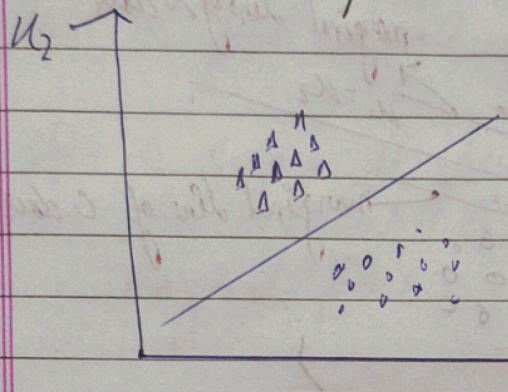
SVM & NAVIE BAYES

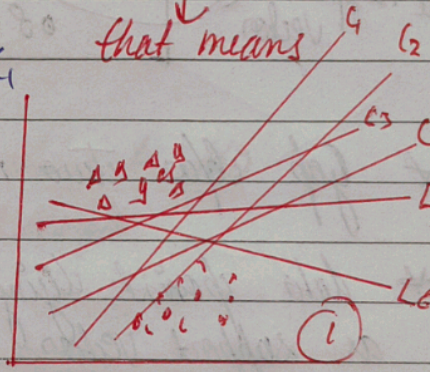
- Introduction to Support Vector Machine
  1. Support Vector classifier — classification
  2. Support Vector Regressor — Regression

① what is the problem with logistic Regression that we need to come up with SVM → SVC



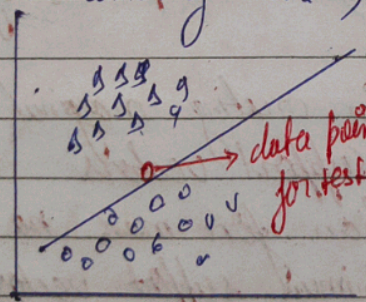
in logistic regression we find linear decision boundary which seperates two class

that means



Here all ~~lets~~ lines were seperating two classes the all ~~are~~ are possible line.

then lets say we got a datapoint of o class just above ~~a~~ decision boundary like, then it is ~~to~~ classifies as △ class



but if some line pass through the center of gap b/w two classes datapoints. it ~~s~~ would have classify it correctly.

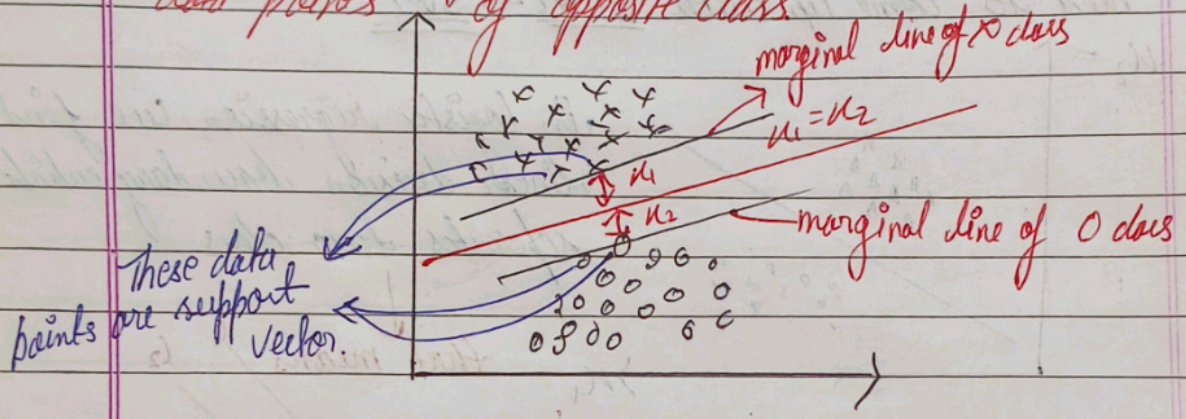
~~#~~ Logistic regression model doesn't care about the margin/ space / distribution across two classes as in img ① we doesn't all lines are possible.

in logistic Regression, when data points comes slightly left/right to decision line it is misclassify

→ in order to solve this we got SVC

* Support Vector Classifier

→ make a decision boundary such that it is equidistant from two or more different class data points of opposite class.



* Gap b/w two marginalline is margin

* data points lying on marginal line are called as support vector.

Note → Support vector are nearest datapoints of opposite class.

→ as they helps us in creating marginal line so they called as support vectors

→ There are no limitation for max no. of support vectors but minimum support vectors are required are one, One → 1+1 = 2

with help of support vector → we create marginal line/plane.

and marginal plane/line ~~help~~ helps in deciding best fit line as it will be equidistant.

