

Support vector machine (SVM).
Linear classifier

- Let us first consider a linear classifier which is basically a linear discriminant function. So if we have an input feature vector x & if we define a linear discriminant function $g(x)$ and for simplicity we consider a 2-class problem, two classes are C_1 & C_2 . So any unknown feature vector say x is to be classified as either belonging to class C_1 or to class C_2 and the linear discriminant function is $g(x)$.
 $\therefore g(x) = w^T x + b \leftarrow$ As linear fn.
 wt. vectors input feature vector
 $w^T x$ bias
 $w^T x + b = 0 \Rightarrow$ st. line
 $w^T x + b = 0 \Rightarrow$ if x is 2D
 $w^T x + b = 0 \Rightarrow$ plane
 $\quad \quad \quad$ if x is 3D
 $g(x)$ is a st. line
 $g(x)$ is a plane

but a vector per hyperplane (i.e. normal vector).
 \vec{W} represents the orientation of the hyperplane
 $\& b$ represents what is the position of the hyperplane
 in D -dimensional space, b is usually known as a bias term which is biasing the position of the hyperplane.
 For classification problem, here for every feature vector x_1 , we have to compute $g(x_1) = \vec{W}^T x_1 + b$. If $g(x_1) > 0$
 if x_1 lies on the +ve side of the hyperplane
 and if x_1 lies on -ve side of the hyperplane then
 $g(x_1) < 0$. And if x_1 lies on the hyperplane
 then $g(x_1) = 0 \Leftrightarrow \vec{W}^T x_1 + b = 0$.
 The hyperplane divides the D -dimensional space into two halves.

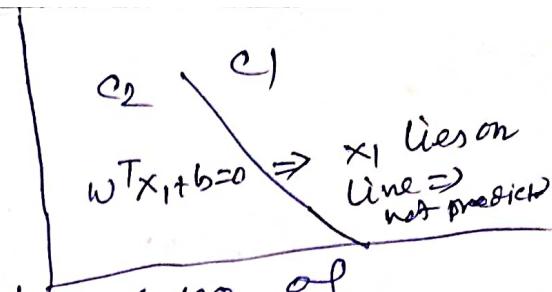


3A

∴ The classifier rule is:

$$\text{If } w^T x_1 + b > 0 \Rightarrow x_1 \in C_1$$

$$\text{If } w^T x_1 + b < 0 \Rightarrow x_1 \in C_2$$



So for training we must have large no. of samples, some of which are of C_1 & some of which are of C_2 . We have to train the model, i.e., estimate w & b such that certain objective function will be optimised need to optimize. (like some of square error).

Training will be in iterative fashion.

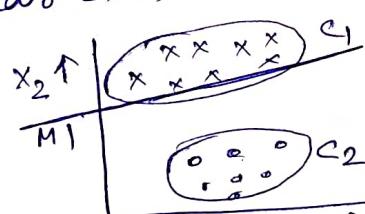
Let us consider any sample x_1 from class C_1 . Then we start with an initial value of \tilde{w} and b & compute $w^T x_1 + b$. ~~If~~ It should be > 0 . But if it is ≤ 0 , then we modify \tilde{w} & b in such a way that the position and orientation of the hyperplane is so modified that the sample x_1 moves to the side of the hyperplane.

Similarly if we take a input feature vector x_2 of class C_2 , the same situation will happen.

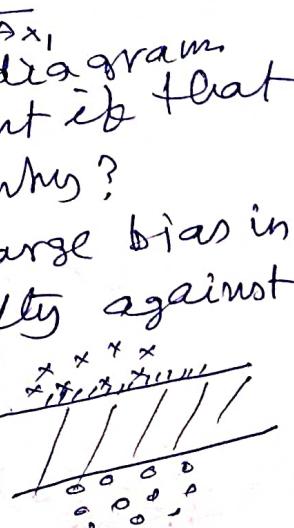
This is done in case of linear discriminant function.

Now let x_1 & x_2 are two features & the samples are features of classes C_1 & C_2 . And our linear classifier (M_1) is as shown in the diagram. It correctly classified all samples. But is that classifier is desired? Answer is NO: why?

