Why to use SVC over Logistic regression?

Non-linear Boundaries: SVC can handle non-linear decision boundaries using kernel tricks.
Outlier Robustness: SVC focuses on support vectors, making it more robust to outliers.
Small Datasets: SVC performs well on smaller datasets by emphasizing critical points.
Margin Maximization: SVC maximizes the margin between classes, enhancing generalization.
Complex Relationships: SVC is suited for complex, non-linear problems requiring precise decision boundaries.

Definition of SVC
A Support Vector Classifier (SVC) is a supervised machine learning model used for classification tasks. It aims to find the optimal hyperplane that separates data points of different classes by maximizing the margin between them. SVC uses support vectors, which are the critical data points nearest to the hyperplane, to define this separation. For non-linear classification, it employs kernel functions to transform data into higher-dimensional spaces. SVC is effective in high-dimensional spaces and memory-efficient, but can be computationally intensive for large datasets and sensitive to noise and overlapping classes.

Support Vector Machine (SVM) is a supervised machine learning algorithm used for classification and regression tasks. It is particularly powerful in high-dimensional spaces and for situations where the number of dimensions exceeds the number of samples. Here's a comprehensive overview of SVM:

Key Concepts:
Hyperplane: In SVM, a hyperplane is a decision boundary that separates different classes in the feature space. In two dimensions, this is a line; in three dimensions, it's a plane; and in higher dimensions, it's a hyperplane.

Support Vectors: These are the data points that are closest to the hyperplane and influence its position and orientation. The SVM algorithm tries to find the hyperplane that maximizes the margin between these points and the decision boundary.

Margin: This is the distance between the hyperplane and the nearest data point from either class. SVM aims to maximize this margin, making it a maximum margin classifier.

Types of SVM:
Linear SVM: Used when the data is linearly separable, meaning a straight line (or hyperplane in higher dimensions) can separate the data points of different classes.

Non-linear SVM: Used when the data is not linearly separable. This involves using kernel functions to transform the data into a higher-dimensional space where a linear separator can be found.

Mathematical Formulation of SVC

The mathematical formulation of a Support Vector Classifier (SVC) involves finding the optimal hyperplane that maximizes the margin between two classes. Here's a description in words:
1. Objective: Minimize the magnitude of the normal vector to the hyperplane, which is equivalent to maximizing the margin between the classes. This is represented as minimizing 1/2w2, where w is the normal vector to the hyperplane.
2. Constraints: Ensure that all data points are correctly classified with a margin of at least 1. For each data point, the constraint can be written as:
o If the data point belongs to the positive class (yi=+1), it should lie on one side of the hyperplane, meaning wxi+ b>=1.
o If the data point belongs to the negative class (yi=1), it should lie on the other side of the hyperplane, meaning wxi+<=1.
Combining these, the constraint is: yi(wxi+b)>=1 for all data points i.

Soft Margin in SVC (see its mathematical formulation)

A soft margin in Support Vector Classification (SVC) allows some misclassification of data points to achieve a balance between maximizing the margin and minimizing classification errors. It introduces slack variables to permit certain points to be within or outside the margin boundaries. The regularization parameter C controls this trade-off: a higher CCC value prioritizes fitting the training data closely, while a lower C value allows for more misclassifications, aiming for better generalization. This flexibility makes soft margin SVC suitable for handling non-linearly separable and noisy data.

Support Vector Regression (see mathematical formulation)

Support Vector Regression (SVR) is a machine learning technique for regression tasks. It identifies a hyperplane in a high-dimensional space that best fits the data points, aiming to minimize error within a specified tolerance (epsilon). SVR uses kernel functions to map input data into higher dimensions, allowing it to capture complex relationships between features and target values. Unlike traditional regression models that focus on minimizing errors globally, SVR focuses on minimizing deviations only within a specified margin around the predicted values. It's effective for datasets with non-linear relationships, offering robust performance in various domains such as finance, image processing, and bioinformatics.

Goal: SVR aims to predict a value y (like predicting house prices) based on input data x (like number of bedrooms, location, etc.).
Method: Instead of finding a simple line or curve to fit all the data points closely (like in standard linear regression), SVR finds a margin (zone) around the predicted line or curve where predictions are allowed to be a bit wrong. This margin is controlled by a parameter called epsilon.
Training: SVR finds this line or curve by considering only the data points near the margin (these are called support vectors). It tries to find the line or curve that fits these support vectors as closely as possible while still respecting the margin.
Prediction: Once trained, SVR can predict the value y for new input data x by using the line or curve it found during training.

SVM Kernel

A Support Vector Machine (SVM) kernel transforms data into higher-dimensional spaces to find optimal hyperplanes for classification or regression. Kernels like linear, polynomial, radial basis function (RBF), or sigmoid map data nonlinearly, enhancing SVM's ability to handle complex relationships. Linear kernel computes inner products directly in original space. Polynomial kernel computes inner products with polynomial expansion, useful for non-linear data. RBF kernel computes similarity using Gaussian function, effective for clustering and high-dimensional data. Sigmoid kernel uses hyperbolic tangent function to map data, suitable for neural networks. Choosing the right kernel balances model complexity and performance, crucial for SVM's effectiveness in diverse applications.

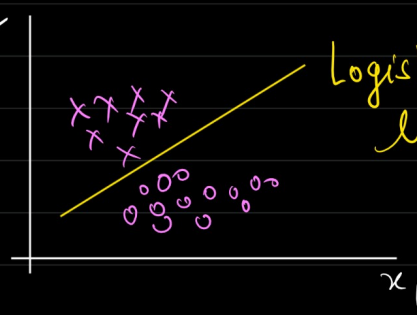
# Support Vector machines

① Support Vector classifier ⟶ Classification
② Support Vector Regressor → Regressor

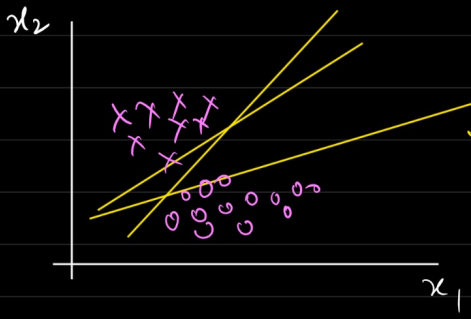
SVC, SVR

## ① Support Vector classifier

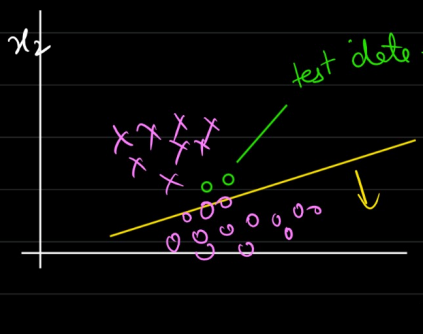
In Log Reg



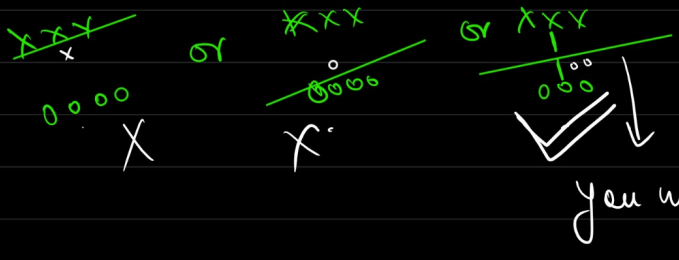

Logistic reg model finds linear decision boundary which separates two classes.



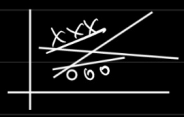

→ Any line can be decision boundary in Logistic Reg.



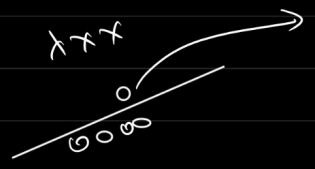

→ Logistic regression decision boundary is prone to error.




* Logistic regression model doesn't care about margin / space across two class.



* if the data points changes slightly, Log reg model will give error.

