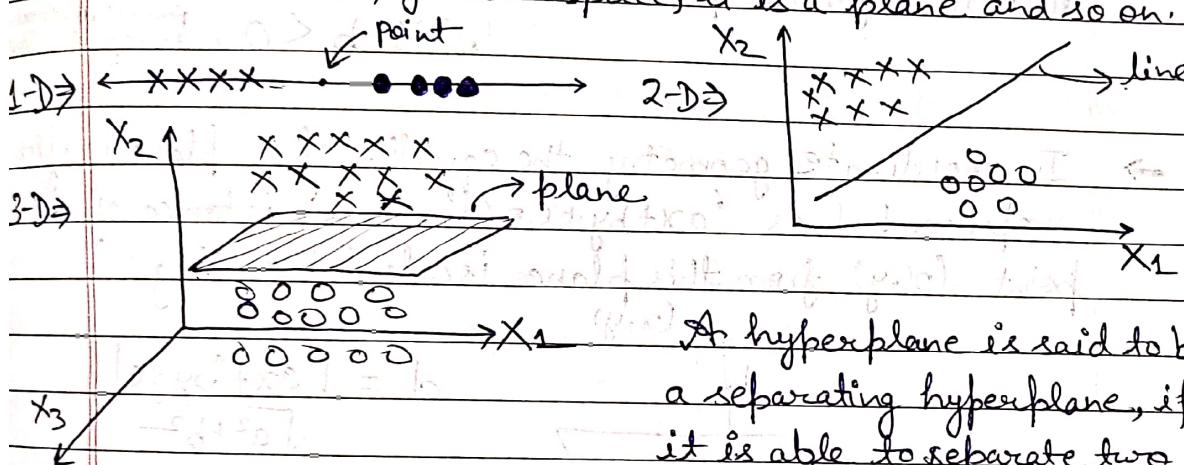


Support Vector Machine (SVM)

Support Vector Machines (SVMs) are powerful supervised learning models used primarily for classification, but can also be used for regression tasks. They are effective in high-dimensional spaces and versatile in handling both linear and non-linear data through the use of kernel functions.

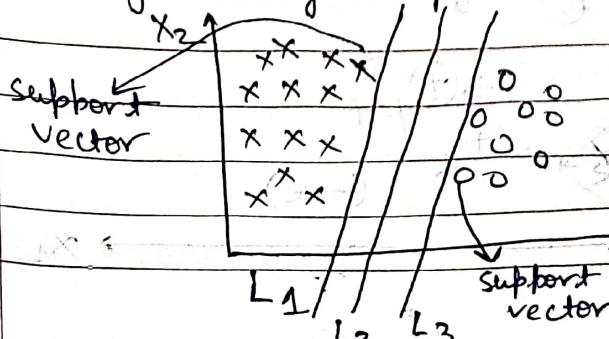
- **Hyperplane:** A hyperplane is a plane of ' $n-1$ ' dimensions in a ' n ' dimensional feature space, that separates two classes. For a 1-D space, it is a point, for 2-D, it is a line, for 3-D space, it is a plane and so on.



A hyperplane is said to be a separating hyperplane, if it is able to separate two classes. The main goal of SVM is to find an optimal hyperplane that best separates data points of different classes.

- **Support Vectors:** These are the data points that are closest to the hyperplane and are critical in defining the position and orientation of the hyperplane. The SVM tries to maximize the margin between these support vectors and the hyperplane.

- **Margin:** The margin is the distance between the hyperplane and the nearest data points from both the classes. A larger margin implies better generalization on data.



Out of three separating hyperplanes, L_1 , L_2 , L_3 , L_3 is the best hyperplane because from that distance to support vectors is maximum.

(MV2) Machine Learning & Deep learning

In 2-D, the equation of a line (hyperplane) is given as $W^T X + b = 0$, where W is the weight vector (in 2D, it has two components, $W = [w_1, w_2]$), X is the input vector (in 2D, $X = [x_1, x_2]$), b is the bias term/intercept.