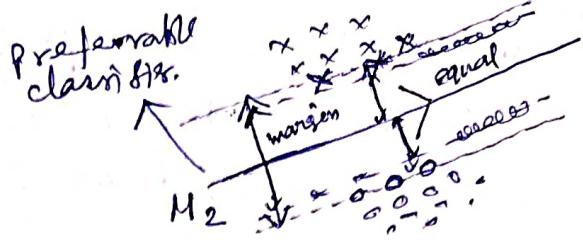
~~It is not desired because M_1 gives a large bias in favour of C_2 , whereas it puts a penalty against class C_1 . The reason is → the entire space between two type of samples is given to C_2 & only a small margin is given to C_1 .~~



Initially on left side of hyper plane $g(x)$
 $g(x)$ modified to $g'(x)$ & p now lies on right side of $g'(x)$.

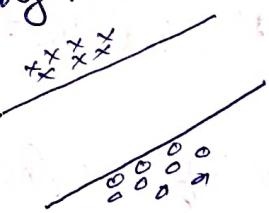


M_2 is more preferable classifier because the two training vectors of both C_1 & C_2 are equally apart from the classifier.



The SVM actually tries to find a classifier which will be positioned in a form like M_2 . Let us see how such a SVM can actually be designed. So the basic aim of the SVM is: boundary (B^2) If a sample x is one side of B^2 and it seems to be saved if its distance from B^2 is larger. If the distance is very small even a little disturbance/noise can push x to move to other side of B^2 which implies a misclassification. So it is desirable that the margin of x from the classifier will be quite large so that the sample x will be in the same side of the classifier until a large disturbance/noise is applied.

∴ large distance from classifier \Rightarrow safe sample.
∴ large distance to design the classifier that maximizes the distance of the separating boundary between two classes C_1 & C_2 .



$$\vec{x}_i = (x_{i1} x_{i2} \dots x_{id}) \& w = (w_1 w_2 \dots w_d)$$

$$\vec{x}_i^T w + b \geq 0$$

$$\therefore \text{if } x_i \in C_1 \Rightarrow \vec{x}_i^T w + b \geq 0$$

$$\& \text{if } x_i \in C_2 \Rightarrow \vec{x}_i^T w + b < 0$$

As SVM is supervised learning so we know the class belongings of every x_i . Let class belongings of x_i is y_i . $\therefore y_i = \pm 1$

so for any $(x_i y_i)$ $y_i (\vec{x}_i^T w + b) > 0$ since if $y_i = +1$ then

$$+1 \Rightarrow C_1 \text{ class}$$

$$-1 \Rightarrow C_2 \text{ class}$$

$$y_i (\vec{x}_i^T w + b) > 0 \text{ since if } y_i = +1 \text{ then } \vec{x}_i^T w + y_i > 0$$

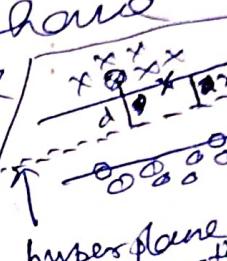
$$\& \text{if } y_i = -1 \text{ then } \vec{x}_i^T w + y_i < 0$$

	x_1	x_2	\dots	x_d	y
:					
x_i	x_{i1}	x_{i2}	\dots	x_{id}	y_i

Using this concept, i.e., $y_i(w^T x_i + b) > 0$, we are designing the classifier. [$E=0$ only when x_i lies on classifier] Once we set w & b , for an unknown feature vector p , we don't have y_i , so we compute $w^T p + b$. If $w^T p + b > 0$ then we classify p to class C_1 and if $w^T p + b < 0$ then we classify p to class C_2 .

In SVM we have the objective ~~is~~ that we have to minimize the distance of the hyperplane or the separating boundary from each of the feature vector, so that every feature vector feels that they are safer. So far this classifier is concerned.

SVM made a modification in our linear



so far this classifier is con
 To do that, we have made a modification in our linear
 classifier design. Earlier we have considered \rightarrow
 $w^T x_i + b > 0 \text{ if } x_i \in C_1$
 $\& w^T x_i + b < 0 \text{ if } x_i \in C_2$

NOW we consider

If $x_i \in C_1$ then $w^T x_i + b \geq \gamma$ where, γ = some margin which is nothing but a measure of distance of x_i from the separating plane.