This is why SVM is also called as margin classifier

Steps to get best fit boundary

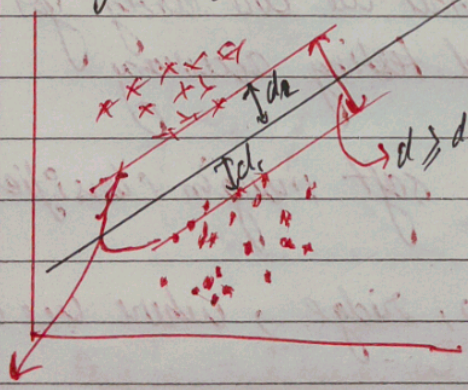
Step 1 → Find out the marginal line or plane using SVM.

Step 2 ⇒ find out all possible decision boundary

Step 3 ⇒ Among all possible decision boundary find out the one that is equidistant from both the marginal plane.

We have two more concept related to this

① Soft margin ② Hard margin

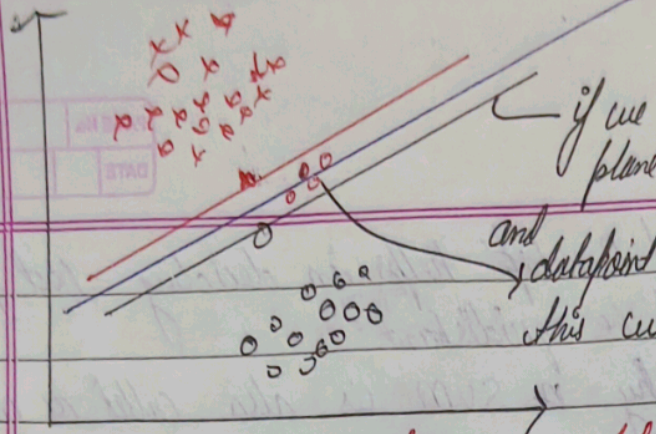


→ in this d none of data points falls

→ Hard margin → none of data points are overlapping or misclassified.

What we want is that in the distance d, none of our data points lie. But in practical it is not possible

Case 1

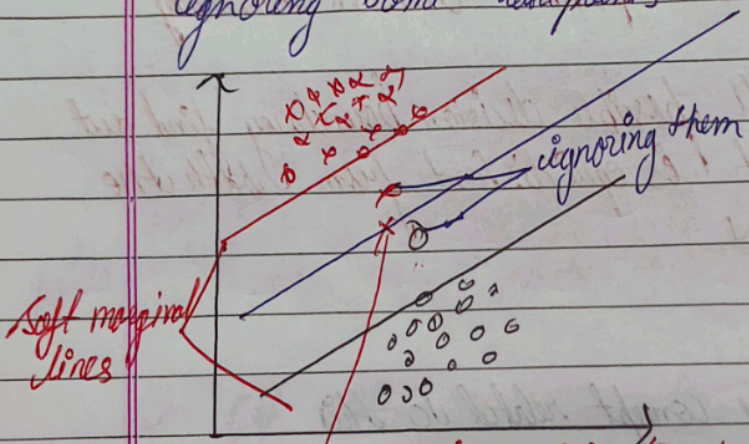


↳ if we create marginal plane ~~to~~ like this,

and datapoint in testing arise like this will misclassify.

as in this case alot of data point are clustered away from marginal plane, because only few or single data points were near to each other, which is causing a problem as ~~dist~~ distance of b/w the marginal plane/line is very short.

↳ that's where we create a soft marginal line by ignoring some data points



↳ this data point is misclassifing in train but at overall we are ready for it in order to have good testing accuracy

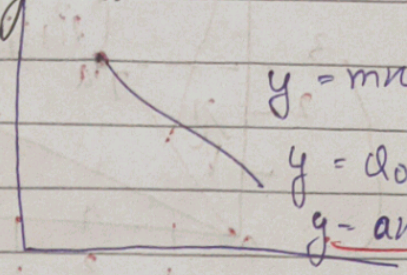
↳ it is called as soft margin classifier

This is similar to lasso, ridge, where we are introducing errors.

Maths behind Support Vector Machine

eqn of line / plane / hyper plane

① eqn of line



but when it is 2D or more than 2D $y = -\frac{a}{b}x - \frac{c}{b}$

then $y = a_0 + a_1x_1 + a_2x_2 \cdots a_nx_n$ ($m = -\frac{a}{b}$, $c = -\frac{c}{b}$)

this is also a eqn of hyperplane ②

$y = mx + c$

$m = -\frac{a}{b}$ $c = -\frac{c}{b}$

also written as ②

$y = b + w_1x_1 + w_2x_2 \cdots w_nx_n$

b = bias w = weights

$$w = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix} \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

$$w^T = [w_1\ w_2\ w_3\ w_4] \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