In a binary classification problem, the goal is to separate two classes ($y = +1$ and $y = -1$), the decision boundary should satisfy the condition:

$$W^T X + b \geq 0 \quad \text{for all points belonging to positive class}$$

such that for all points:

$$W^T X + b > 0, \quad W^T X + b < 0, \quad \text{for all points belonging to negative class.}$$

→ In coordinate geometry, the equation of a plane is often represented as ' $ax+by+c=0$ ', and distance of a point (x,y) from this plane is calculated using:

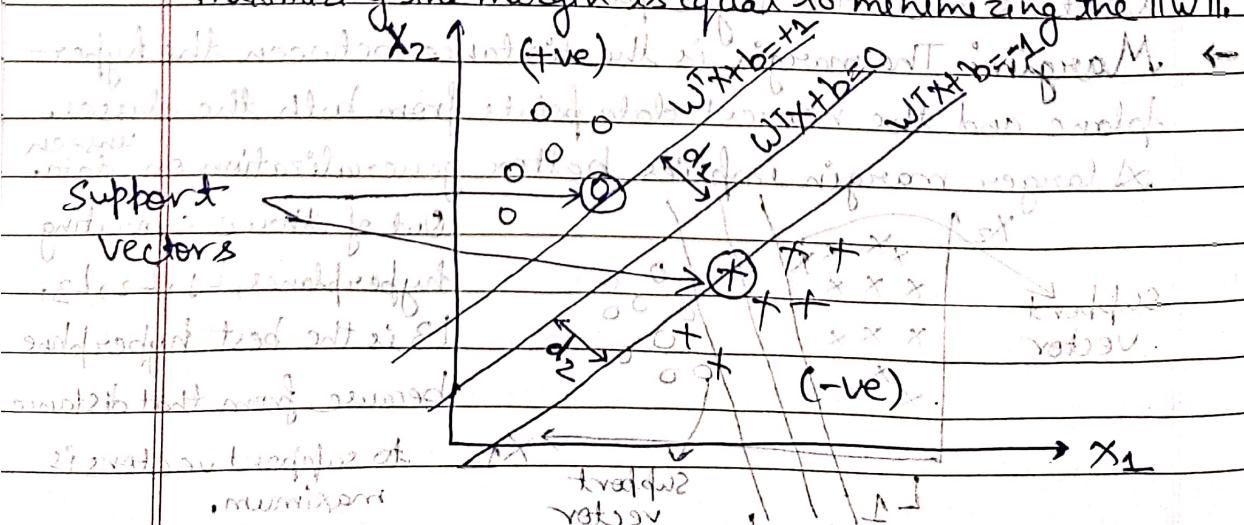
$$d = \frac{|ax+by+c|}{\sqrt{a^2+b^2}}$$

In our case, the distance can be represented as

$$d = |W^T X + b|$$

$\rightarrow \|W\|_2 \rightarrow L2 \text{ Norm}$

Our goal is to maximize the distance between the plane and the support vectors. The distance/margin is inversely proportional to norm of the weight vector $\|W\|$. Therefore, maximizing the margin is equal to minimizing the $\|W\|$.



$$d_1 = \frac{|\mathbf{w}^T \mathbf{x}_{\text{pos}} + b|}{\|\mathbf{w}\|} = \frac{1}{\|\mathbf{w}\|}, \text{ for positive support vector}$$

$$d_2 = \frac{|\mathbf{w}^T \mathbf{x}_{\text{(neg)}} + b|}{\|\mathbf{w}\|} = \frac{|-1|}{\|\mathbf{w}\|} = \frac{1}{\|\mathbf{w}\|}, \text{ for negative support vector.}$$