If $x_i \in C_2$ then $w^T x_i + b \leq -\gamma$

So if we have a hyperplane whose equation is $w^T x + b = 0$ then distance of the point x from the hyperplane is $d = \frac{|w^T x + b|}{\|w\|}$

So we want $d = \frac{|\mathbf{w}^T \mathbf{x} + b|}{\|\mathbf{w}\|} \geq \gamma \Rightarrow -\gamma \leq \mathbf{w}^T \mathbf{x} + b \leq \gamma$
 Our decision regarding the correct classification of \mathbf{x} is independent of scaling vector $\vec{\mathbf{w}}$ because $\vec{\mathbf{w}}$ tells us only about the orientation of the plane.
 So whatever scaling factor is applied to $\vec{\mathbf{w}}$, our decision remains the same.
 \therefore we may write $\mathbf{w}^T \mathbf{x} + b \geq \gamma \|\mathbf{w}\| \Rightarrow 1$
 \therefore we may set, $\mathbf{w}^T \mathbf{x} + b \geq 1$.

And by proper scaling, we may set, $w^T x + b \geq 1$

$$\left[\frac{w^T x + b}{\|w\|} \geq \frac{1}{\|w\|} \right] \Rightarrow \left[\frac{w^T x + b}{\|w\|} - \frac{1}{\|w\|} \leq 0 \right]$$

$$\Rightarrow w^T x + b - \frac{1}{\|w\|} \leq 0 \Leftrightarrow w^T x + b \leq \frac{1}{\|w\|}$$

i.e., if $y > \frac{1}{\|w\|}$

36/1

So after in this situation we can say that for every feature vector x ,

$$w^T x + b \geq 1 \text{ if } x \in C_1$$

$$\leq -1 \text{ if } x \in C_2$$

$$C_1: w^T x + b = 1$$

$$C_2: w^T x + b = -1$$

∴ During learning process, for every vector x_i

always $y_i(w^T x_i + b) \geq 1$

~~If~~ $y_i(w^T x_i + b) = 1$ if x_i is a support vector and
 $y_i(w^T x_i + b) > 1$ if x_i is not a "

So what is ~~not~~ the support vector?

Let us consider 2-D feature vectors.

Let us consider 2-D feature vectors with features x_1 & x_2 . Then the samples

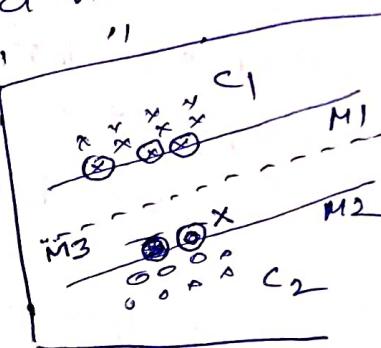
are plotted. If we consider a st. line M_1 of class C_1 , where 3 samples are passing through M_1 .

Similarly, to 3 samples of class C_2 are passing through another st. line M_2 .

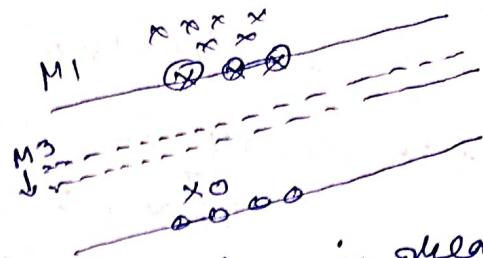
So these 3 samples of the training set are just on boundary. If any disturbance occurs from there is the possibility that these samples may be misclassified.

→ So we should place our classifier somewhere in the middle of M_1 & M_2 . So M_3 be the position of the classifier which will be more reliable or it is called more generalized classifier. So the risk of misclassification will be less.

∴ the position of ~~the~~ M_3 depends on the position of these ~~4~~ $(3+1)$ feature vectors or training samples. And it does not depend on the position of other feature vectors. Even if we remove some other feature vectors from the training set, the position of the hyper plane (M_3) remains the same. But if we remove, say x_1 (which lies on M_2) then our hyper plane



is going to be different,
so these $3+1 = 4$ feature
vectors are called support
vectors, others are not.



