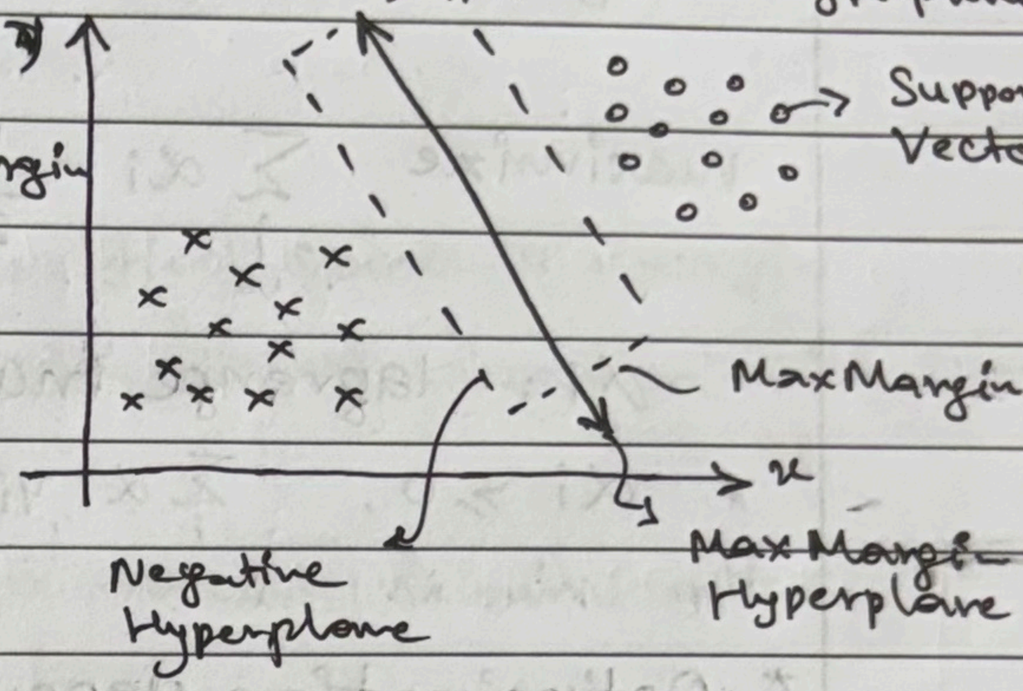
MAXIMISE MARGIN

__/__/____

(11) Support Vector Machines (SVM)

Goal: Is to maximise the margin b/w the two classes.
Larger the margin, better the model performs on unseen data.

1. The best hyperplane - Hard Margin is the one that maximises the distance b/w hyperplane and nearest data points



Find w, b to: max margin $= 2 / \|\vec{w}\|$ Euclidean Norm

Equivalent Optimization: $\min_{w,b} 1/2 \|w\|^2$

subject to $y_i (w^T x_i + b) \geq 1$, $y_i \in \{-1, 1\}$

* No errors allowed ↳ Ensures each data point is correctly
* Sensitive to outliers. classified and lies outside margin.

2. A soft margin allows for some misclassifications or violations of margin to improve generalization.

$$\min_{w,b,\xi} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i$$

subject to $y_i (w^T x_i + b) \geq 1 - \xi_i$, $\xi_i \geq 0$

ξ_i: slack variable - how much the point violates the margin

C: regularization parameter. (trade off b/w margin width and misclassification)

large C: behaves like hard margin

Small C: allows more misclassifications - better generalization

__/__/____

→ Dual Problem in SVM

Dual Problem involves maximizing the Lagrange multipliers associated with the support vectors.

This transformation allows solving the SVM optimization using kernal functions for non-linear classifications.

maximize $\sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j (x_i^T x_j)$

α_i : lagrange multipliers, kernal function that computes similarity b/w datapoints

$\alpha_i \geq 0$, $\sum_i \alpha_i y_i = 0$

* Optimization depends only on dot products $x_i^T x_j$
  → Allows kernal trick later.
* α_i tells how important each training point is in defining the decision boundary.
  If $\alpha_i > 0$: that point lies inside/on the margin - support vector
  If $\alpha_i = 0$: point is far away.

* Weight vector w : $\sum_i \alpha_i y_i x_i$
  only support vectors ($\alpha_i > 0$) contribute.

Once we solve Dual Problem →

* Decision Function (Prediction Rule)
  → After training to classify a new point x,

$$f(x) = \sum_i \alpha_i y_i (x_i^T x) + b \quad \text{and then}$$

Predicted class = $\text{sign}(f(x))$

The magnitude of $f(x)$ is proportional to the confidance of classification.

Large $|f(x)|$ → far from boundary - confident pred.
Small $|f(x)|$ → near boundary - uncertain pred

- Primal to Dual -

__/__/____

o 1. Primal Objective (minimize $1/2 \|w\|^2$)

2. Dual Form → Reframe using Lagrange multipliers

3. Solution ~ find support vectors

4. Decision Function. $f(x) = \sum \alpha_i y_i (x_i^T x_i) + b$: Prediction

5. Classification ~ sign($f(x)$) Determines the label

→ Hinge loss: Penalizes misclassified points or margin violations and is combined with regularization with sum.

for margin maximization and penalty minimization

$h(y_i, f(x_i)) = \max(0, 1 - y_i f(x_i))$; $f(x_i) = w^T x_i + b$

If $y_i f(x_i) \geq 1$: correctly classified

If $y_i f(x_i) < 1$: inside margin / misclassified

→ Soft Margin with Hinge loss

$$\min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \max(0, 1 - y_i(w^T x_i + b))$$

Margin Maximization ↓ ↓ Hinge loss

* Linear SVM is thus a linear model trained with Hinge lose + L2 Regularization.

(12) SVM Kernals

In dual form, classifier depends upon only dot products

$$f(x) = \sum_i \alpha_i y_i (x_i^T x) + b$$

We can replace this dot product with a kernal function.

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

where $\phi(\cdot)$ maps data to a larger dimension

__/__/____

A kernal is a function that maps data points into a higher dimensional space without explicitly computing the coordinates in that space.

1. Linear Kernal, when data is linearly seperable.
   : $K(x, y) = x \cdot y$

2. Polynomial kernal, allows SVM to model more complex relationships by introducing polynomial terms. : $K(x, y) = (x \cdot y + c)^d$

3. RBF (Radial Basis Function) kernal,
   It maps data into an infinite - dimensional space making it highly effective for complex classification problems. : $K(x, y) = e^{-}(\gamma \|x - y\|^2)$
   γ : parameter that controls influence of each samp

4. Sigmoid kernal ~ behave similarly to the activation function of a neuron.
   $K(x, y) = \tanh(\gamma \cdot x^T y + r)$

→ Basis Expansion (Feature Transformation)
Sometimes we explicitly add non-linear features to make data linearly seperalde.
- Suppose we've a 1D data that is not linearly seperable.
- We can't seperate it using a line in 1D.
- Expand the features: $\phi(x) = [1, x, x^2]$
  in 2D/3D it becomes linearly seperable.

1. $\phi(x) = [1, x, x^2]$ → Non-monotomic function →
   can make non seperable data, seperalde
2. $\phi(x) = [1, x^3]$ → Monotomic → Doesn't change seperability.