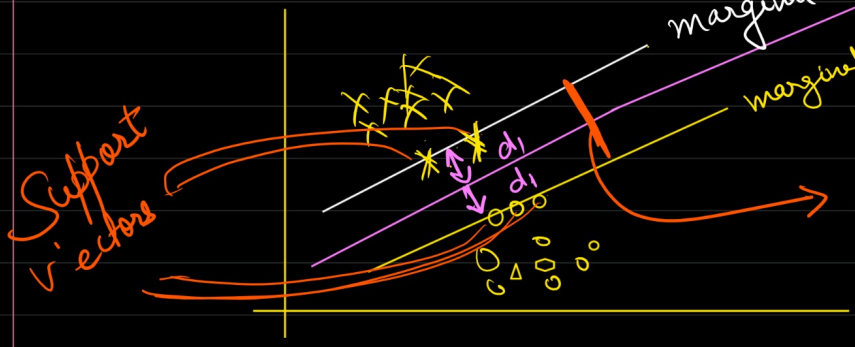


It will belong to other class.

Hence Log reg is prone to error.

## * Support Vector classifier


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




→ best fit line decision boundary

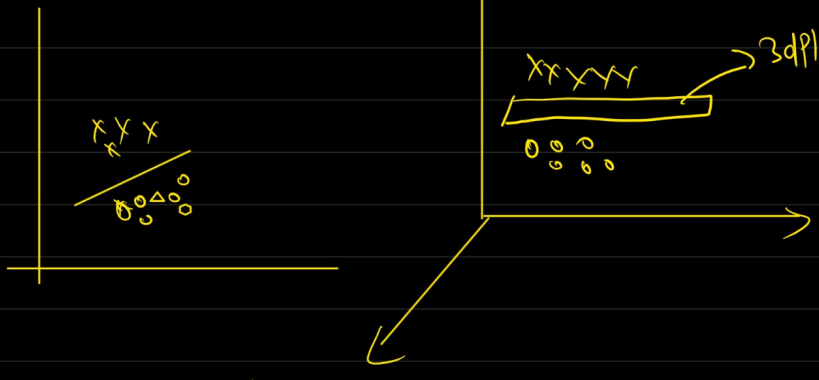
margin → the gap between the nearest datapoints of both class.

(nearest dp's can be multiple)

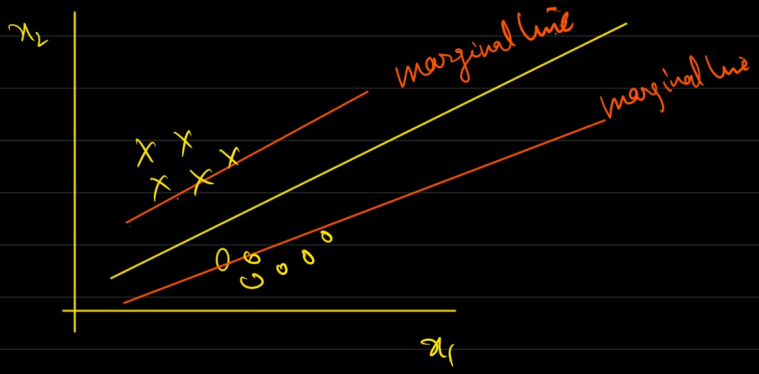
↓

Support Vectors.

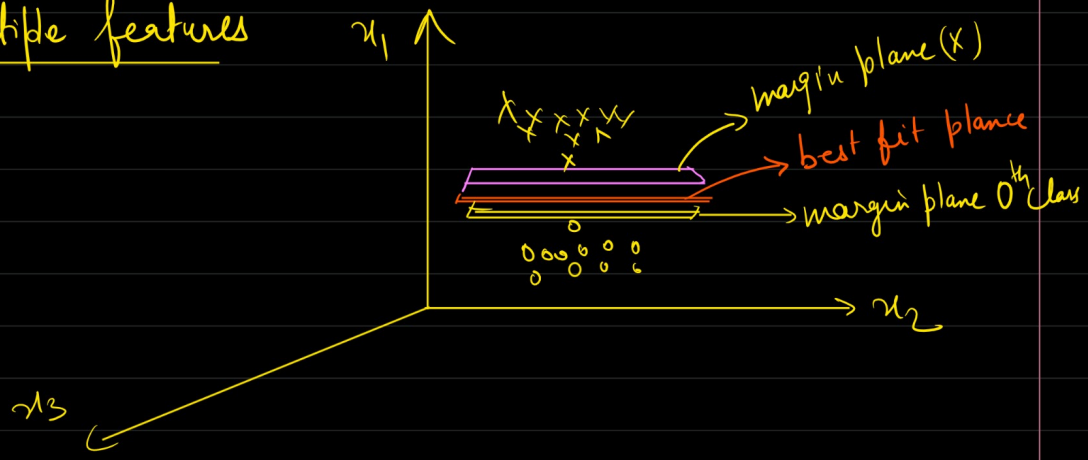
Log regression

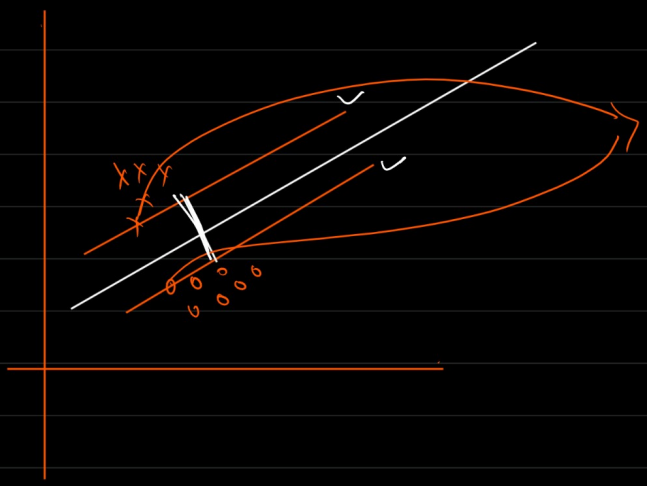


## Similarly in SVM




## In case of multiple features



**Support vectors**

↓

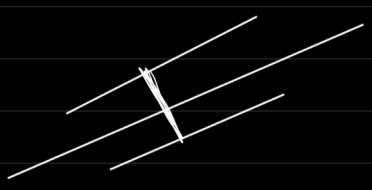
nearest datapoint of either class.

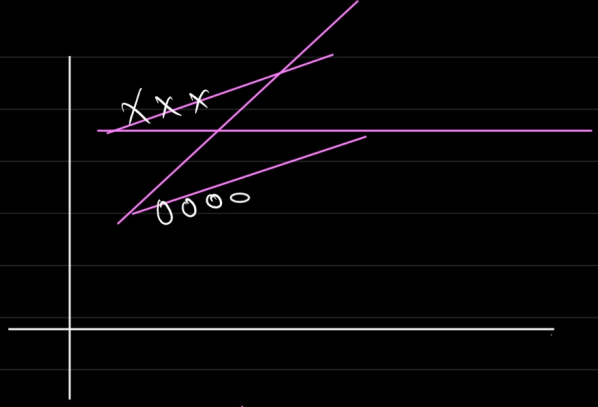
It is called support vectors because these nearest data points helps to create the right classifier

⇒ There are no limitations on support vectors.

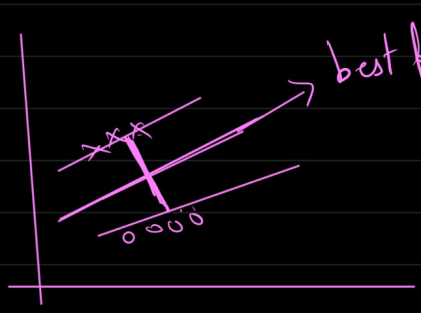
* Min no of support vector (1+1) = 2

* Support vectors will always choose central lines passing exactly through the centre; that's why Support Vector machine is also called margin classifier.