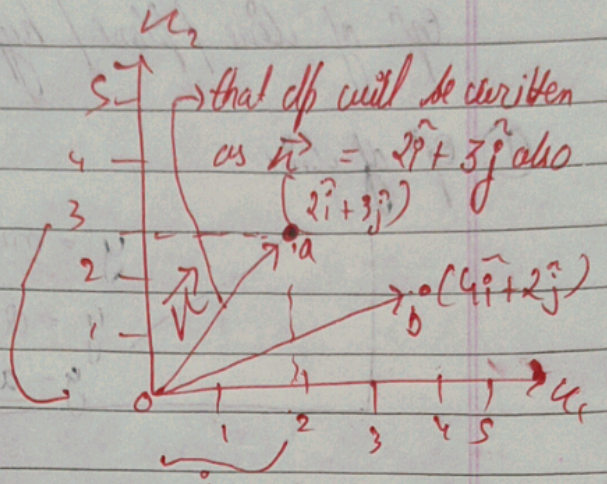
$$w^T \cdot x = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4$$

$$w^T \cdot x + b = b + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4$$

$$y = w^T \cdot b + b$$

② data points as ~~co~~ vectors

u_1	u_2
2	3
–	–
–	–
–	–



$\vec{a} = \vec{u} = 2\hat{i} + 3\hat{j}$

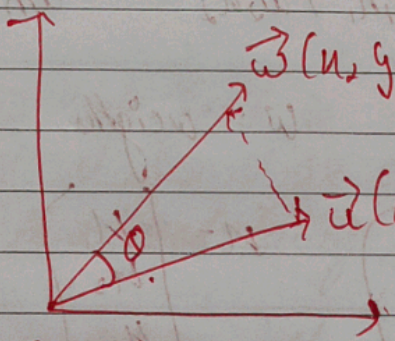
$\vec{b} = \vec{u} = 4\hat{i} + 2\hat{j}$

$\vec{a} + \vec{b} = (2+4)\hat{i} + (3+2)\hat{j}$ → Vector addition. ①

$\vec{a} + \vec{b} = 6\hat{i} + 5\hat{j}$

$\vec{a} - \vec{b} = (2-4)\hat{i} + (3-2)\hat{j} = 2\hat{i} + 1\hat{j}$ → Vector Substraction. ②

③ Dot product



$\vec{w} \cdot \vec{u} = |w| \cos\theta \cdot |u|$

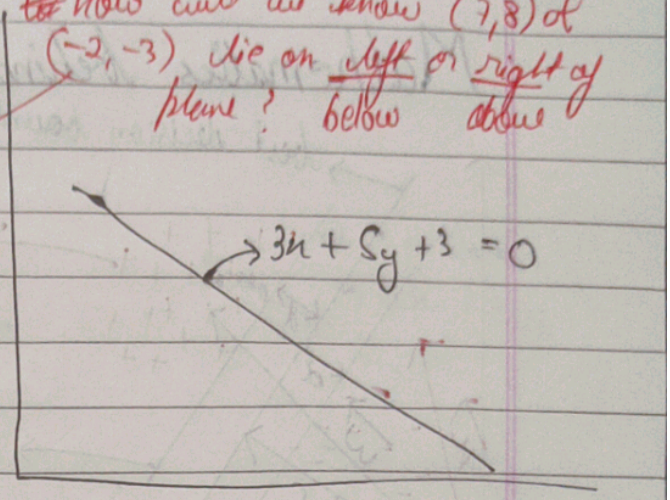
= magnitude of w · magnitude of u

$\vec{w} \cdot \vec{u}$ ⇒ ~~prod~~ projection of $\vec{w}$ on $\vec{u}$, θ is angle b/w $\vec{w}$ & $\vec{u}$

shadow

9) Cross Product

How will we know (7,8) & (-2,-3) lie on left or right of plane? below above



$3x + 5y + 3 = 0$

$3 \times 7 + 5 \times 8 + 3 = 0$

$21 + 40 + 3 = 0$

$64 > 0$

k

If $k > 0$ → lies on above/right to plane.
↳ ans we get after putting value to eq.