\therefore A SVM is a linear machine whose design is greatly influenced by the positions of the support vectors, and are not that much influenced by the non-support support vectors. OK.

So we have $y_i (w^T x_i + b) = 1$ if x_i is a support vector.

\therefore We want margin γ as maximum as possible where, $\frac{w^T x + b}{\|w\|} \geq \gamma$. So we have to minimize w and maximize b .

So when we try to find the support vectors, we have to minimize w and maximize b .

To minimize w : it is same as, we want to minimize a function, $\phi(w) = w^T w = \|w\|^2$

$$\begin{aligned} & w_1 x_1 + w_2 x_2 + b \\ & \sqrt{w_1^2 + w_2^2} \\ & \|w\|^2 = w_1^2 + w_2^2 \\ & \text{minimize } \|w\|^2 = \frac{1}{2} w \cdot w \end{aligned}$$

\therefore We need to minimize $\phi(w) = \frac{1}{2} w \cdot w$

Trivial value of w is $\vec{0}$, but we are not looking for it.

Here my constraint is $y_i (w \cdot x_i + b) = 1$

So we have to minimize w or $\phi(w) = \frac{1}{2} w \cdot w$

subject to the constraint $y_i (w \cdot x_i + b) \geq 1$ but as support vectors ~~satisfy~~ satisfy equality so we consider the constraint $y_i (w \cdot x_i + b) = 1$.

So it is a constrained optimization problem. This problem can be converted to an unconstrained optimization problem by using the Lagrangian multiplier.

$$\begin{aligned} x_i &= (x_{i1}, x_{i2}, \dots, x_{id}) \\ w &= (w_{11}, w_{12}, \dots, w_{1d}) \\ &\vdots \\ &\vdots \\ &\text{No wts.} \end{aligned}$$

36/2

Let M_1 is the hyperplane & x' is a feature vector of C_1 class.

i.e. position vector of x is \vec{ox} .
∴ position vector of x' from x to hyperplane
say it meets at x'
∴ position vector of $x' = \vec{ox}'$

let $x'x = \vec{r}$
The hyperplane is $w^T x + b = 0$ where
 \hat{w} is the unit vector normal to the
hyperplane.

$$\therefore \vec{r} = x'x = \vec{ox} - \vec{ox}' = \vec{x} - \vec{x}'$$

* For x' , $w^T x' + b = 0$
 \vec{r} is \perp to hyperplane, so \vec{r} is \parallel to \hat{w}
 $\vec{r} = \vec{x} - \vec{x}'$ gives $x' = \vec{x} - \vec{r} = \vec{x} - \frac{\vec{r}}{\|\vec{w}\|}$ [As we are interested
in margin]

$$\therefore \vec{r} = \gamma \hat{w} \sim \gamma \frac{\vec{w}}{\|\vec{w}\|}$$

$$\therefore \vec{r} = \vec{x} - \vec{x}' \text{ gives } x' = \vec{x} - \frac{\vec{r}}{\|\vec{w}\|} = \vec{x} - \frac{\gamma \vec{w}}{\|\vec{w}\|}$$

$$\therefore w^T x' + b = 0 \Rightarrow w^T \left(\vec{x} - \frac{\gamma \vec{w}}{\|\vec{w}\|} \right) + b = 0$$

$$\Rightarrow w^T \vec{x} - \frac{\gamma w^T \vec{w}}{\|\vec{w}\|} + b = 0$$

$$\Rightarrow w^T \vec{x} - \gamma \|\vec{w}\| + b = 0$$

$$\Rightarrow \gamma = \frac{w^T \vec{x} + b}{\|\vec{w}\|}$$

$$\therefore w^T w = \|\vec{w}\|^2$$

$$\|\vec{w}\| = \sqrt{w_1^2 + w_2^2 + \dots}$$

Similarly, if x is in class C_2 :