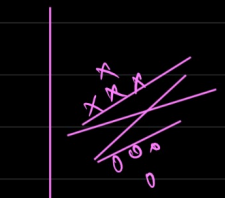




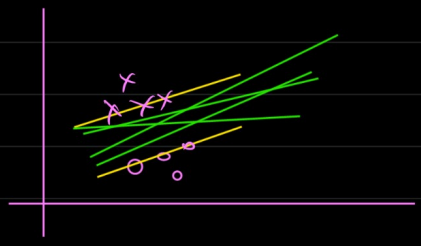
Always

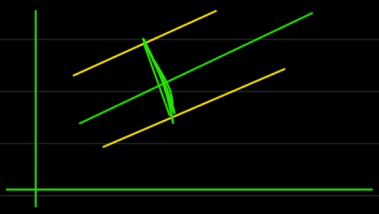


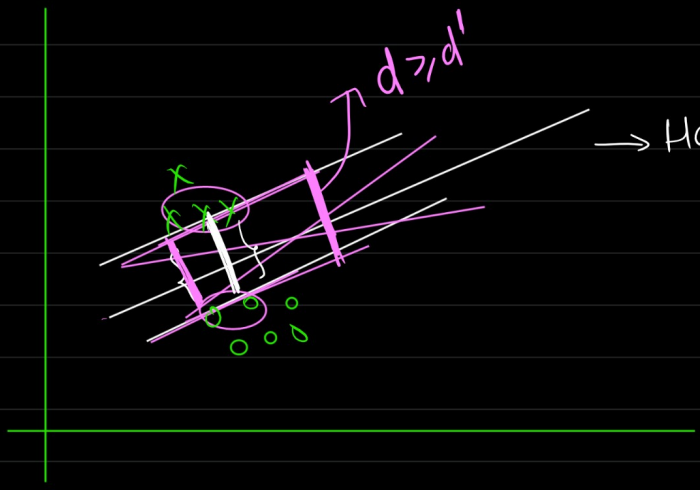
best fit line.



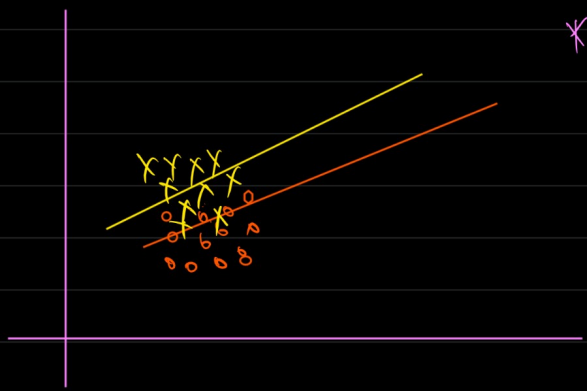
Step-1 - find out all possible classifier
Step-2 Find out margin of each classifier.
Step-3 Select which has maximum margin as classifier



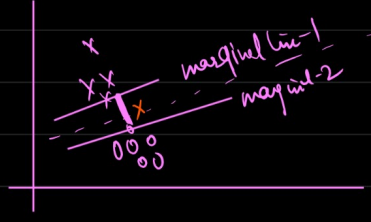




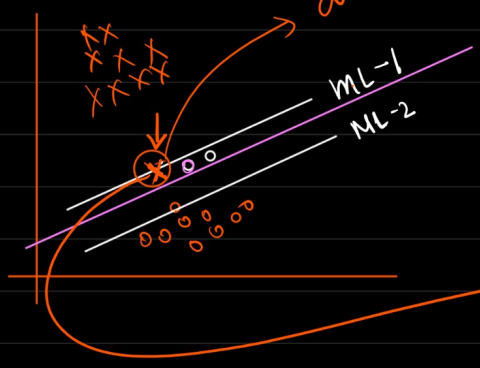

→ Hard margin ⇒ none of the data points are overlapping / misclassified



* Soft margin
⇓
Some data points are misclassified (Error)

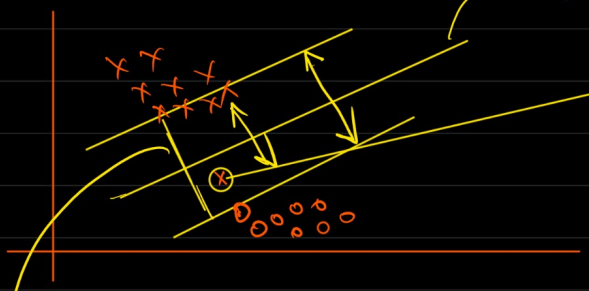


Scen-1




due to this one X, ML-1 is very close to ML-2, that's why best fit line is also very close to class 0.

→ This dp is causing overfitting.



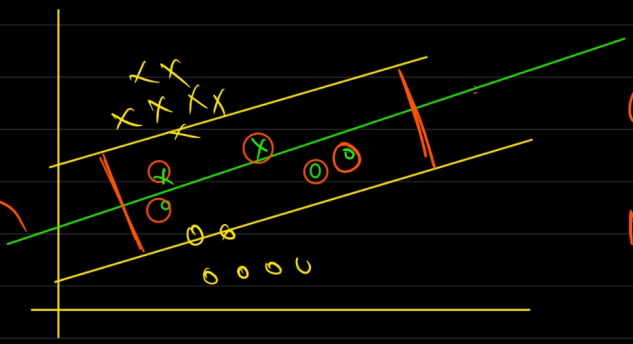

→ misclassified | error. during training.

but the best fit line is more generic.

→ Soft margin = Some points are misclassified

→ you are ready to misclassify some of the data points in order to have good testing accuracy, its a soft margin classifier.

This parameter is 'C' (regularisation parameter)



C = 5

↓

No of datapoints, you sacrificed so that best fit line is not overfitting.

→ linear Support Vector classifier.

# Support Vector classifier in depth Maths

## ① Eqn of line, plane, hyperplane.

$y = mx + c$

$y = \theta_0 + \theta_1 x_1$

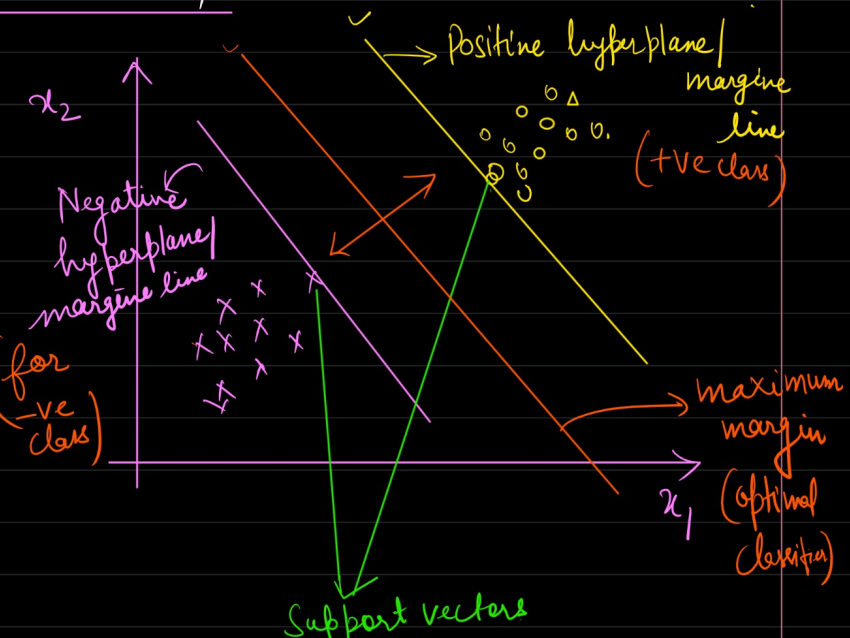
$ax + by + c = 0$

$by = -ax - c$

$y = \boxed{-\frac{a}{b}} x - \boxed{\frac{c}{b}}$

$m$

$y = mx + c$




## in more than 3d

$y = \theta_0 + \theta_1 x_1$

$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots \theta_n x_n$ → hyperplane

