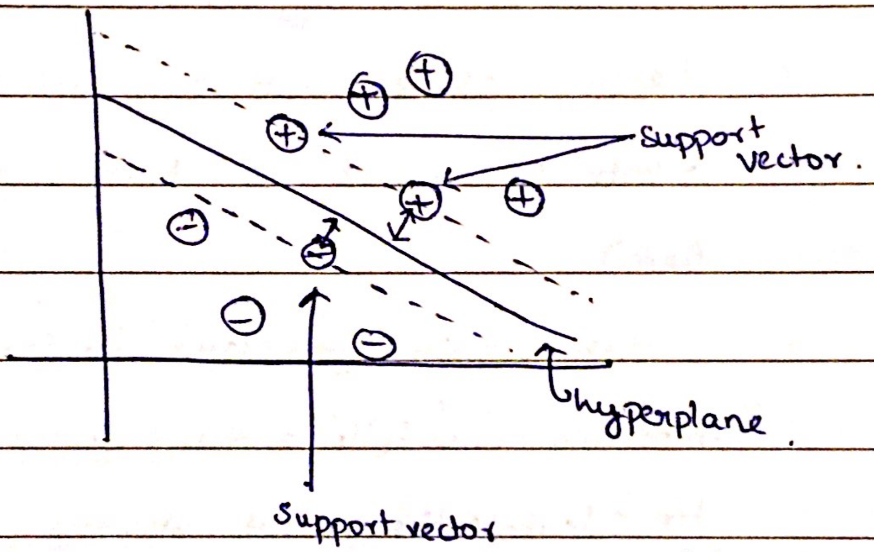
Support Vector Machines

→ Let us start with the basic intuition of SVM. Let us say we have a 2-D space (can be n-D, but for simplification let us assume a 2-D space)



The objective of SVM is to find a hyperplane of dimensions (n-1) which seperates the two classes such that the hyperplane is at a maximum distance from the points which define the boundary of the classes (a.k.a. support vectors)

→ Some basic info on SVMs:

- Supervised learning algo
- Unlike Logistic Regression this can also work on non-linearly seperable data using kernels.

→ <u>The mathematics of it :-</u>

The hyperplane can be represented by the equation

$$\vec{w} \cdot \vec{x} + b = 0$$

where $\vec{w} \in$ coefficients $\vec{x} \in$ features, $b \in$ bias.

The hyperplane equation is such that it is equidistant to both the support vector planes (and is therefore in between them)

With just the hyperplane $wx + b = 0$, we have two unknowns → 'w' & 'b' and we donot have enough constraints to calculate them, but with the support vector plane equations defined as

$$w \cdot x + b = 1 \quad , \quad w \cdot x + b = -1$$

where if $w \cdot x + b \leq -1$ $y = -1$

& $w \cdot x + b \geq 1$ $y = 1$ we can find w & b.

we can now combine the equations as

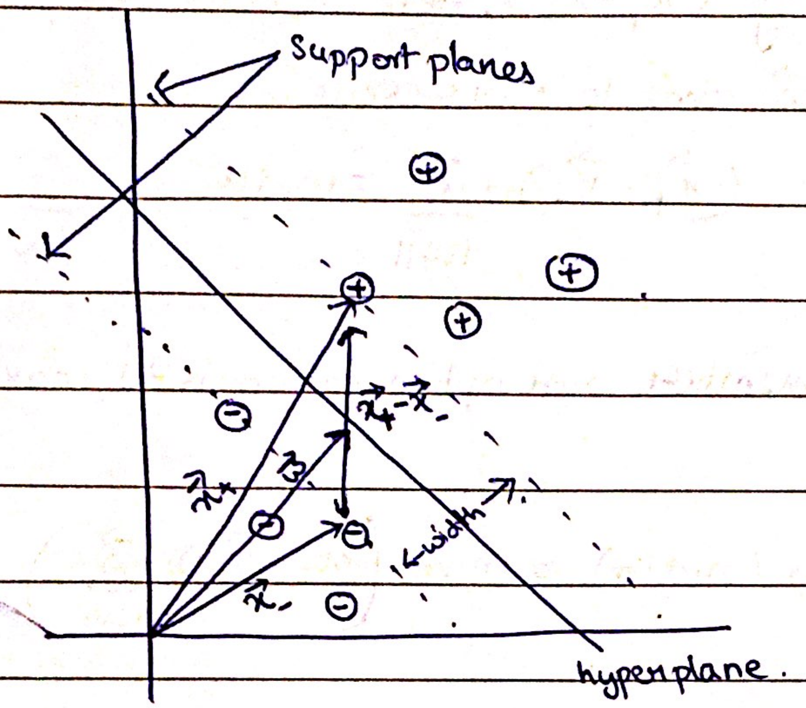
$$y * (\vec{w} \cdot \vec{x} + b) \geq 1$$

$$\Rightarrow \boxed{y_i (\vec{w} \cdot \vec{x}_i + b) - 1 = 0}$$

← for all samples which lie on the support vector planes i.e. the support vectors

(*)

Let us take the graph again



Let us try to understand a few more things about it.

1) $\vec{w}$, which were our coefficients can be thought of as a vector perpendicular to the hyperplane. Therefore $\vec{w} \cdot \vec{x}$ is the projection of $\vec{x}$ on $\vec{w}$. Now if this projection + bias is ≥ 1 that means that it is a $y_i = 1$ classification. Similarly if projection + bias ≤ -1 means that it is a $y_i = -1$ classification. Therefore our intuition with $\vec{w}$ as a perpendicular vector to hyperplane is justified.