$3x + 5y + 3 = 0$

$3(-2) + 5(-3) + 3 = 0$

$-6 + (-15) + 3 = 0$

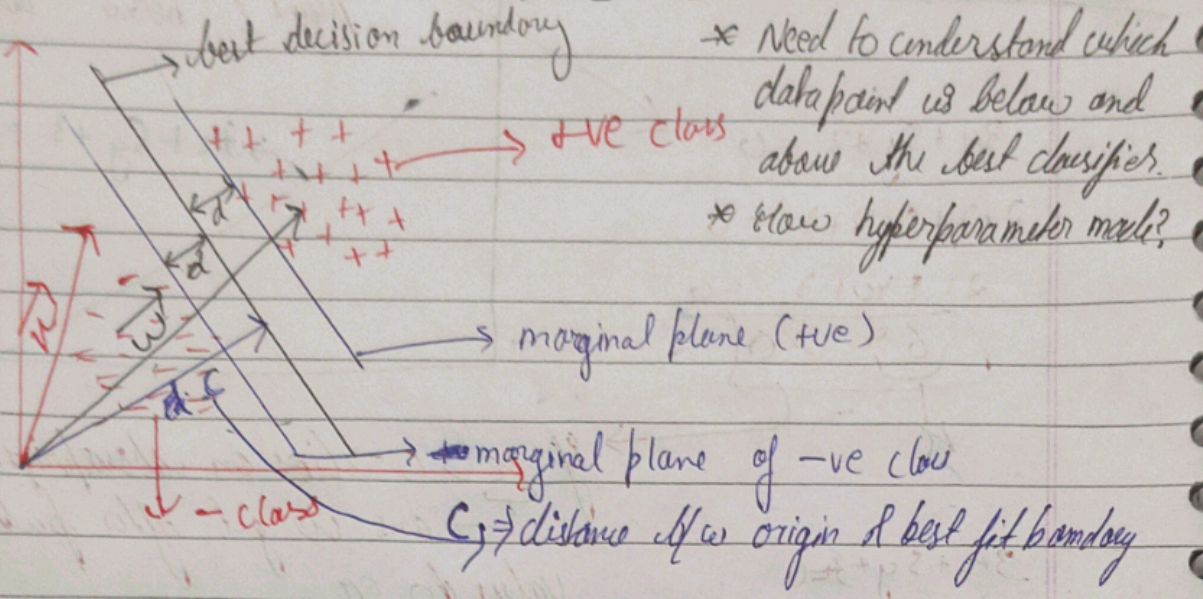
$-21 + 3 = 0$

$-18 < 0$

if $k < 0$ → lies on below/left to plane / line.

* Now we understood the basic mathematics behind sum, then now lets dive deep into deep mathematics that How actually these concepts applied on sum.

Mathematics behind S.V.C

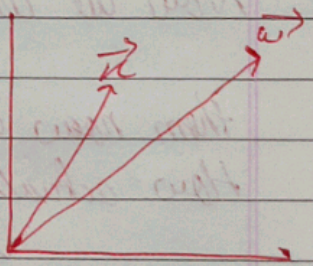


* Need to understand which datapoint is below and above the best classifier.
* How hyperparameter work?

C_1 ⇒ distance of w origin & best fit boundary

→ First we need a $\vec{w}$, which is ⊥ perpendicular to hyperplane (best decision fit line)
  • also ⊥ to both marginal plane
→ C is distance of hyperplane/optimal line from origin
→ now we try to project $\vec{x}$ on $\vec{w}$

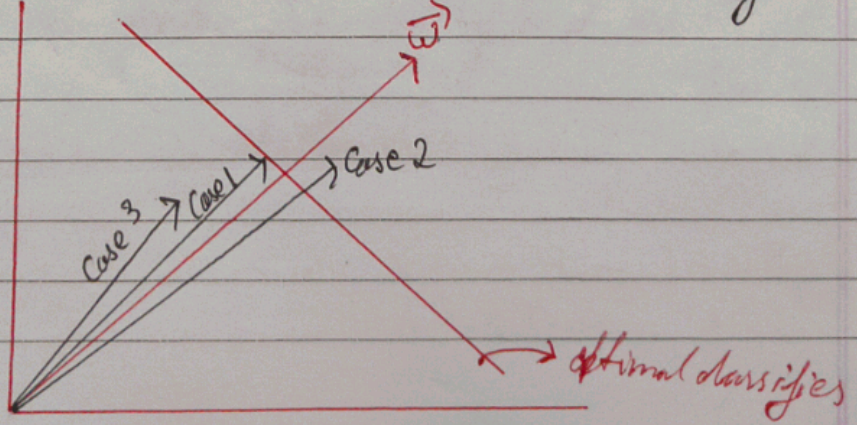
$$\vec{x} \cdot \vec{w} = |x| \cos\theta \cdot |w|$$