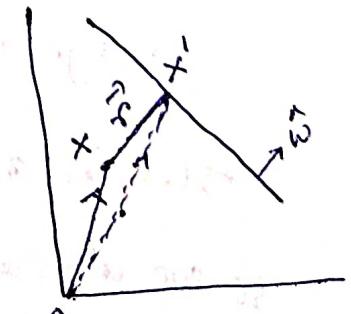
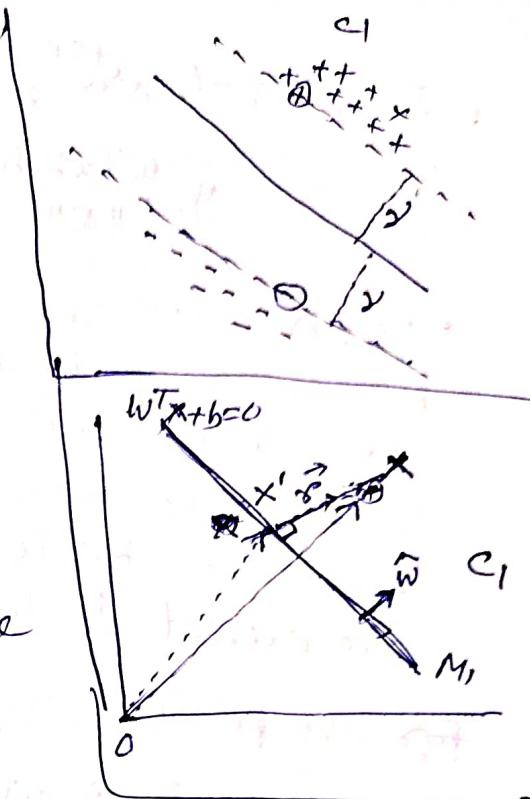
$$\vec{r} = x'x = \vec{ox}' - \vec{ox} = \vec{x}' - \vec{x}$$

$$\text{For } x', w^T x' + b = 0. \quad \vec{r} = \gamma \hat{w} \text{ & } x' = \vec{r} + \vec{x} = \frac{\gamma \vec{w}}{\|\vec{w}\|} + \vec{x}$$

$$\therefore w^T x' + b = 0 \Rightarrow w^T \left(\vec{x} + \frac{\gamma \vec{w}}{\|\vec{w}\|} + b \right) = 0$$

$$\Rightarrow w^T \vec{x} + \frac{\gamma w^T \vec{w}}{\|\vec{w}\|} + b = 0 \Rightarrow w^T \vec{x} + \gamma \|\vec{w}\| + b = 0$$

$$\Rightarrow \gamma = -\frac{w^T \vec{x} + b}{\|\vec{w}\|}$$



i. In general, the margin for any point C may be in C_1 or C_2

$$\text{ii. } \gamma = y \cdot \frac{\mathbf{w}^T \mathbf{x} + b}{\|\mathbf{w}\|} \quad (\gamma = \text{label of } \mathbf{x} \text{ if } \mathbf{x} \text{ in } C_1 \Rightarrow \gamma = 1 \\ \text{if } \mathbf{x} \text{ in } C_2 \Rightarrow \gamma = -1)$$

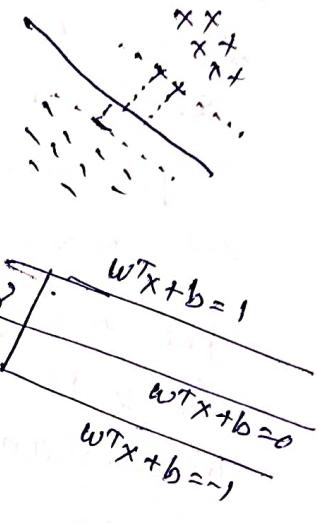
The objective of SVM is to maximize this margin.
We can compute the margin for every data point but
cannot maximize all margins.