$$\text{distance } d = d_1 + d_2 = \frac{2}{\|\mathbf{w}\|}$$

$\|\mathbf{w}\| \rightarrow \text{L2 Norm.}$

Thus, optimization problem can be formulated as -

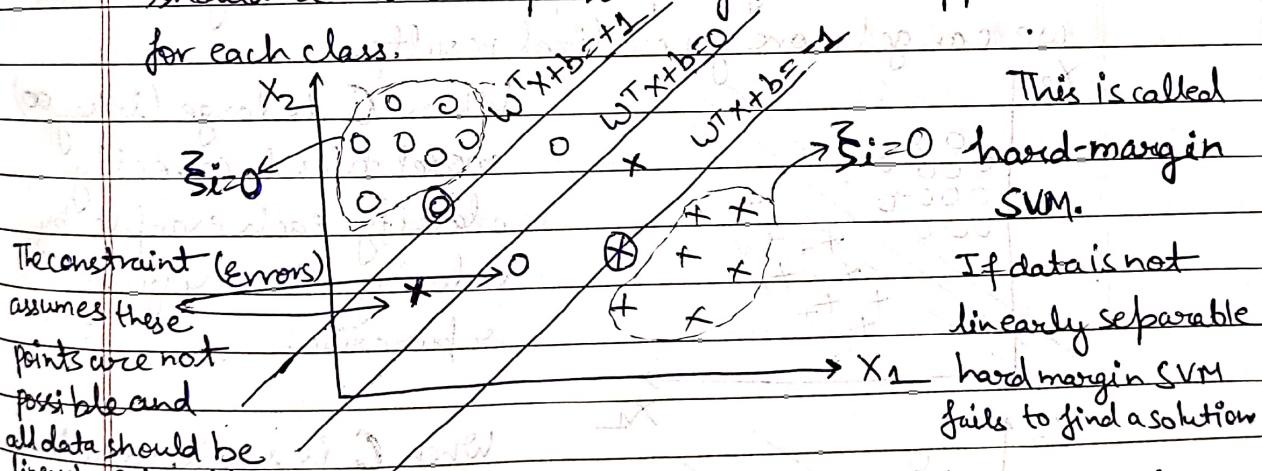
final sum objective

$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|_2^2, \text{ subject to the constraint } y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1 \quad \forall i$$

Here, y_i is the label of the data point \mathbf{x}_i (either +1 or -1).

However, this constrained optimization problem is not straight forward, and directly solving it can lead to a non-convex function because of the hard margin constraints.

The constraints we have defined assumes that there should be no data points beyond the support vectors for each class.



But in real-world data may not be that simple and such cases can happen because of presence of outliers or noise.

Soft Margin SVM allows for some misclassification by allowing/introducing slack variable ξ_i (zeta) that measures the degree to which a data point violates the margin constraint.

* The goal of soft margin SVM is to find a balance between maximizing the margin and minimizing the classification error (misclassifications). This is controlled by a C-parameter, which determines the trade-off between having a larger

margin and allowing more misclassifications.

For these misclassifications/error points, we want to add some penalty to our SVM objective which is given as -

Hinge loss

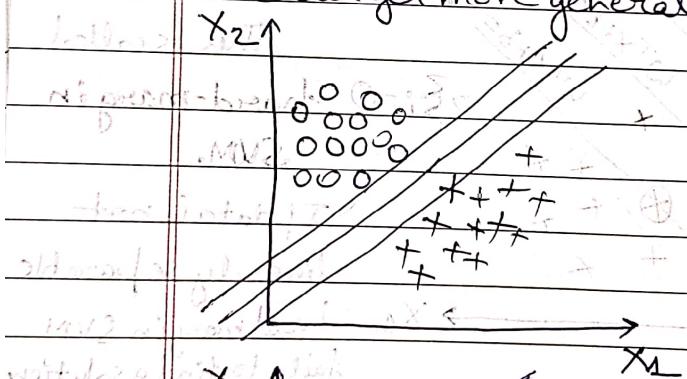
Primal
objective

$$\min_{W, b} \frac{1}{2} \|W\|_2^2 + C \sum_{i=1}^n \xi_i \quad \text{subject to } y_i (W^T X_i + b) \geq 1 - \xi_i \quad \xi_i \geq 0$$

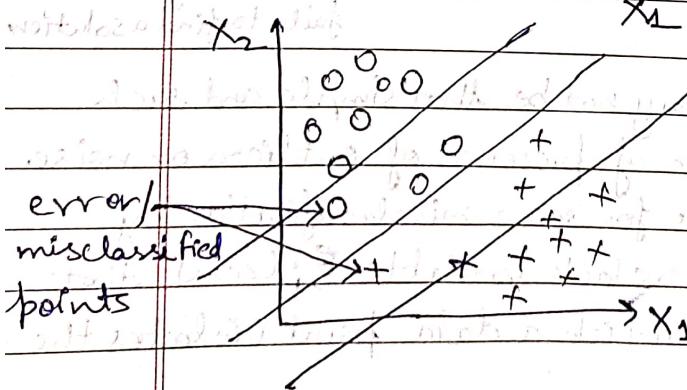
- Here, $\xi_i \geq 0$ are the slack variables that allow for some margin violations.

C is a regularization parameter that controls the penalty for misclassifications:

- A larger C puts more emphasis on minimizing misclassifications (i.e., fewer misclassified points but possibly a small margin) which can be prone to overfitting.
- A smaller C allows for a larger margin but potentially more misclassified points. At the cost of some error, we can get more generalized results.