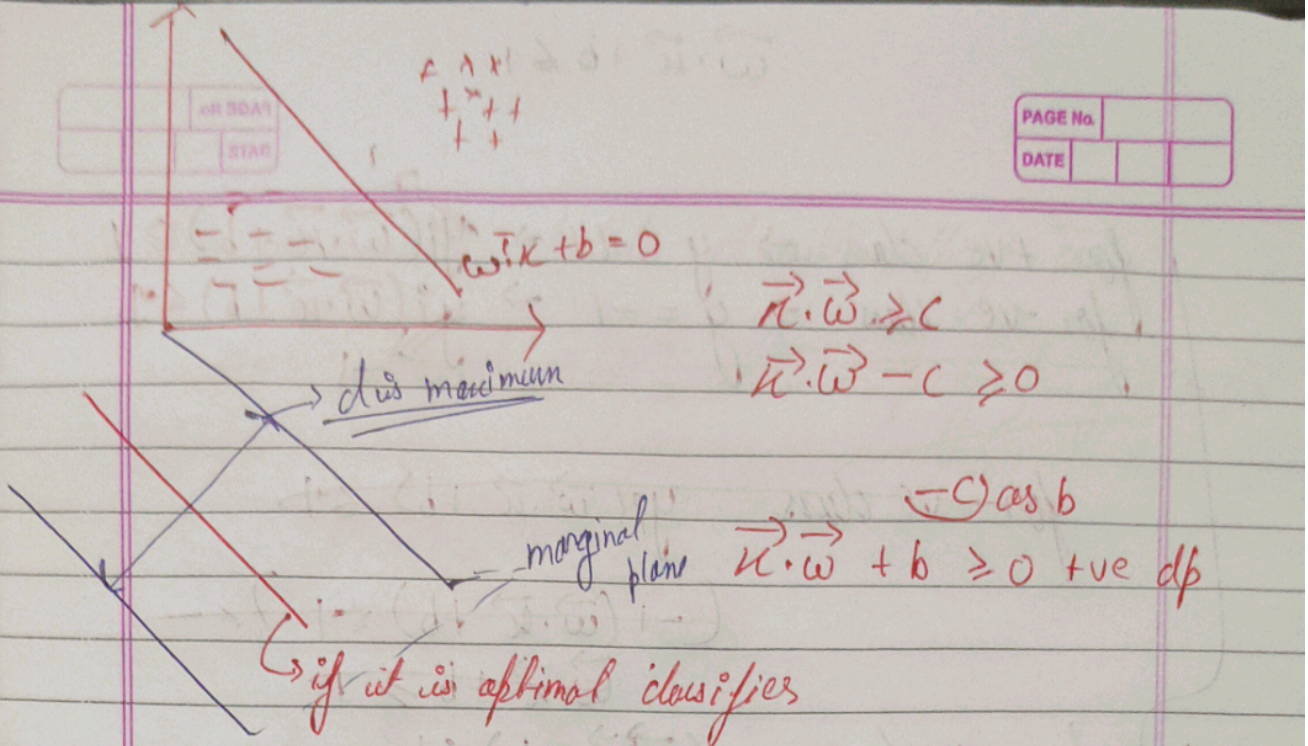


Case 1 if $\vec{x} \cdot \vec{w} = C$ {lie on best decision boundary}

Case 2 if $\vec{x} \cdot \vec{w} > C$ {lie above the decision boundary}

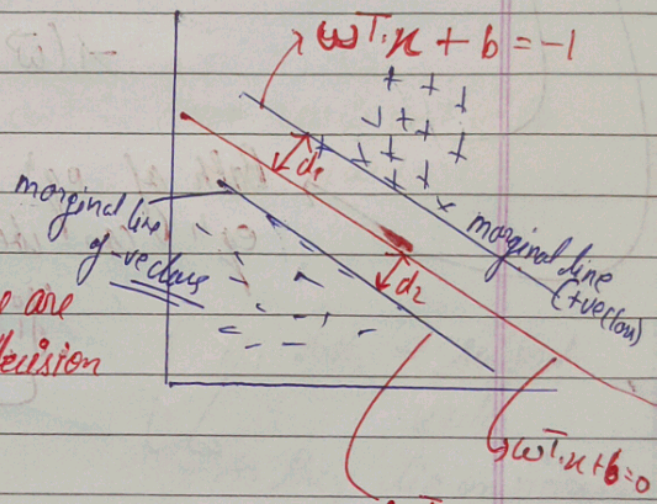
Case 3 if $\vec{x} \cdot \vec{w} < C$ {lie below the decision boundary}





y is +ve(+1) if $\vec{x} \cdot \vec{w} + b \geq 0$
y is -ve if $\vec{x} \cdot \vec{w} + b < 0$
(-1)

→ why equal (both (+1, -1))
→ why only 1 (+1, -1)



→ because both marginal line are equidistant from best decision boundary. $\{d_1 = d_2\}$

→ why only 1 ?
→ it doesn't make a difference
→ $x + y = 1$
$2x + 2y = 2$ → even if we multiply whole eq^n with other no.. line doesn't change

Now for distance d → we want to calculate distance(d) in such a way that no positive or negative point can cross marginal line

for +ve class dp's +1 $\vec{w} \cdot \vec{x} + b \geq 1$

for -ve class dp's -1 $\vec{w} \cdot \vec{x} + b \leq -1$

we want to maximized such that this constant holds true.

