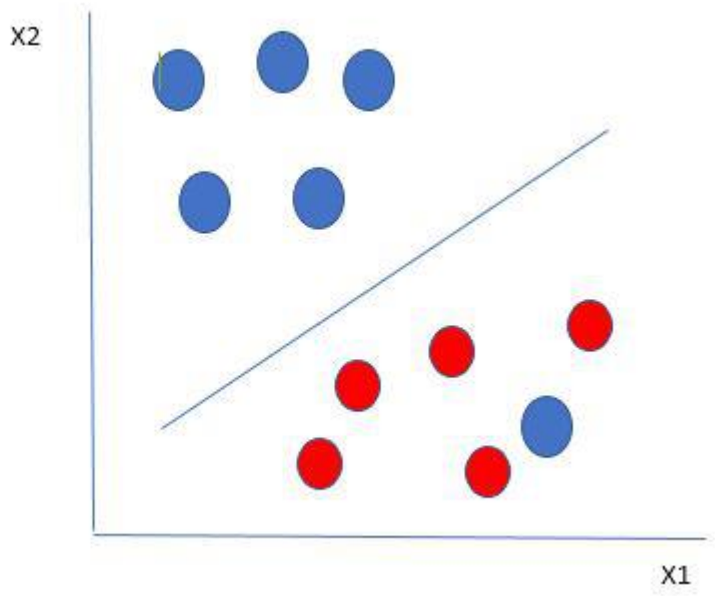
One reasonable choice as the best hyperplane is the one that represents the largest separation or margin between the two classes.



So we choose the hyperplane whose distance from it to the nearest data point on each side is maximized. If such a hyperplane exists it is known as the maximum-margin hyperplane/hard margin. So from the above figure, we choose L2. Let’s consider a scenario like shown below

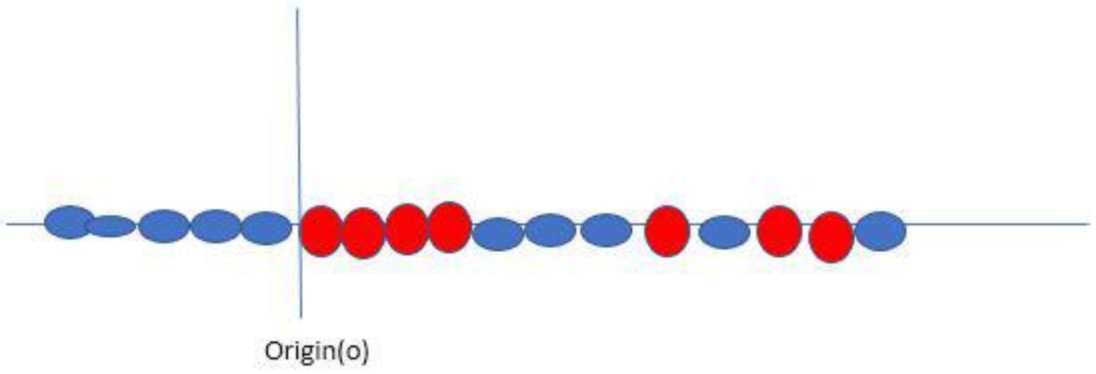


Here we have one blue ball in the boundary of the red ball. So how does SVM classify the data? It’s simple! The blue ball in the boundary of red ones is an outlier of blue balls. The SVM algorithm has the characteristics to ignore the outlier and finds the best hyperplane that maximizes the margin. SVM is robust to outliers.