$y = b + w_1 x_1 + w_2 x_2 + w_3 x_3 + \cdots w_n x_n$

b — bias
w — weights

$y = b + w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4$

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

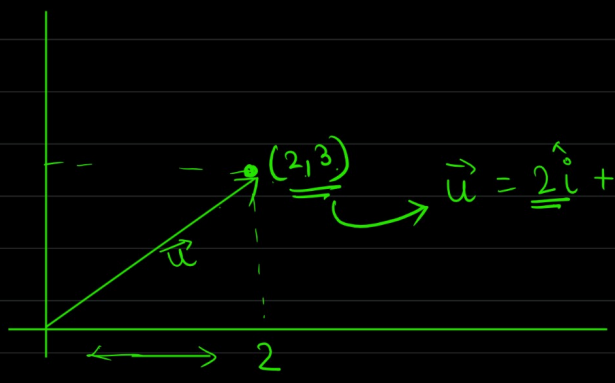
$$w^T = \begin{bmatrix} w_1 & w_2 & w_3 & w_4 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \Rightarrow w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4 + b$$

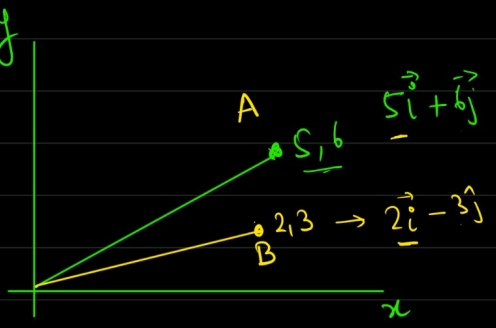
$w^T$ · $x$

$$y = w^T x + b \quad (y = mx + c, \; ax + by + c = 0)$$

$$\Downarrow$$

$$w^T x + b = 0$$

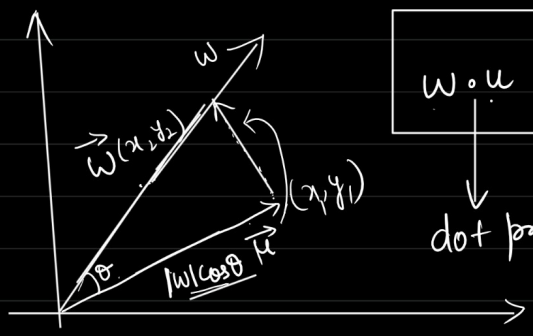
② 


③ Vector subtraction $\vec{A} - \vec{B}$

$$(5-2)\hat{i} + (6-3)\hat{j} = 3\hat{i} + 3\hat{j}$$

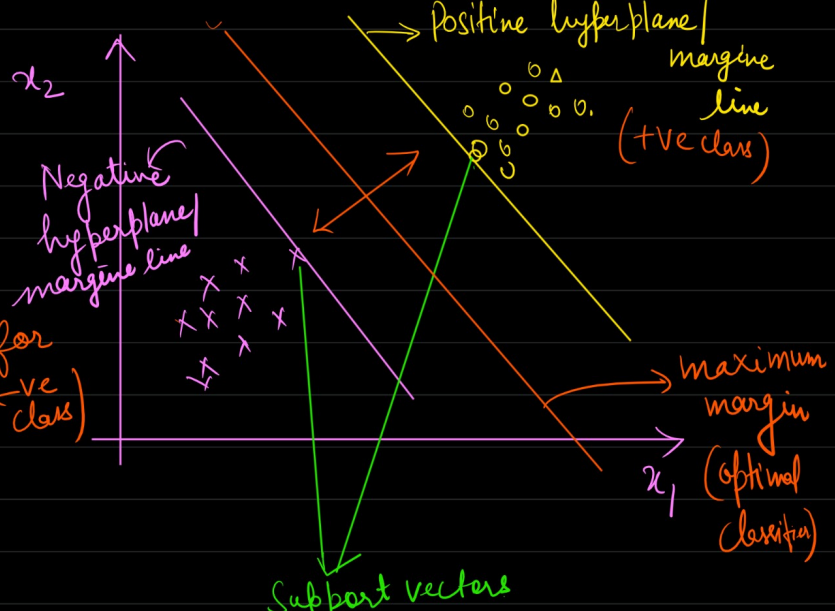
④ Dot product of vectors.

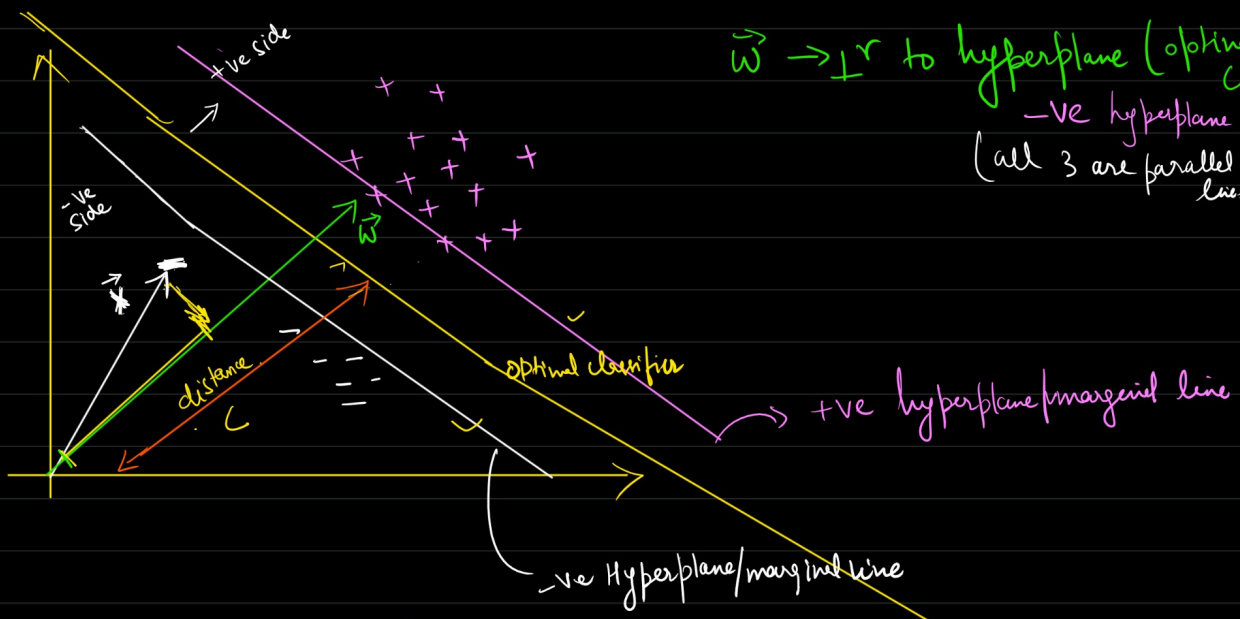



magnitude of w, magnitude of u

$$w \cdot u = |w| \cos\theta \cdot |u|$$

dot product means projection of $\vec{u}$ on $\vec{w}$






$\vec{w} \rightarrow \perp^r$ to hyperplane (optimal classifier)

-ve hyperplane / +ve hyperplane

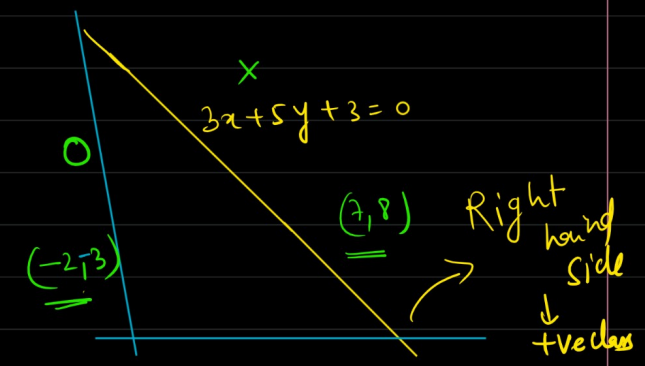
(all 3 are parallel lines)

Project is dot product.