When C is large (let's say ∞) model tries to correctly classify each example by reducing the margin of separation.



When C is small, we allow some misclassifications, thus margin of separation can be maximized helping us to get more generalized result.

Our primal objective function has some constraints so we cannot directly use it for finding the global minimum by reducing the distance of margin.

Instead we need to convert constrained optimization problem into an unconstrained optimization problem.

We use the concept of Lagrange's Multiplier using which we can optimize our objective without thinking about the constraints.

- Using Lagrange's Multiplier, our primal objective can be written as -

$$L = \frac{1}{2} \|w\|^2 - \sum \alpha_i [y_i (w^T x_i + b) - 1]$$

The next step is to eliminate w & b from Lagrangian by setting the derivatives of L w.r.t. w & b to zero leading to a dual problem:

$$\frac{\partial L}{\partial w} = w^T x_i - \sum \alpha_i y_i x_i = 0$$

$$\Rightarrow \bar{w} = \sum \alpha_i y_i x_i$$

This tells us that vector \bar{w} can be obtained by a linear sum of the sample vectors. At the end of optimization, a lot of x_i is going to be 0 and non-zero x_i corresponds to support vectors.

$$\frac{\partial L}{\partial b} = \sum \alpha_i y_i = 0$$

$$\text{If we substitute these values of } \bar{w} \text{ & } \sum \alpha_i y_i = 0$$

in the Lagrangian equation we will get

$$L = -\sum \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j (\bar{x}_i \cdot \bar{x}_j)$$

This signifies that optimization depends on dot product of pairs of samples.

* SVM in case of non-linearly separable data:

When data is not linearly separable, then SVM uses

Kernel Trick which helps SVM to project the data in a higher dimensional space, without explicitly computing the coordinates in that space which enables SVM to classify data points and handle non-linear decision boundaries.

So from Lagrangean eqⁿ, we inferred that decision function / optimization was only dependent on the dot product of $\bar{x}_i \cdot \bar{x}_j$, thus we can use any transformation function / kernel function to represent the dot product of $\bar{x}_i \cdot \bar{x}_j$. Thus Lagrange eqn in this term can be expressed as —

Kernel function

$$L = -\sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j K(\bar{x}_i \cdot \bar{x}_j)$$

here $K(\bar{x}_i, \bar{x}_j)$ is the kernel function that computes the dot product in the higher dimensional space, allowing the SVM to find a non-linear decision boundary in the original feature space.

* Common Kernel functions are—

(1) Linear Kernel : Formula : $K(\bar{x}_i, \bar{x}_j) = (\bar{x}_i, \bar{x}_j)$

The kernel corresponds to the standard dot product in the original feature space. It is equivalent to using no kernel at all and is used when the data is linearly separable.

(2) Polynomial Kernel : Formula : $K(\bar{x}_i, \bar{x}_j) = (\bar{x}_i \cdot \bar{x}_j + c)^d$

This kernel maps the data into a higher dimensional polynomial space. The parameter 'd' determines the degree of polynomial, and 'c' is a constant that trades off the influence of higher order and lower order terms.

(3) Radial Basis Function (RBF) / Gaussian Kernel :

$$\text{Formula : } K(\bar{x}_i, \bar{x}_j) = \exp(-\gamma \|\bar{x}_i - \bar{x}_j\|^2)$$

The kernel maps the data into an infinite dimensional space and is widely used for non-linear problems. The parameter γ controls the width of the gaussian function determining how far the influence of a single training example reaches.

(4) Sigmoid Kernel : Formula : $K(\bar{x}_i, \bar{x}_j) = \tanh(\alpha \langle \bar{x}_i, \bar{x}_j \rangle + c)$

This kernel is inspired by neural networks, where \tanh is the activation function.

Advantages of Kernel Trick:

- ① No explicit mapping \rightarrow The kernel trick avoids the explicit computation of coordinates in the higher dimensional space, reducing computational complexity.
- ② It can be used for wide range of non-linear classification problems.