Anyway coming back:-

Lets take one point on the ⊕ support plane as $\vec{x}_+$

Lets take on point on the ⊖ " " as $\vec{x}_-$

$\therefore$ we can say that the width between the support planes can be the perpendicular projection of $\vec{x}_+ - \vec{x}_-$ on the hyperplane.

looking at the graph.

The question is how do we find the perpendicular projection of $(\vec{x}_+ - \vec{x}_-)$. Here we have our perpendicular vector $\vec{w}$ come to our rescue.

$$\therefore (\vec{x}_+ - \vec{x}_-) \cdot \frac{\vec{w}}{||w||} = \text{width} \qquad \frac{\vec{w}}{||w||} = \hat{w} \in \text{unit vector}$$

Now remember our optimization is to maximise this width

$$\therefore \max(\text{width}) = \max\left((\vec{x}_+ - \vec{x}_-) \cdot \frac{\vec{w}}{||w||} \right)$$

now remember our eq^n where we said for all support vectors

$$y_i(\vec{w} \cdot \vec{x}_i + b) - 1 = 0$$

$\therefore$ for $x_+ \Rightarrow y_i = 1 \quad \Rightarrow \vec{w} \cdot \vec{x}_+ + b - 1 = 0$

$$\therefore \vec{w} \cdot \vec{x}_+ = 1 - b$$

for $x_- \Rightarrow y_i = -1 \quad \Rightarrow -(\vec{w} \cdot \vec{x}_- + b) - 1 = 0$

$$\therefore \vec{w} \cdot \vec{x}_- = -(1+b)$$

$$\therefore (\vec{x}_+ - \vec{x}_-) \cdot \frac{\vec{w}}{||w||} = \frac{\vec{x}_+ \cdot \vec{w} - \vec{x}_- \cdot \vec{w}}{||w||} = \frac{1 - b + 1 + b}{||w||}$$

$$= \frac{2}{||w||}$$

$\therefore \max(\text{width}) = \max\left(\frac{(\vec{x}_+ - \vec{x}_-)\cdot\vec{w}}{\|w\|}\right) = \max\left(\frac{2}{\|w\|}\right)$

$= \min(\|w\|)$

$= \min\left(\frac{1}{2}\|w\|^2\right)$

↑ for mathematical purposes.

Now to solve any maximise/minimise eq^n with constraints we shall use Lagrange Multiplier.

$L = (\text{eq}^n \text{ to max/min}) - \sum \alpha_i (\text{constraint})$

Substituting our terms.

$L = \frac{1}{2}\|w\|^2 - \sum \alpha_i [y_i(\vec{w}\cdot\vec{x}_i + b) - 1]$

Now we need to equate $\frac{\partial L}{\partial w} = 0$ (to find maxima/minima)

$\frac{\partial L}{\partial w} = \vec{w} - \sum \alpha_i y_i x_i = 0 \Rightarrow \boxed{\vec{w} = \sum \alpha_i y_i \vec{x}_i}$

Similarly $\frac{\partial L}{\partial b} = -\sum \alpha_i y_i = 0 \Rightarrow \boxed{\sum \alpha_i y_i = 0}$

Now lets plug this into our lagrange equation.

$$L = \frac{1}{2}\left(\sum_i \alpha_i y_i \vec{x}_i\right)\cdot\left(\sum_j \alpha_j y_j \vec{x}_j\right) - \left(\sum_i \alpha_i y_i x_i\right)\cdot\left(\sum_j \alpha_j y_j x_j\right)$$

$$- \sum_i \alpha_i y_i b + \sum_i \alpha_i$$

$$L = -\frac{1}{2}\left(\sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i \cdot x_j\right) - b \underbrace{\sum_i \alpha_i y_i}_{0} + \sum_i \alpha_i$$

0 . from previous eq^n

Lagrange eq^n

$$\boxed{L = \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \vec{x}_i \cdot \vec{x}_j}$$

This eq^n shows that the entire thing just boils down to a dot product $(\vec{x}_i \cdot \vec{x}_j)$

Let us also substitute $\vec{w} = \sum \alpha_i y_i \vec{x}_i$ in our hyperplane eq^n.

$$\vec{w} \cdot \vec{u} + b = 0$$

$$= \boxed{\sum \alpha_i y_i \vec{x}_i \cdot \vec{u} + b = 0}$$

This is our optimization equation.

→ Solving the Lagrange eq^n: in PYTHON:-

$$L = \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \vec{x}_i \vec{x}_j$$

maximise α_i

given $\alpha_i \geq 0$ $\quad \sum_i \alpha_i y_i = 0$.

for all support vectors

This eq^n can be solved using quadratic programming where eq^n's of the form can be solved by CVXOPT library's QP solver

$$\min_{x} \frac{1}{2} x^T P x + q^T x$$

subject to $Gx \leq h$

$Ax = b$.

The lagrange eq^n can be written as

$$-\left(\underbrace{(-1)^T}_{q} \alpha + \frac{1}{2} \alpha^T \overbrace{\begin{bmatrix} y_1 y_1 x_1^T x_1 & y_1 y_2 x_1^T x_2 & \cdots & y_1 y_N x_1^T x_n \\ y_2 y_1 x_2^T x_1 & y_2 y_2 x_2^T x_2 & \cdots & y_2 y_N x_2^T x_N \\ \vdots & \cdots & \cdots & \cdots \\ y_N y_1 x_N^T x_1 & y_N y_2 x_N^T x_2 & \cdots & y_N y_N x_N^T x_N \end{bmatrix}}^{P} \alpha \right)$$

subject to

$\alpha \geq 0$ & $\overset{A}{y^T} \alpha = \underset{b}{0}$.

$\Rightarrow \underset{G}{(-1)} \alpha \leq \underset{h}{0}$

(8)