$\vec{x} \cdot \vec{w} = (c)$ distance (the point lies on decision boundary)

$\vec{x} \cdot \vec{w} > (c)$ distance (positive class)

$\vec{x} \cdot \vec{w} < (c)$ distance (negative class)



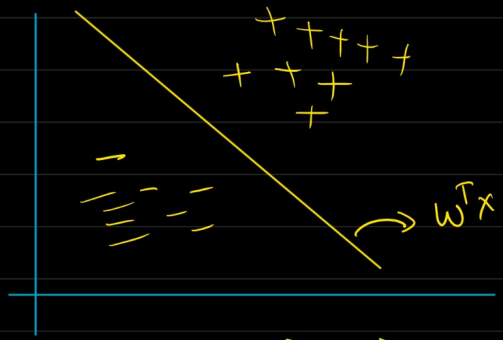

$(7,8) \Rightarrow 3\times7 + 5\times8 + 3 > 0$

$(-2,-3) \Rightarrow 3\times-2 + 5\times-3 + 3 < 0$

↓

-ve side (left side)

-ve class.




$\vec{x} \cdot \vec{w} \geq c$

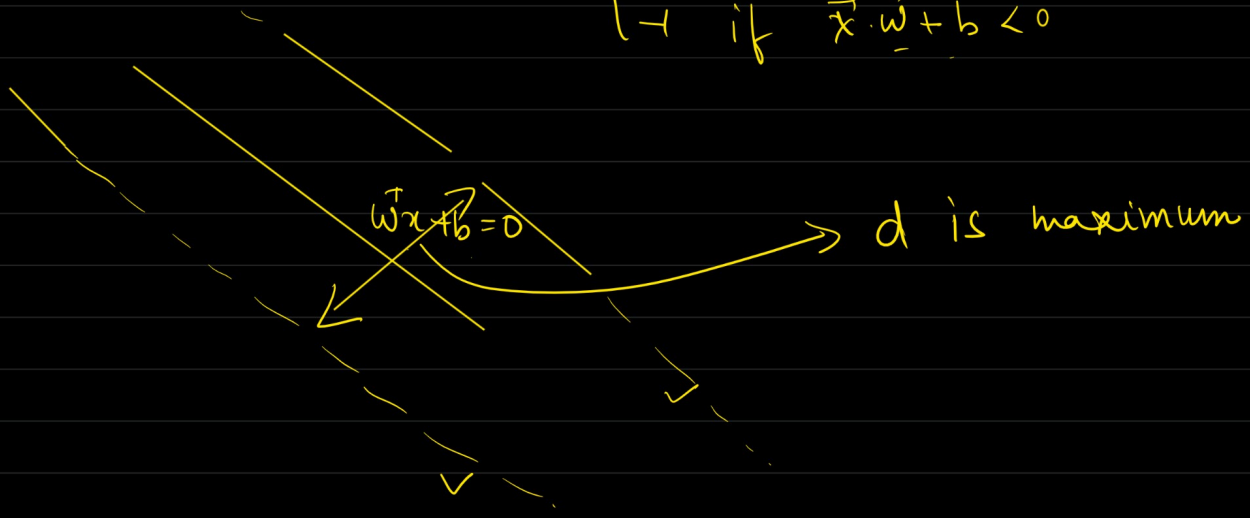
$\vec{x} \cdot \vec{w} - c \geq 0$

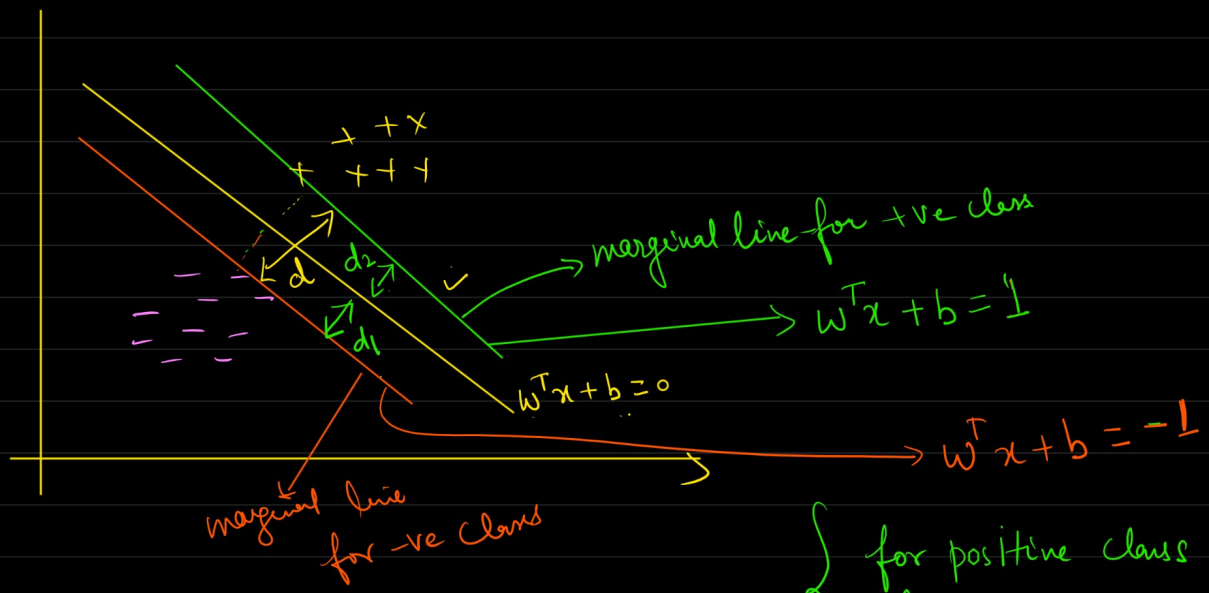
$-c$ as $b$

$\vec{x} \cdot \vec{w} + b \geq 0 \longrightarrow$ +ve point

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

$$y = \begin{cases} +1 & \text{if } \vec{x} \cdot \vec{w} + b \geq 0 \\ -1 & \text{if } \vec{x} \cdot \vec{w} + b < 0 \end{cases}$$



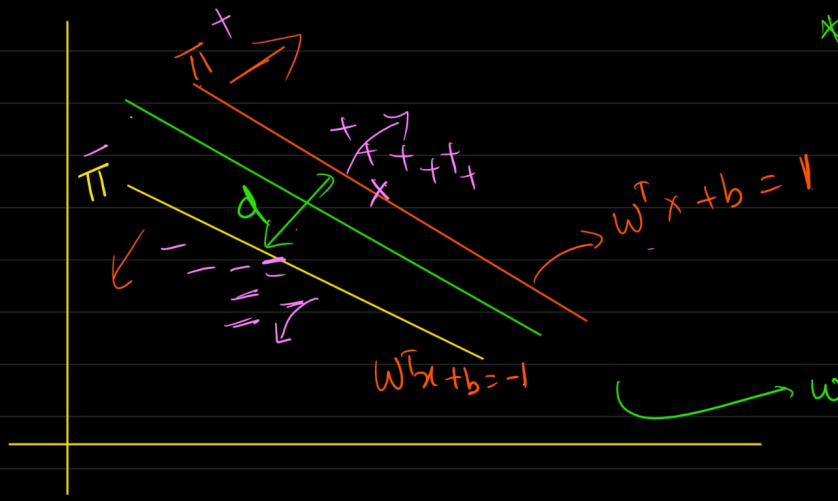

{ for positive class +1
for negative class −1

* Why equal (both 1)? → $d_1 \& d_2$ should be equidistant (optimal line should pass through center) of margin.