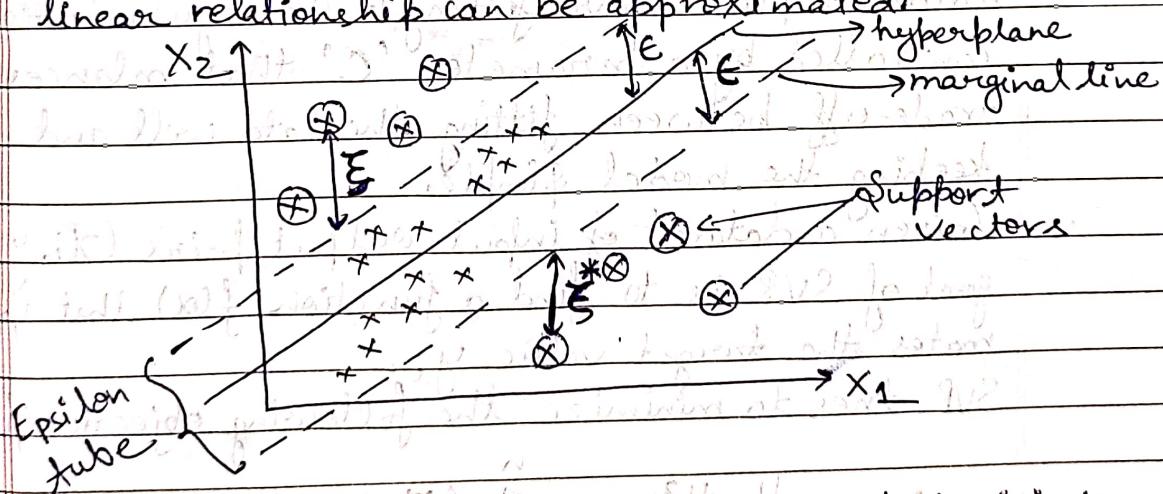
Support Vector Regressor (SVR)

In classification tasks, SVM tries to find a hyperplane that separates data points into different classes by finding maximum margin between support vectors.

In SVR, we aim to predict a continuous value by fitting a hyperplane (or line) that best represents the relationship between input features and target values.

SVR can also handle both linear and non-linear data:

It can apply the kernel trick to transform the data into a higher dimensional space where a linear relationship can be approximated.



- Unlike traditional regression techniques that minimize the sum of squared errors, SVR introduces the concept of ϵ -insensitive loss. In this approach, the algorithm ignores errors that fall within a margin of $\pm \epsilon$ around the predicted value. This means that the small deviations from the true value (within the ϵ margin) are not penalized, focusing the model on capturing larger deviations that matter more.

- The ϵ parameter is a threshold defining the width of the tube around the regression hyperplane, within which no penalty is given for errors. The goal is to fit a function that deviates from the actual target values by at most ϵ .
 - Support Vectors in SVR are the points that lie on the boundary of the ϵ -tube or outside it. These points determine the shape and position of the regression line (or hyperplane in higher dimensions). These are called "support vectors" because they "support" or define the model.
- The objective in SVR is to minimize a regularized loss function that includes:
- * The ϵ -insensitive loss (to ensure that the error is minimized outside the margin).
 - * A regularization term (which penalizes overly complex models to prevent overfitting). The regularization is controlled by a parameter 'C' that balances the trade-off between fitting the data well and keeping the model simple.
- (Given a dataset of input-output pairs (x_i, y_i) , the goal of SVR is to find a function $f(x)$ that approximates the target value y_i .
SVR tries to minimize the following objective function:

$$\min \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n (\xi_i + \xi_i^*)$$

subject to: $y_i - (w \cdot x_i + b) \leq \epsilon + \xi_i$

$(w \cdot x_i + b) - y_i \leq \epsilon + \xi_i^*, \xi_i, \xi_i^* \geq 0$

Here, w represents the weights of the model,

C is the regularization parameter,
 ξ_i and ξ_i^* are slack variables representing the degree to which data points deviate from ϵ -tube.
 b is the bias term.

* The regularization parameter (C) controls the trade off between the flatness of the model and the amount of deviation tolerated beyond the ϵ margin. A small value of C makes the model more tolerant to deviations (simpler models), while a larger value of C makes the model penalize deviations more strictly (complex model).

Advantages of SVR:

- (a) SVR is robust to outliers since it uses the ϵ -insensitive loss function, which ignores small deviations.
- (b) It can be used for modeling both linear as well as non-linear data by using different kernels.
- (c) Using the help of regularization parameter C , it helps preventing overfitting by penalizing complex models.

Disadvantages:

- (a) The quadratic programming optimization used in SVR can be computationally expensive, especially for large datasets.
- (b) SVR requires careful tuning of hyperparameters (kernel, C , ϵ) which can be time consuming.