We try to maximize the margins for
the data points of both classes which
are closest to the hyper plane.

for +ve points (in C_1), $\gamma = \frac{\mathbf{w}^T \mathbf{x} + b}{\|\mathbf{w}\|}$

i. for +ve points (in C_1) on margin (\Leftrightarrow support vectors),

$$\gamma = \frac{1}{\|\mathbf{w}\|} \quad [\because \mathbf{w}^T \mathbf{x} + b = 1 \text{ for those points}]$$



Similarly, for -ve points on margins,

$$\gamma = -\frac{\mathbf{w}^T \mathbf{x} + b}{\|\mathbf{w}\|} = \frac{-(-1)}{\|\mathbf{w}\|} \quad [\because \text{for -ve points on margins (support vectors), } \mathbf{w}^T \mathbf{x} + b = -1]$$

$$= \frac{1}{\|\mathbf{w}\|}.$$

i. Total margin is $\gamma = \frac{2}{\|\mathbf{w}\|}$.

i. we want to maximize margin \Rightarrow we want to

$$\text{maximize } \frac{2}{\|\mathbf{w}\|}$$

i. we have to minimize, $\phi(\mathbf{w}) = \|\mathbf{w}\| \Rightarrow \frac{1}{2} \|\mathbf{w}\|^2 = \frac{1}{2} \mathbf{w} \cdot \mathbf{w}$
optimization formulation is:

$$\text{minimize}_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2$$

subject to constraints, $y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, \text{ for } i = 1, 2, \dots, m$
~~that are considering only the support vectors~~

This is a constrained optimization problem.
so we ~~can't~~ cannot apply here gradient descent method.

[For regression or perception we have unconstrained optimization]

36/3

We can't apply linear programming, as though constraints are linear but objective function is non-linear (quadratic).

- we have quadratic objective function with m-inequality constraints. To solve this we have many methods (like in MATLAB → quadprog, in C CVXOPT (convex optimization package), guess that some package in python should exist.

But this is not the good solution.

Another best technique is used

Lagrangian method

Method 1: for solving constrained optimization problems of differentiable functions.

$$\underset{x,y}{\text{minimize}} \quad f(x,y) = x^2 + y^2 \quad (\text{quadratic})$$

$$\text{subject to} \quad g(x,y) = x+y-1 = 0 \quad (\text{equality})$$

Need to supply initial weights and constraints to get the result

This convert the constrained opt. meth. to unconstrained opt. problem

A Lagrangian multiplier (α) combine the two equations into one: minimize $L(x,y,\alpha) = f(x,y) - \alpha g(x,y)$

whatever path we are getting will satisfy the constraints.

Method 2: for m-constraints

$$\underset{x,y}{\text{min}} \quad f(x,y) = x^2 + y^2$$

$$\text{s.t.} \quad g_1(x,y) = x+y-1 = 0 \Rightarrow \underset{x,y,\alpha}{\text{min}} \quad L(x,y,\alpha) = f(x,y) - \sum_{i=1}^m \alpha_i g_i(x,y)$$

Method 3: Handling In-equality constraints:

$$\underset{x,y}{\text{minimize}} \quad f(x,y) = x^2 + y^2$$

$$\text{s.t.} \quad g(x) = x-1 \geq 0$$

Here, inequality constraints are transferred as constraints and $g(x) \geq 0 \Rightarrow \alpha \geq 0$; $g(x) \leq 0 \Rightarrow \alpha \leq 0$ and $g(x) = 0 \Rightarrow \alpha$ is unconstrained.

$$\min_{w, b} L(w, y, \alpha) = f(x, y) + \alpha g(x)$$

subject to $\alpha \geq 0$

f is still a constrained optimization problem but simpler one.

This helps to solve our SVM problem:

$$\min_{w, b} \frac{1}{2} \|w\|^2 - \sum_i y_i (w^T x_i + b) \geq 1, \quad i=1, 2, \dots, m$$

s.t. $y_i (w^T x_i + b) \geq 1, \quad i=1, 2, \dots, m$ we

Introducing Lagrange multipliers $\alpha_1, \alpha_2, \dots, \alpha_m$ we

$$\text{get } \min_{w, b, \alpha} L(w, b, \alpha) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^m \alpha_i [y_i (w^T x_i + b) - 1]$$

Here also we may use package and get solution.