* Why only 1? → It doesn't make a difference

Both will be same line ⟵ $\begin{cases} 2x + y = 1 \\ 2x + y = 2 \end{cases}$

→ even if we multiply the whole equation with some other number the line doesn't change




* we want calculate distance (d) such that no positive or negative point can cross the margin line

+1 for +ve class dps → $\vec{w}^T \cdot \vec{x} + b \geq 1$

-1 for -ve class datapoints → $\vec{w}^T \cdot \vec{x} + b \leq -1$

we want to maximize 'd' such that this constraint holds true

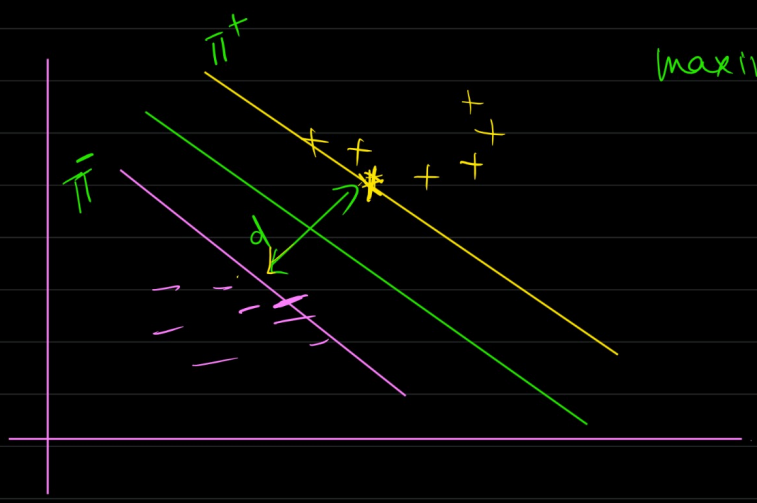
for +ve class → $y_i (\vec{w}^T \cdot \vec{x} + b) \geq 1$

($y_i$ = +1, $\vec{w}^T \cdot \vec{x} + b$ = +ve)

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

$= +1 (\vec{w}^T \cdot \vec{x} + b) \geq +1$

$= y_i (\vec{w}^T \cdot \vec{x}_i + b) \geq 1$



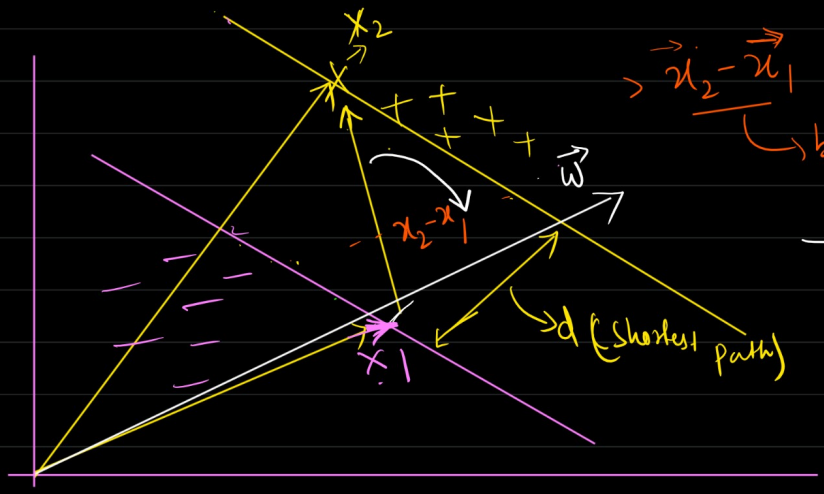

maximise d

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

for support vector

$y_i(\vec{w}^T\vec{x} + b) = 1$

equality because Support vectors falls on marginal hyperplane.




$\rightarrow \vec{x}_2 - \vec{x}_1$

↳ b/w vector $x_2, x_1$

→ To get shortest distance we need a unit vector $\perp^r$ to all marginal hyperplane

→ Projection of $\vec{x}_2 - \vec{x}_1$ on unit vector $\vec{w}$ to get d

$$d = (\vec{x}_2 - \vec{x}_1) \cdot \frac{\vec{w}}{\|w\|}$$

$$\frac{x_2 \cdot \vec{w} - \vec{x}_1 \cdot w}{\|w\|} \quad \text{—①}$$

($x_1$ & $x_2$ are support vectors, they lie on marginal hyperplane)

Since $x_1, x_2$ are support vectors, it should follow

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

for +ve class $= y_i = 1$ for $x_1$

$$1 \times (w^T x_1 + b) = 1$$

$$\vec{w} \vec{x}_1 = 1 - b \quad \text{—②}$$

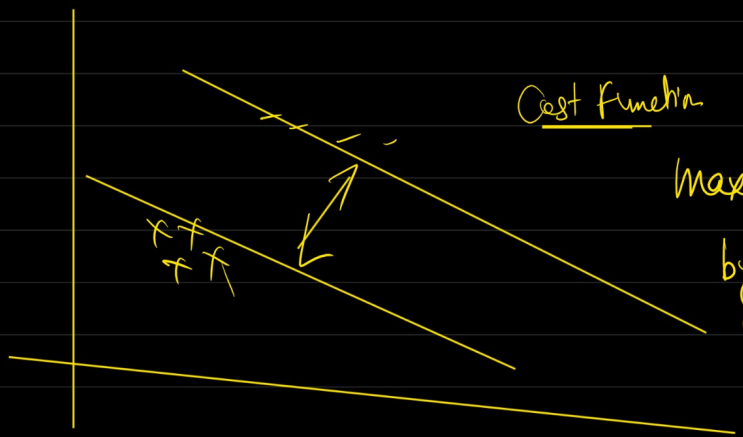
for -ve class $-1(w^T x_2 + b) = 1$

$$\vec{w}^T \cdot \vec{x}_2 = -b - 1 \quad \text{—③}$$

putting eqn ② & ③ in ①

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

$$\underset{(w,b)}{\text{maximise}} \ \frac{2}{\|w\|} \quad \text{such that} \quad y_i(\vec{w}^T \cdot \vec{x} + b) \geq 1$$



Cost function

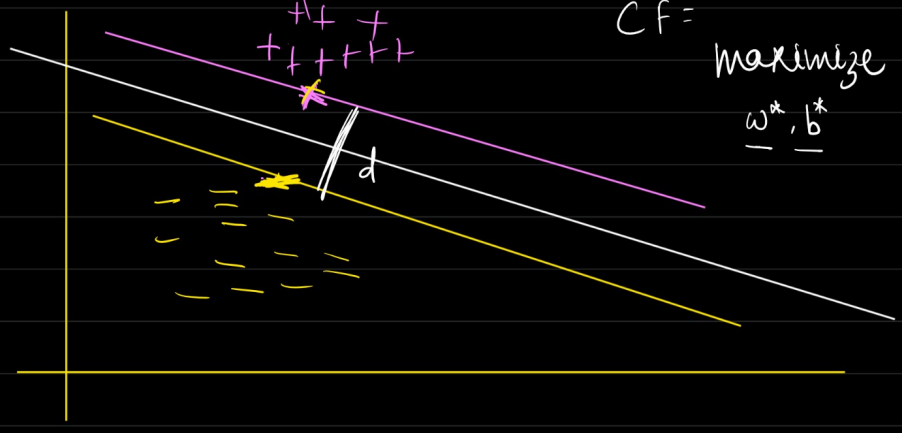
maximize $\frac{2}{\|w\|}$

by changing $w, b$ (intercept / coeff)

* Modified cost fn for Hard margin SVC

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

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




CF =

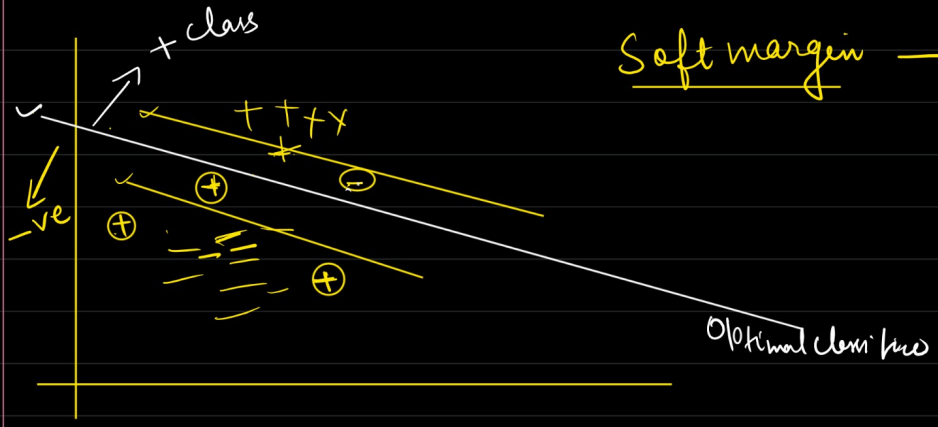
maximize $\underset{w^*, b^*}{} \frac{2}{\|w\|}$

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

$\max f(x) \Leftrightarrow \min \frac{1}{f(x)}$

$$CF = \min_{w^* b^*} \frac{\|w\|}{2}$$

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




**Soft margin** — No of datapoint you want to sacrifice / error datapoint / misclassified datapoint because such distinguishable scenario is not possible, there will be some overlapping datapoints.