$\vec{w} \cdot \vec{x} + b \leq 1$

for +ve class → $y = +1$ → $y_i(\vec{w} \cdot \vec{x} + b) \geq 1$

for -ve class → $y = -1$ → $y_i(\vec{w} \cdot \vec{x} + b) \leq 1$

~~for -ve class~~ ~~$y_i(\vec{w} \cdot \vec{x} + b) \leq +1$~~

~~$(-1)(\vec{w} \cdot \vec{x} + b) \leq -1$~~ ×

~~$\vec{w} \cdot \vec{x} + b \geq 1$~~

for -ve class $y_i(\vec{w} \cdot \vec{x} + b) \leq 1$

$-1(\vec{w} \cdot \vec{x} + b) \leq 1$

↳ changing $\leq$ to $\geq$

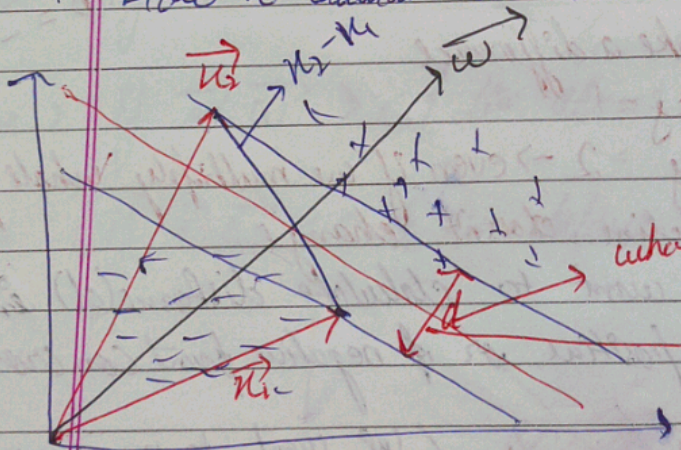
$-1(\vec{w} \cdot \vec{x} + b \geq -1$

→ Both of eq^n due to y taking $(+1, -1)$ eq^n can be re-written as:

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

↳ this will take care both class

* How to calculate d? and what is w?



Now we will calculate d with help of $\vec{w}$

~~d = project of $\vec{x_2}$ on $\vec{w}$ − projection of $\vec{x_1}$ on $\vec{w}$~~

~~$d = \vec{x_1} \cdot \vec{w} - \vec{x_2} \cdot \vec{w}$~~

~~$d = \vec{w} \cdot (\vec{x_1} - \vec{x_2})$~~

to get shortest distance we ~~xxx~~ need a unit vector $\hat{h}$ to all marginal plane / hyperplane and that unit vector is $\vec{w}$

$$d = \vec{x_2} - \vec{x_1} \cdot \frac{\vec{w}}{|w|}$$

since ~~xxx~~ $\vec{w}$ is unit vector $|w| = 1$

$$d = \frac{\vec{x_2} - \vec{x_1} \cdot \vec{w}}{|w|}$$

(3) ⟶ $$d = \frac{\vec{x_2} \cdot \vec{w} - \vec{x_1} \cdot \vec{w}}{|w|}$$

x_1 & x_2 ⇒ support vector

so they lie on marginal hyper plane.

Since they are support vector they should follow $y_i(\vec{w} \cdot \vec{x_i} + b) = 1$

for +ve class $y_i = 1$

$$w^T x_1 = 1 - b \rightarrow (1)$$

for -ve class ~~xxx~~ $y_i = -1$

$$w^T x_2 = -b - 1 \rightarrow (2)$$

putting all in ~~xx~~ eq^n (3)

$$d = \frac{b - 1 - 1 + b}{|w|} = \frac{2}{|w|}$$

This is CF = $\frac{2}{|w|}$

Cost function $cf = \frac{2}{|w|}$ by changing (w, b)

Coefficient

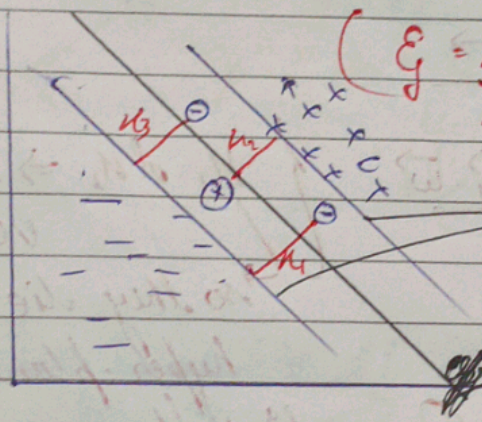
* modified cost function for hard margin support vector classifier

minimise $\frac{\|w\|}{2}$ by varying w & b

Constraints such that $y_i(w^T x_i + b) \geq 1$

max $f(u) = \min \frac{1}{f(u)}$ max converts to minimise in case of inverse

$cf = \min_{w,b} \frac{|w|}{2}$ constraint $y_i(w^T x_i + b) \geq 1$



$\xi = \xi_1 + \xi_2 + \xi_3$

sum of all dp's distance which are misclassifying.

Soft margin = no of dp's you want to sacrifise/misclassify dp's because such distinguishable scenario is not possible, there will be some overlapping data points.

Error dp's <<<< correctly classified dp's

due to this our cost function changes

* Soft margin SVC cost function:-

$$\underset{w^* b^*}{\text{minimise}} \frac{\|w\|}{2} + C \sum_{i=1}^{n} \xi_i$$ → zeta

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

C = no of misclassified dp's (Hyperparameter)

ξ_i (zeta) ⇒ is the distance of all misclassified dp's to correct margind plans.