We define a Lagrangian function which is to be minimized

$$\min_{w, b} L(w, b) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^m \alpha_i [y_i (w^T x_i + b) - 1]$$

s.t. $\alpha_i \geq 0$

We will use dual programming problem so its dual is:

$$\begin{aligned} \max_{\alpha} d(\alpha) &= \max_{\alpha} \min_{w, b} L(w, b) \\ &= \max_{\alpha} \left\{ \min_{w, b} \left[\frac{1}{2} \|w\|^2 - \sum \alpha_i [y_i (w^T x_i + b) - 1] \right] \right\} \end{aligned}$$

$$\begin{aligned} \sum_{i=1}^m \alpha_i y_i (w \cdot \vec{x}_i) &= \alpha_1 y_1 (w \cdot \vec{x}_1) + \alpha_2 y_2 (w \cdot \vec{x}_2) + \dots + \alpha_m y_m (w \cdot \vec{x}_m) \\ &= \alpha_1 y_1 [(\sum \alpha_i y_i \vec{x}_i) \cdot \vec{x}_1] + \alpha_2 y_2 [(\sum \alpha_i y_i \vec{x}_i) \cdot \vec{x}_2] + \dots \\ &= \alpha_1 y_1 [\alpha_1 y_1 \vec{x}_1 + \alpha_2 y_2 \vec{x}_2 + \dots], \vec{x}_1 + \dots \\ &\geq \alpha_1 y_1 (\alpha_1 y_1 \vec{x}_1 + \alpha_2 y_2 \vec{x}_2 + \dots) \quad \downarrow \\ &= \sum \alpha_i \alpha_j y_i y_j (\vec{x}_i \cdot \vec{x}_j) \end{aligned}$$

$$\vec{w} \cdot \vec{w} = (\sum \alpha_i y_i \vec{x}_i) \cdot (\sum \alpha_j y_j \vec{x}_j) = \sum \alpha_i \alpha_j y_i y_j (\vec{x}_i \cdot \vec{x}_j)$$

37.

So we can define a Lagrangian function as $L(w, b)$ (1)

$L(w, b) = \frac{1}{2}(w \cdot w) - \sum \alpha_i [y_i(w \cdot x_i + b) - 1]$ and
 minimize w maximize α_i
 we have to optimize ~~also~~ the
 Lagrangian multiplier function.

where α_i = Lagrangian multiplier.

For optimization, we have to minimize w & maximize α_i of this Lagrangian L we know

$\therefore \frac{\partial L}{\partial b} = 0$ that this optimization has to be done

by computing $\frac{\partial L}{\partial w}$ & $\frac{\partial L}{\partial b}$

Note expanded form of L is

$$L(w, b) = \frac{1}{2}(w \cdot w) - \sum \alpha_i y_i (w \cdot x_i) - \sum \alpha_i y_i b + \sum \alpha_i \quad (2)$$

$\therefore \frac{\partial L}{\partial b} = - \sum \alpha_i y_i = 0$ for ~~maximize~~ optimization. (3)

\therefore one constraint is: $\sum_{i=1}^m \alpha_i y_i = 0$ where $m =$
 no. of feature vectors.

Note $\frac{\partial L}{\partial w} = ?$

$$L(w, b) = \frac{1}{2}(w \cdot w) - \sum \alpha_i y_i (w \cdot x_i) - \sum \alpha_i y_i b + \sum \alpha_i$$

$\therefore \frac{\partial L}{\partial w} = w - \sum \alpha_i y_i x_i = 0 \Rightarrow$

$$w = \sum_{i=1}^m \alpha_i y_i \vec{x}_i$$

(4) Another constraint
 we get the Lagrangian expression:

Now putting w from (4) in (2) we get

$$L = \frac{1}{2}(w \cdot w) - \sum \alpha_i y_i (w \cdot x_i) - \sum \alpha_i y_i b + \sum \alpha_i$$

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

$$= \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum \alpha_i \alpha_j y_i y_j (\vec{x}_i \cdot \vec{x}_j) \quad (5)$$

\therefore we have to ~~maximize~~ this Lagrangian L with
 different values of α_i (which are the Lagrangian
 multipliers & they are always positive).

Maximize because
 we apply dual method
 see later.

so we have ~~another~~^{one} constraint, $d_i \geq 0$. — (6)
 & the other constraint is $\sum_{i=1}^m d_i y_i = 0$

∴ we have to find out the Lagrangian multipliers which
~~maximize~~^{minimize} the expression L .