→ Error dp <<< correctly classified dp's

## CF for soft margin SVC

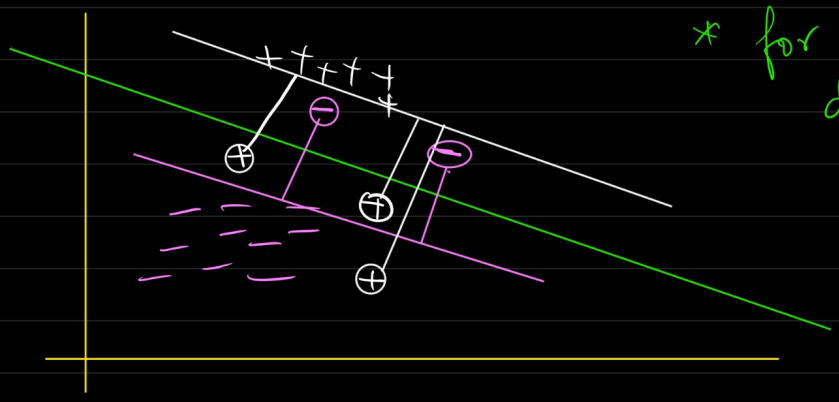
$$\text{minimise}_{w^* b^*} \frac{\|w\|}{2} + C \sum_{i=1}^{n} \xi_i \quad \text{(Zeta)}$$

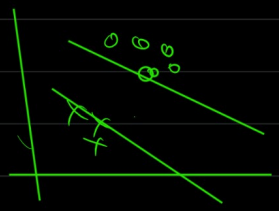
Such that $y_i (w^T x_i + b) \geq 1$

C → no of misclassified dp's (hyperparameter)

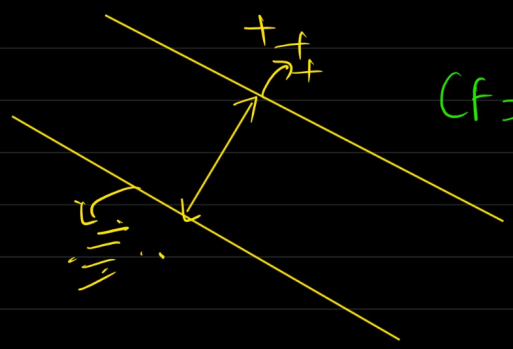
$\xi$ → is the distance of all misclassified dp's to correct marginal plane.

* for all correctly classified dp's ⇒ hard margin $\xi = 0$





→ higher the d ⇒ distance b/w the hyperplanes of two class, lower the error



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

→ Hinge loss

$\frac{\|W\|}{2}$ → SVM error

$\sum_{i=1}^{n} \xi_i$ → classification error.

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

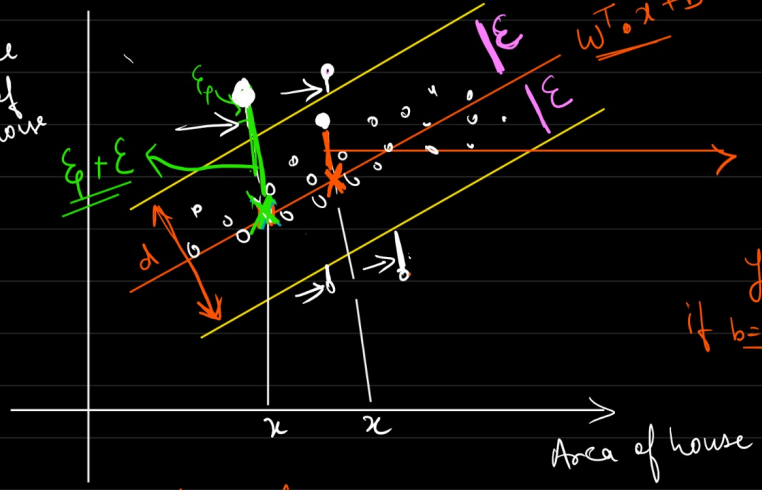
$$C = \frac{1}{\lambda}$$

## Support Vector Regressor

SVC

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

hinge loss




→ We want all our dp's surrounding best line.

⇒ All the data points should be in the boundary of marginal plane.

All the dp's $w^T x + b + \varepsilon$ —— $w^T x + b - \varepsilon$

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

as less as possible $\xi_i$ $\varepsilon$

Constraints $|y_i - w^T x_i| \leq \varepsilon + \xi_i$

$y_{act} - y_{pred}$ should be lesser than $\varepsilon + \xi_i$

* All the dP's will not be in between marginal plane.

$C \sum_{i=1}^{n} \xi$ — Zeta
$\varepsilon$ — Epsilon
→ hyperparameter

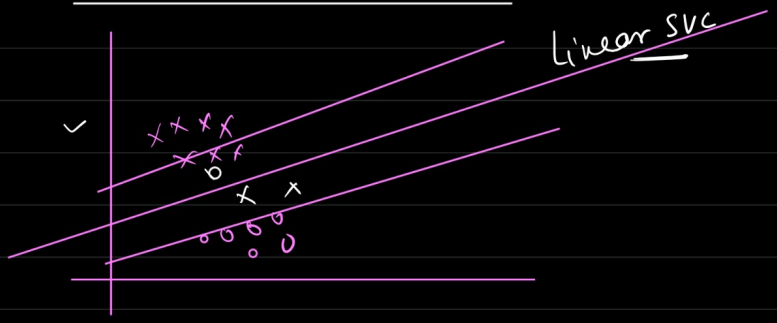
$\Downarrow$

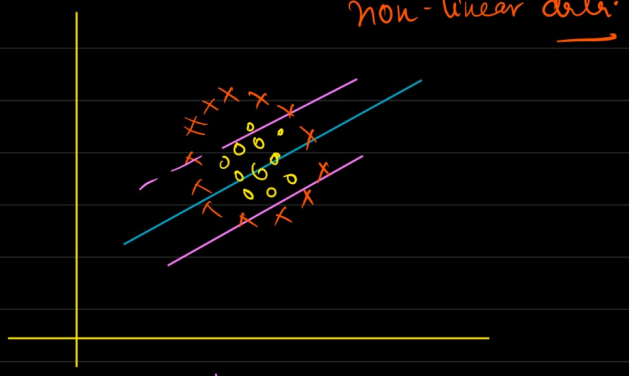
$\xi_i$ is the distance of dP's from its correct marginal plane.

$\Downarrow$

$\xi_i$ should be minimum.