* higher the 'd' → distance b/w hyperplane of two classes.
↓
Lower the error → in train we sacrifice some d's but while testing it gives performs good.

$$CF = \frac{\|w\|^2}{2} + C\sum_{i=1}^{n} \xi_i$$

→ also called as hinge loss

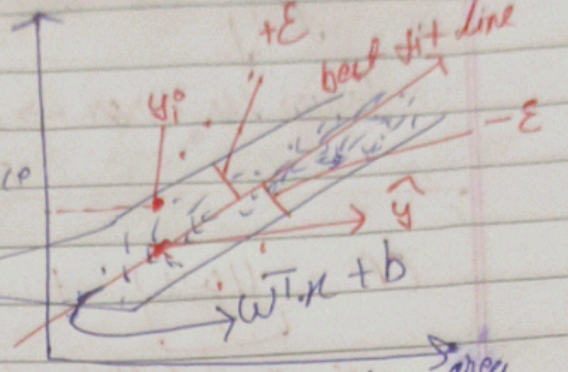
such that $y_i(w^T \cdot x + b) \geq 1$

soft margin Suppor vector Classifie CF

→ Sum error

Support Vector Regressor

SVC $\min_{w,b} \frac{|w|}{2} + C\sum_{i=0}^{n} \xi_i$



we want all our datapoint surrounding to best line.

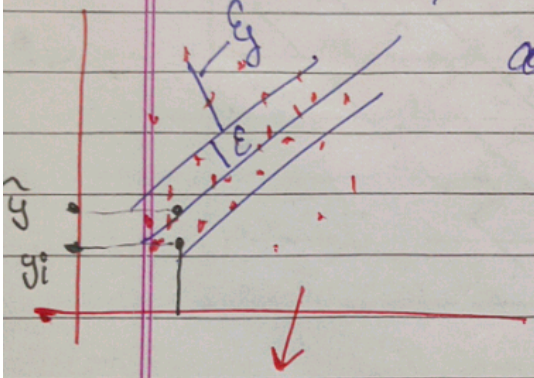
All datapoint should be on the boundary of marginal plane

all dp's should be in between

$\omega^T.x + b - \varepsilon \quad — \quad \omega^T.x + b + \varepsilon$

ε = epsilon

Initial CF = $\underset{\omega, b}{\text{minimise}} \frac{|w|}{2} + C\sum_{i=1}^{n} \xi_i$



constraints $|y_i - (\omega^T.x_i + b)| \leq \varepsilon$

y_i actual, $\omega^T.x_i + b$ → y pred.

as we want dp in b/w that

but few of dp's will be away from marginal line

↳ So we want ξ (zeta) & ε (epsilon) to be minimum. as we want all dp's to be close to best fit line.

CF = $\underset{\omega^*, b^*}{\text{minimise}} \frac{|w|}{2} + C\sum_{i=1}^{n} (\xi_i + \varepsilon)$

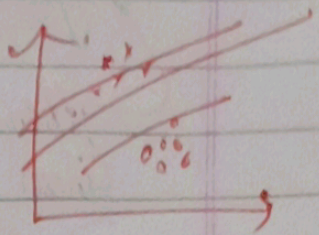
Constraint $|y_i - \omega^T.x_i| \leq \varepsilon + \xi_i$ distance b/w nearest marginal plane and datapoint

→ change dimension to higher dimension by mathematical transformation to distinguish b/w classes

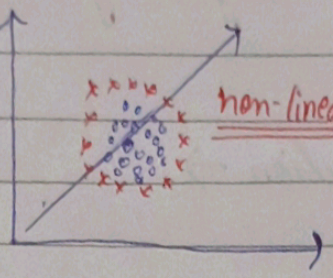
Now (C, ε) → Hyperparameters to be tuned.