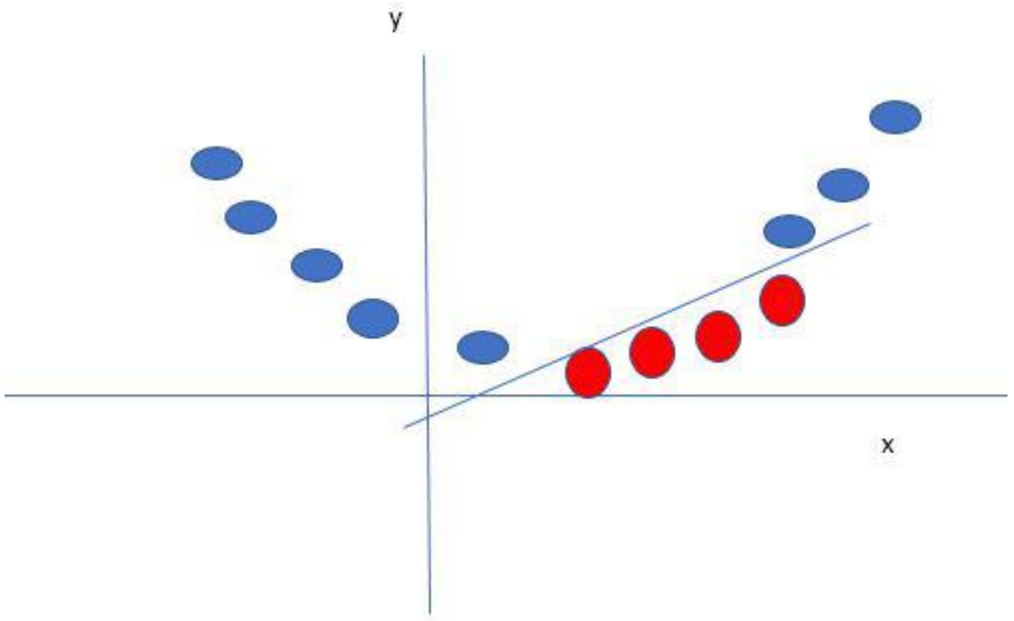


So in this type of data point what SVM does is, finds the maximum margin as done with previous data sets along with that it adds a penalty each time a point crosses the margin. So the margins in these types of cases are called soft margins. When there is a soft margin to the data set, the SVM tries to minimize (1/margin+∧(∑penalty)). Hinge loss is a commonly used penalty. If no violations no hinge loss.If violations hinge loss proportional to the distance of violation.

Till now, we were talking about linearly separable data(the group of blue balls and red balls are separable by a straight line/linear line). What to do if data are not linearly separable?



Say, our data is shown in the figure above. SVM solves this by creating a new variable using a kernel. We call a point xi on the line and we create a new variable yi as a function of distance from origin o.so if we plot this we get something like as shown below



In this case, the new variable y is created as a function of distance from the origin. A non-linear function that creates a new variable is referred to as a kernel.

Support Vector Machine Terminology

Hyperplane: Hyperplane is the decision boundary that is used to separate the data points of different classes in a feature space. In the case of linear classifications, it will be a linear equation i.e. wx+b = 0.

Support Vectors: Support vectors are the closest data points to the hyperplane, which makes a critical role in deciding the hyperplane and margin.

Margin: Margin is the distance between the support vector and hyperplane. The main objective of the support vector machine algorithm is to maximize the margin. The wider margin indicates better classification performance.

Kernel: Kernel is the mathematical function, which is used in SVM to map the original input data points into high-dimensional feature spaces, so, that the hyperplane can be easily found out even if the data points are not linearly separable in the original input space. Some of the common kernel functions are linear, polynomial, radial basis function(RBF), and sigmoid.

Hard Margin: The maximum-margin hyperplane or the hard margin hyperplane is a hyperplane that properly separates the data points of different categories without any misclassifications.

Soft Margin: When the data is not perfectly separable or contains outliers, SVM permits a soft margin technique. Each data point has a slack variable introduced by the soft-margin SVM formulation, which softens the strict margin requirement and permits certain misclassifications or violations. It discovers a compromise between increasing the margin and reducing violations.

C: Margin maximisation and misclassification fines are balanced by the regularisation parameter C in SVM. The penalty for going over the margin or misclassifying data items is decided by it. A stricter penalty is imposed with a greater value of C, which results in a smaller margin and perhaps fewer misclassifications.

Hinge Loss: A typical loss function in SVMs is hinge loss. It punishes incorrect classifications or margin violations. The objective function in SVM is frequently formed by combining it with the regularisation term.

Dual Problem: A dual Problem of the optimisation problem that requires locating the Lagrange multipliers related to the support vectors can be used to solve SVM. The dual formulation enables the use of kernel tricks and more effective computing.