# SVM - Kernel trick

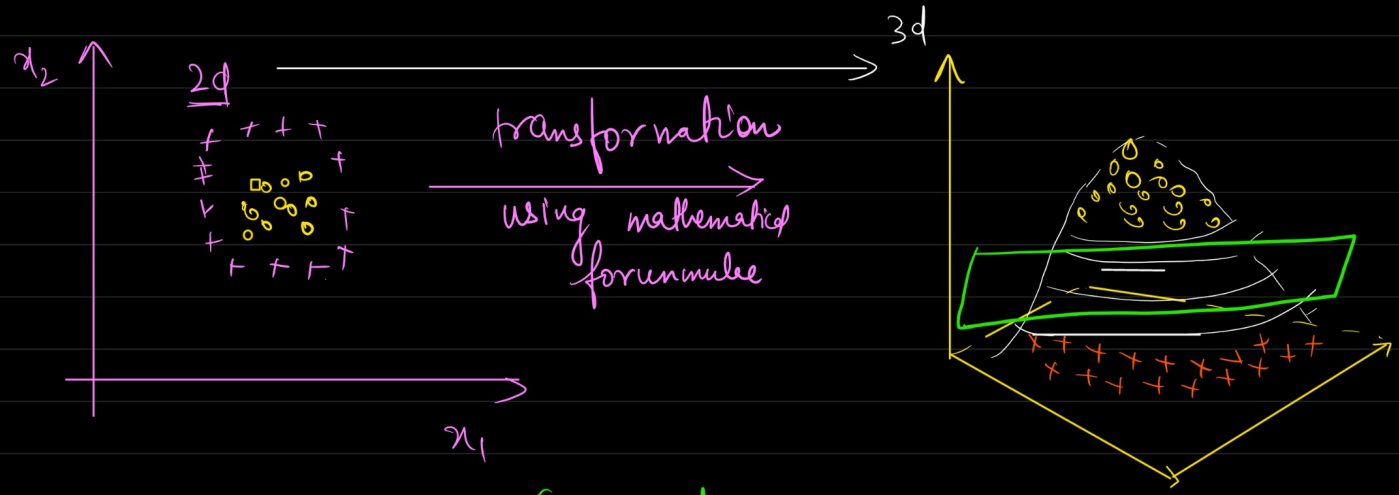



⇓

We can not solve using linear SVC.

## * SVM kernel trick




* 2d - 3d (Lower dimension to higher dimension using mathematical transformation, you can easily segregate between the classes.

$f(x) \Rightarrow$ Kernel $\Rightarrow$ Kernel transformation.

(mathematical transf)

→ There can be some overlapping after kernel transformation.

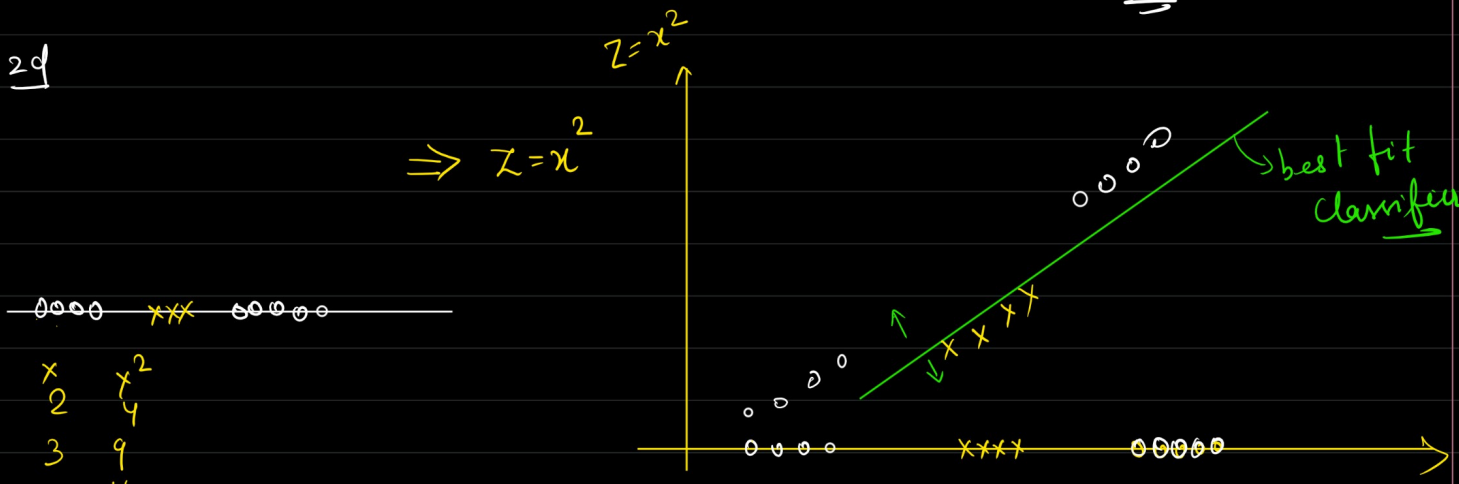
| X | Y |
|---|---|
| 2 | Yes |
| 3 | No |
| 4 | Yes |
| 5 | No |
| : | Yes |

$\Rightarrow$ 

↳ you cannot segregate both the class using

a linear SVC.

2d




| x | $x^2$ |
|---|---|
| 2 | 4 |
| 3 | 9 |
| 4 | 16 |
| 5 | 25 |
| - | - |
| - | - |

Idea :- to change the data from 1d to 2d, 2d - 3d, or increase the dimension by mathematical transformation to distinguish between the classes.

Why? → you are not sending the data in higher dimension, you are using mathematical transformation to achieve it

→ Using SVC kernel trick, you can also classify non-linear data.

* Kernel function

① Polynomial
② Rbf (Radial basis function)
③ Sigmoid

$ax^2 + bx + c = 0$
$ax^3 + bx^2 + cx + d = 0$

① Polynomial Kernel

$$f(X_1, X_2) = (X_1^T \cdot X_2 + c)^d$$

→ d is degree of Polynomial

constant

$(X_1^T \cdot X_2 + 1)^d$

Suppose we have two features $X_1$ & $X_2$

Y is output variable

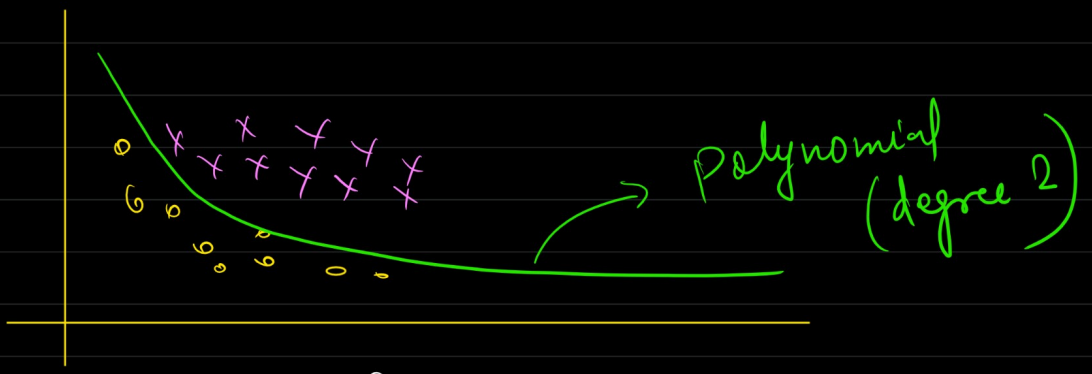
$$X_1^T \cdot X_2 = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \cdot \begin{bmatrix} X_1 & X_2 \end{bmatrix}$$

dot product

$$= \begin{bmatrix} X_1^2 & X_1 X_2 \\ X_1 X_2 & X_2^2 \end{bmatrix}$$

$$X_1 X_2 \Longrightarrow X_1, X_2, X_1^2, X_1 X_2, X_2^2$$

2d $\Longrightarrow$ 5 dimension.




## ② Radial basis feature (Rbf kernel).

→ creates non-linear combination of features 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 b/w two point $X_1$ and $X_2$

→ $\sigma$ – the variance/hyperparameter

To remember
Radial → radius.

③ Sigmoid Kernel

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

* Bessel Kernel ⟶ $f(x,y) = \frac{J_{v+1}(\sigma\|x-y\|)}{\|x-y\|^{-n(v+1)}}$

* ANOVA Kernel. ⟶ $f(x,y) = \sum_{k=1}^{n} \exp\left(-\sigma\left(x^k - y^k\right)^2\right)^q$

(ANOVA = multi dimensional regression)

* How to choose right kernel.

⇓

hyperparameter tuning.