SVM Kernal :→

SVM-Kernal-trick till now our data is like

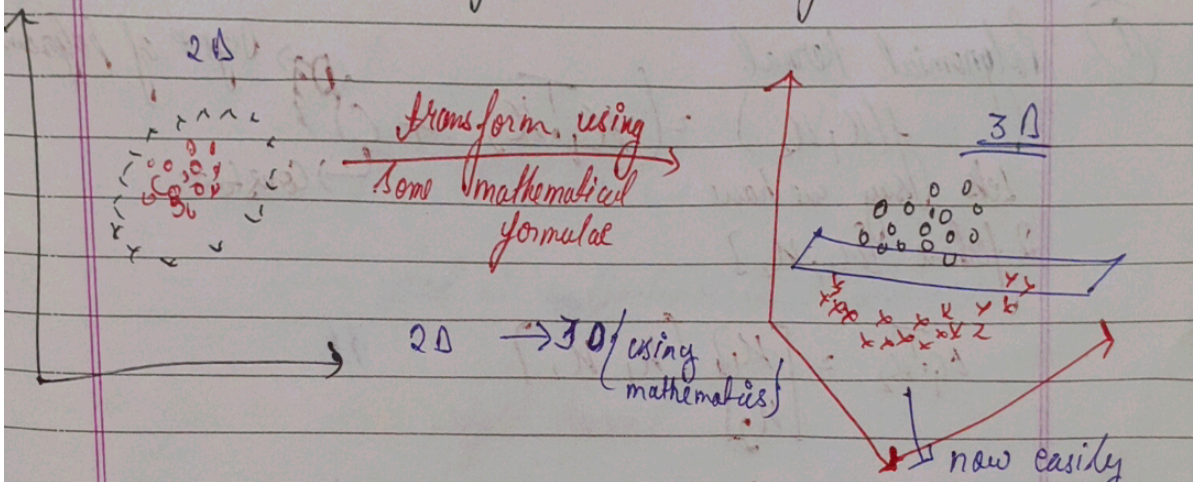


what if it looks like → if data is not linear



↳ if we try to solve it → 50% of data will be misclassified

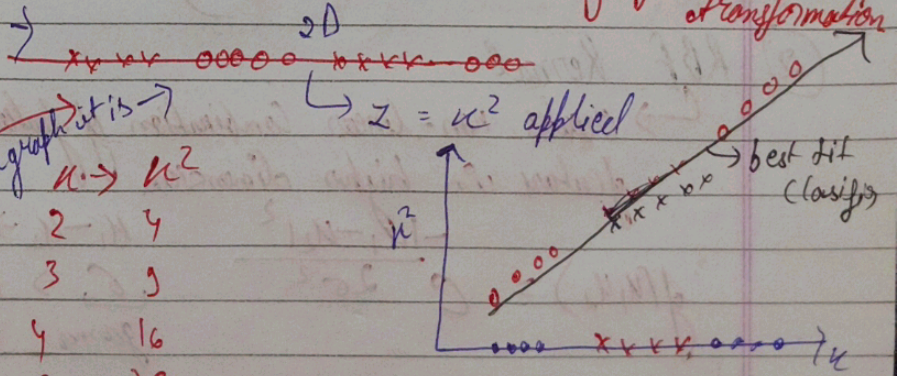
↳ for that we have got SVM kernal trick



f(x) ⇒ Kernel ⇒ kernal transformation

↳ there can be some overlapping after kernal transformation

x	y
2	0
3	0
4	1
5	0
16	0
7	0
⋮	⋮



Why called as Kernal Trick

↳ lets say we are not sending data in higher dimension we just using mathematical transformation to achieve it

↳ then using SVC classifier you can classify non linear data.

Types of Kernal function →

1. Polynomial
2. RBF (Radial Bases function)
3. Sigmoid

① Polynomial kernal

$$f(x_1, x_2) = (x_1^T \cdot x_2 + c)^d$$

d → Degree of polynomial

c → Constant

Lets say we have 2 feature $[x_1, x_2]$

$$x_1^T x_2 = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} [x_1 \; x_2]$$

$$= \begin{bmatrix} x_1^2 & x_1 x_2 \\ x_1 x_2 & x_2^2 \end{bmatrix}$$

$x_1 x_2 \Rightarrow x_1 x_2, x^2, x_1 x_2, x_2^2$

② RBF Kernal

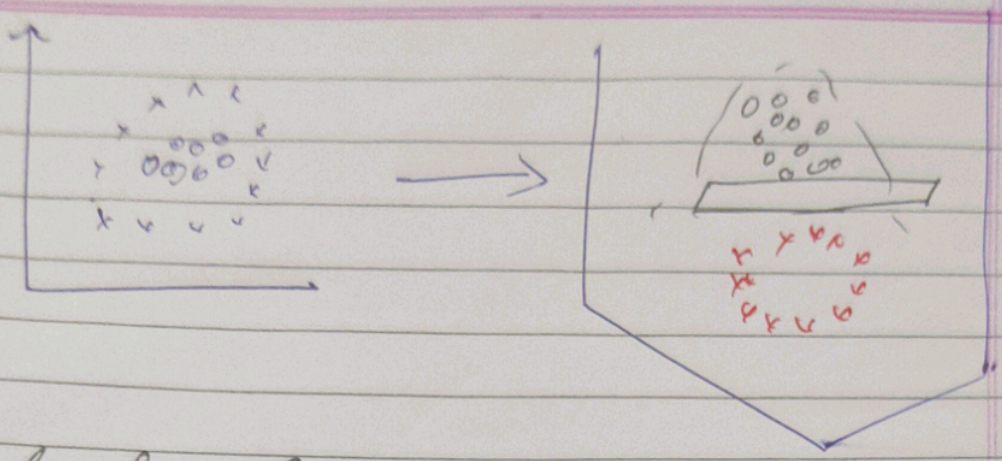
↳ creates non-linear combination of feature to bring your feature in higher dimension

$$f(x_1, x_2) = e^{\frac{-|x_1 - x_2|^2}{2\sigma^2}}$$

$x_1 - x_2$ → euclidean distance

σ → Hyperparameter

gamma



(3) Sigmoid Kernal

$$f(u,y) = \frac{1}{1+e^{-u}}$$

* Bessel kernal →

$$f(u,y) = \frac{J_{v+1}(\sigma \|u-y\|)}{\|u-y\|^{-n(v+1)}}$$

* Anova kernal

$$f(u,y) = \sum_{k=1}^{n} \exp\left(-\sigma (u^k - y^k)^2\right)^d$$

→ used in multidimensional regression.

* How to chose right kernal
  - → build on all
    - → By Hyperparameter
      - → get best one