Mathematical intuition of Support Vector Machine

Consider a binary classification problem with two classes, labeled as +1 and -1. We have a training dataset consisting of input feature vectors X and their corresponding class labels Y.

The equation for the linear hyperplane can be written as:

$$w^T\text{x} + \text{b} = 0$$

The vector W represents the normal vector to the hyperplane. i.e the direction perpendicular to the hyperplane. The parameter b in the equation represents the offset or distance of the hyperplane from the origin along the normal vector w.

The distance between a data point x_i and the decision boundary can be calculated as:

$$d_i = \frac{w^T x_i + b}{||w||}$$

where ||w|| represents the Euclidean norm of the weight vector w. Euclidean norm of the normal vector W

For Linear SVM classifier :

$$\hat{y} = \begin{cases} 1 & : \ w^T x + b \geq 0 \\ 0 & : \ w^T x + b < 0 \end{cases}$$

Optimization:

For Hard margin linear SVM classifier:

$$\underset{w,b}{\text{minimize}}\frac{1}{2}w^Tw = \underset{W,b}{\text{minimize}}\frac{1}{2}\|w\|^2$$
$$\text{subject to } y_i(w^Tx_i + b) \geq 1 \; for \; i = 1, 2, 3, \cdots, m$$

The target variable or label for the ith training instance is denoted by the symbol ti in this statement. And ti=-1 for negative occurrences (when yi= 0) and ti=1positive instances (when yi = 1) respectively. Because we require the decision boundary that satisfy the constraint: t_i(w^T x_i + b)\geq1

For Soft margin linear SVM classifier:

$$\underset{w,b}{\text{minimize}} \; \frac{1}{2}w^Tw + C\sum_{i=1}^{m}\zeta_i$$
$$\text{subject to } y_i(w^Tx_i + b) \geq 1 - \zeta_i \;\; and \; \zeta_i \geq 0 \;\; for \; i = 1, 2, 3, \cdots, m$$

Dual Problem: A dual Problem of the optimisation problem that requires locating the Lagrange multipliers related to the support vectors can be used to solve SVM. The optimal Lagrange multipliers α(i) that maximize the following dual objective function

$$\underset{\alpha}{\text{maximize}} : \frac{1}{2}\sum_{i\to m}\;\sum_{j\to m}\alpha_i\alpha_j t_i t_j K(x_i, x_j) - \sum_{i\to m}\alpha_i$$

where,

αi is the Lagrange multiplier associated with the ith training sample.

K(xi, xj) is the kernel function that computes the similarity between two samples xi and xj. It allows SVM to handle nonlinear classification problems by implicitly mapping the samples into a higher-dimensional feature space.

The term ∑αi represents the sum of all Lagrange multipliers.

The SVM decision boundary can be described in terms of these optimal Lagrange multipliers and the support vectors once the dual issue has been solved and the optimal Lagrange multipliers have been discovered. The training samples that have i > 0 are the support vectors, while the decision boundary is supplied by:

$$w = \sum_{i \to m} \alpha_i t_i K(x_i, x) + b$$
$$t_i(w^T x_i - b) = 1 \Longleftrightarrow b = w^T x_i - t_i$$

Types of Support Vector Machine

Based on the nature of the decision boundary, Support Vector Machines (SVM) can be divided into two main parts:

Linear SVM: Linear SVMs use a linear decision boundary to separate the data points of different classes. When the data can be precisely linearly separated, linear SVMs are very suitable. This means that a single straight line (in 2D) or a hyperplane (in higher dimensions) can entirely divide the data points into their respective classes. A hyperplane that maximizes the margin between the classes is the decision boundary.

Non-Linear SVM: Non-Linear SVM can be used to classify data when it cannot be separated into two classes by a straight line (in the case of 2D). By using kernel functions, nonlinear SVMs can handle nonlinearly separable data. The original input data is transformed by these kernel functions into a higher-dimensional feature space, where the data points can be linearly separated. A linear SVM is used to locate a nonlinear decision boundary in this modified space.