When we try to ~~maximize~~^{minimize} L , it is quite likely that
 some of the Lagrangian multipliers will be equal to 0,
 some value will be very high.

⇒ If some $d_i = 0$ that indicates that the corresponding training feature vector \vec{x}_i is not a support vector. so it does not influence the position of the hyperplane.

⇒ If an d_i is very high \Rightarrow the corresponding \vec{x}_i has a high influence over the position of my decision surface or the hyperplane.

⇒ If d_i is extraordinarily high, we can infer that the corresponding \vec{x}_i is a spurious point, which might have occurred due to noise. It is a disturbed point or outlier.

All these different types of interpretations we can have over d_i .

∴ Maximize $L = \sum_{i=1}^m d_i - \frac{1}{2} \sum d_i d_j y_i y_j (\vec{x}_i \cdot \vec{x}_j)$

s.t constraints $d_i \geq 0$ &

$$\sum d_i y_i = 0$$

using QP gives us d_i 's. With these d_i 's

The solution gives us d_i 's [from (4)] is the value of $W = \sum d_i y_i \vec{x}_i$ in our decision making process. computed and used in our decision making process, so our classification decision will be as follows:

for an unknown feature vector \vec{z} , the classification decision at

$$D(\vec{z}) = \operatorname{sgn} (\vec{w} \cdot \vec{z} + b) = \operatorname{sgn} \left(\sum_{j=1}^m d_j y_j (\vec{x}_j \cdot \vec{z}) + b \right)$$

∴ If $D(\vec{z}) > 0$ then $\vec{z} \in C_1$

< 0 then $\vec{z} \in C_2$

Steps of SVM design

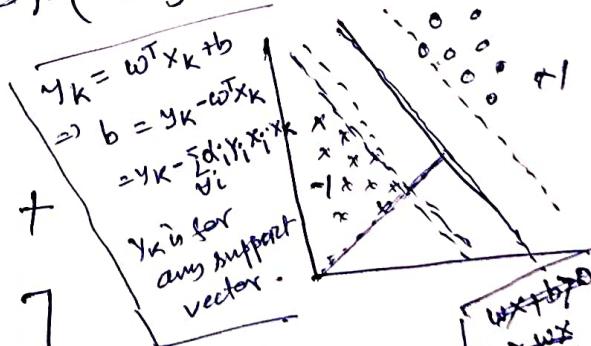
If the original set of feature vectors in lower dimensional space is not linearly separable, then we don't have the SVM because in SVM we assume that the samples are linearly separable. So if they are not linearly separable in lower dimensional space, we have to cast these feature vectors into higher dimensional space by using functions like radial basis functions, which are called kernel functions. So once we cast them into higher dimensional space, then by taking the samples in the higher dimensional space, we can design the SVM. Then for classification of unknown samples, say z , we have to cast them into same higher dimensional space. Cast them into same higher dimensional space. And after casting, we have to compute $D(z)$ and if $D(z)$ is +ve then z will be in class C_1 and if -ve then in class C_2 .

If -ve then in class C_2 .
So we need two terms:

- i) w which has been obtained by optimization as discussed.
- ii) the value of b , how since we may set it?

The value of b can be computed as:
(As this is the margin)

$$b = \frac{1}{2} \left[\min_{i | y_i=+1} \left(\sum d_i y_i (x_i \cdot x_j) \right) + \max_{i | y_i=-1} \left(\sum d_i y_i (x_i \cdot x_j) \right) \right]$$



Thus $w = \sum_{i=1}^m d_i y_i x_i$ and $b =$ are used for

classification of an unknown feature vector z .
This is how our SVM works.

\therefore SVM is a linear M/C that helps in placing the separating boundary between 2-classes or simply States where our hyperplane should be placed so that the classifier we get is more robust or more generalized.

- * Now instead of a 2-class problem, if we have a multiclass problem, then we have multiple number of SVM, tells us whether a sample belongs to class C_1 or does not in C_1 .
- ii) another SVM tells us whether a sample belongs to classes C_2 or does not in C_2 .
- \therefore we have $(P-1)$ SVM if it is a P -class problem, i.e., we have $(P-1)$ linear classifiers.