Popular kernel functions in SVM

The SVM kernel is a function that takes low-dimensional input space and transforms it into higher-dimensional space, ie it converts nonseparable problems to separable problems. It is mostly useful in non-linear separation problems. Simply put the kernel, does some extremely complex data transformations and then finds out the process to separate the data based on the labels or outputs defined.

$$\text{Linear}: \; K(w,b) = w^T x + b$$
$$\text{Polynomial}: \; K(w,x) = (\gamma w^T x + b)^N$$
$$\text{Gaussian RBF}: \; K(w,x) = \exp(-\gamma ||x_i - x_j||^n$$
$$\text{Sigmoid}: K(x_i, x_j) = \tanh(\alpha x_i^T x_j + b)$$

Advantages of SVM

Effective in high-dimensional cases.

Its memory is efficient as it uses a subset of training points in the decision function called support vectors.

Different kernel functions can be specified for the decision functions and its possible to specify custom kernels.

SVM implementation in Python

Predict if cancer is Benign or malignant. Using historical data about patients diagnosed with cancer enables doctors to differentiate malignant cases and benign ones are given independent attributes.

Steps

- Load the breast cancer dataset from sklearn.datasets
- Separate input features and target variables.
- Buil and train the SVM classifiers using RBF kernel.
- Plot the scatter plot of the input features.
- Plot the decision boundary.
- Plot the decision boundary

Program:-

```
# Import necessary libraries

import numpy as np

import matplotlib.pyplot as plt

from sklearn import datasets
from sklearn import svm

from sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy_score


# Load the iris dataset (you can use your own dataset)

iris = datasets.load_iris()

X = iris.data[:, :2]  # We'll use only the first two features for simplicity

y = iris.target


# Split the data into training and testing sets

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)


# Create an SVM classifier

clf = svm.SVC(kernel='linear', C=1)  # Linear kernel


# Train the classifier on the training data

clf.fit(X_train, y_train)


# Make predictions on the test data

y_pred = clf.predict(X_test)


# Calculate the accuracy of the classifier

accuracy = accuracy_score(y_test, y_pred)

print(f'Accuracy: {accuracy * 100:.2f}%')
```

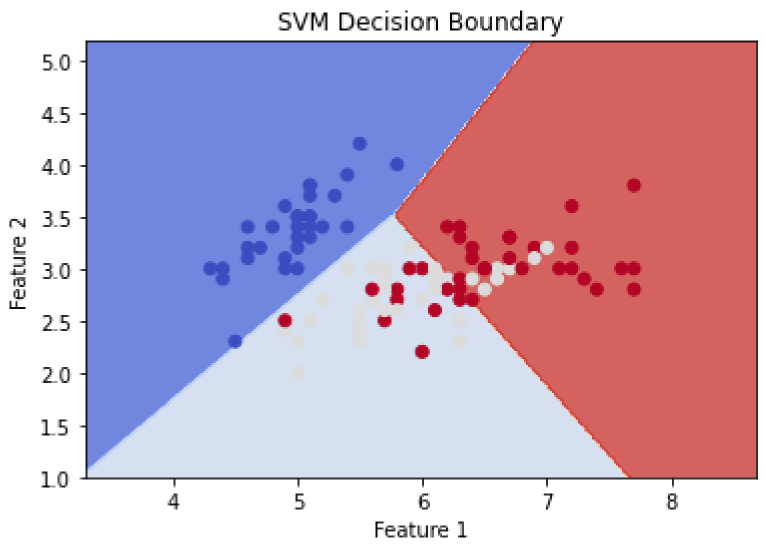
```
# Visualize the decision boundary
def plot_decision_boundary(X, y, classifier):
    h = 0.02  # Step size in the mesh
    x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
    y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
    xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min,
y_max, h))
    Z = classifier.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)

    plt.contourf(xx, yy, Z, cmap=plt.cm.coolwarm, alpha=0.8)
    plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.coolwarm)
    plt.xlabel('Feature 1')
    plt.ylabel('Feature 2')
    plt.title('SVM Decision Boundary')
    plt.show()

# Plot the decision boundary
plot_decision_boundary(X_train, y_train, clf)
```

Output:-

```
Accuracy: 80.00%
```


SVM Decision Boundary
Feature 2
Feature 1