Agenda:

1. Decision Tree classification
2. Decision Tree regression
3. Practical implementation
4. Ensemble technique

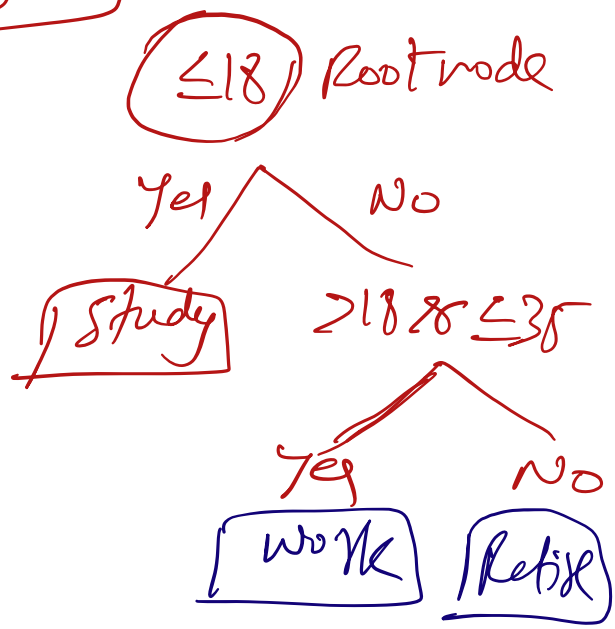
first we check for categorical data, then for continuous data

→ we can solve both classification and Regression problems.

ex:

```
if (age ≤ 18)
    print ("study")
elif (age > 18 & age ≤ 35)
    print ("work")
else:
    print ("Retire")
```

DT

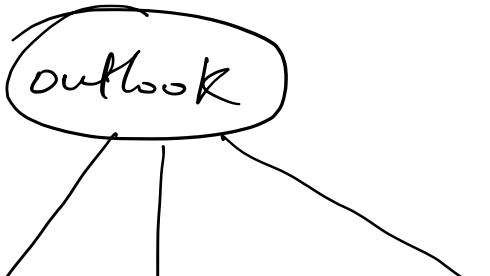


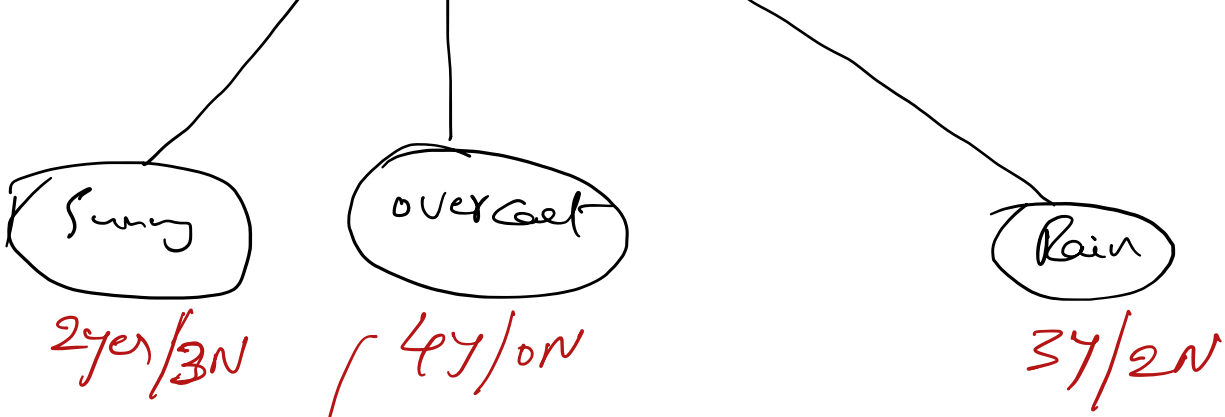
note:

nested if-else ⇒ DT

ex: dataset | Day | outlook | Temp | Humidity | wind | playTennis (Yes or No)

Classification:





pure split

impure split

ex: overcast

Coz, there is no confusion it is clear they will play.

[AKA pure leaf node]

no need to split it even more.

→ we split that particular node until we reach a pure node.

→ this process goes on.

Note: 1. Purity

→ to find whether the split is pure (or) not (or)

→ if the node pure / leaf node or not

we use two methods for that called:—

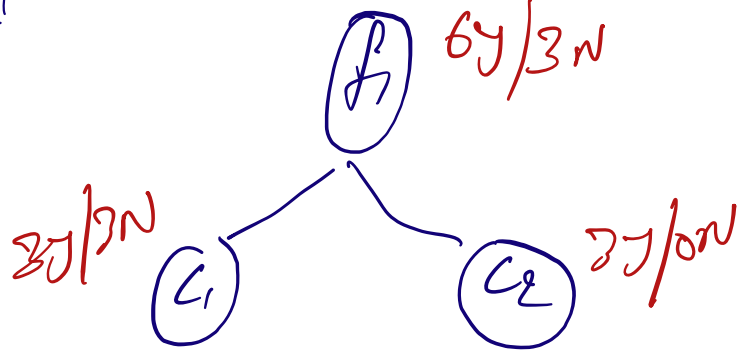
1. Entropy
2. Gini Impurity

2. How the features are selected? ⇒ Information Gain

Entropy:

$$H(S) = -(p_+)\log_2 p_+ - (p_-)\log_2 p_-$$

ex:



+(yes), −(no)

p_+ = probability of yes

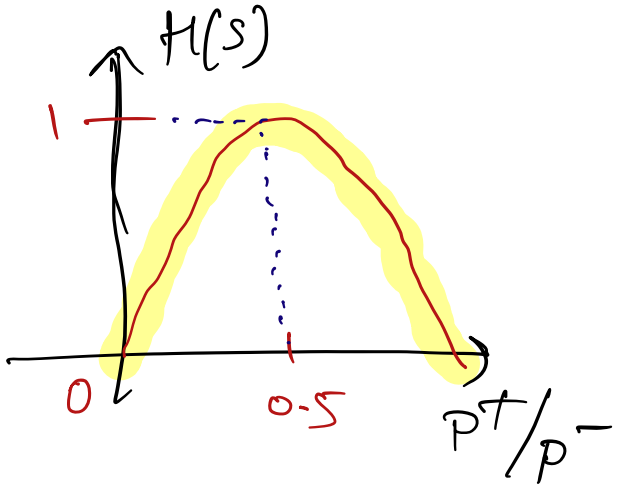
$C_2 = -\left(\frac{3}{3}\right)\log_2^{(3/3)} - \left(\frac{0}{3}\right)\log_2^{(0/3)}$ → (pure split)

$C_2 = -1\log_2^1 = 0 \Rightarrow$ hence, it is a pure split.

p_- = probability of no

$C_1 = -\frac{3}{6}\log_2^{3/6} - \frac{3}{6}\log_2^{3/6}$

$= 1$ (impure split)



note: Entropy is always b/w 0 and 1

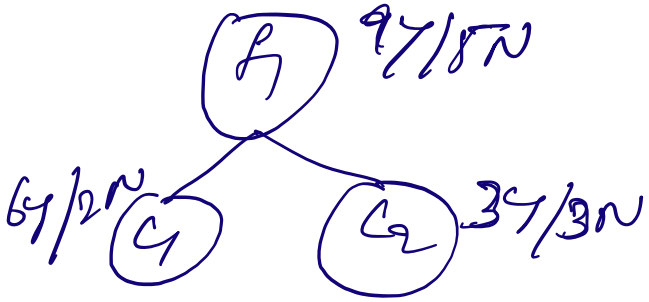
Information Gain :-

root node Entropy

that particular node entropy

$$Gain(S, f_1) = H(S) - \sum_{v \in Val} \frac{|S_v|}{|S|} H(S_v)$$

Ex:1



H(S) => Entropy of the root node

$$H(S) = -p_+ \log_2^{p_+} - (p_-) \log_2^{(p_-)}$$

$$= -\left(\frac{9}{14}\right) \log_2^{(9/14)} - \frac{5}{14} \log_2^{(5/14)}$$

$$\simeq 0.94$$

$$H(S_v) = -\frac{6}{8} \log_2^{(6/8)} - \frac{2}{8} \log_2^{(2/8)}$$

(C1)

$$H(C_1) = 0.81$$

Why

$$H(C_2) = 1$$

$$Gain(S, f_1) = 0.94 - \left[\frac{8}{14} \times 0.81 + \frac{6}{14} \times 1\right]$$

$$= 0.041$$

→ If we get the gain for $(f_2) = 0.051$ then what to use?

$$Gain(S, f_2) \gg Gain(S, f_1)$$

(we choose this)

note: I. Gain should be more to be selected

Gini Impurity :-

range: 0 to 0.5

$$Gini = 1 - \sum_{i=1}^{n} (P_i)^2$$

$$= 1 - \left((P_+)^2 + (P_-)^2 \right)$$

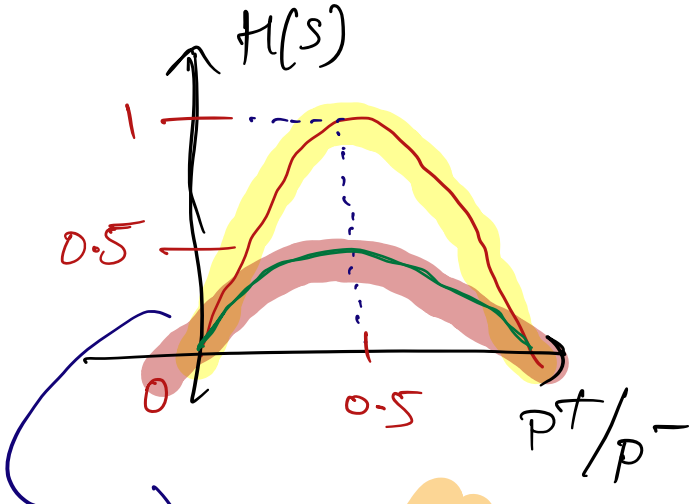
ex:

0 24/2N

↙

we got entropy = 1 for same above

Now,



$$= 1 - \left(\left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^2\right)$$

$$= 1 - \left(\frac{1}{2}\right) = 0.5$$

Graph for Gini Impurity

When to use what?

→ Decision tree itself have very bad time complexity (or) execution time

→ Entropy takes more time to execute than Gini Impurity

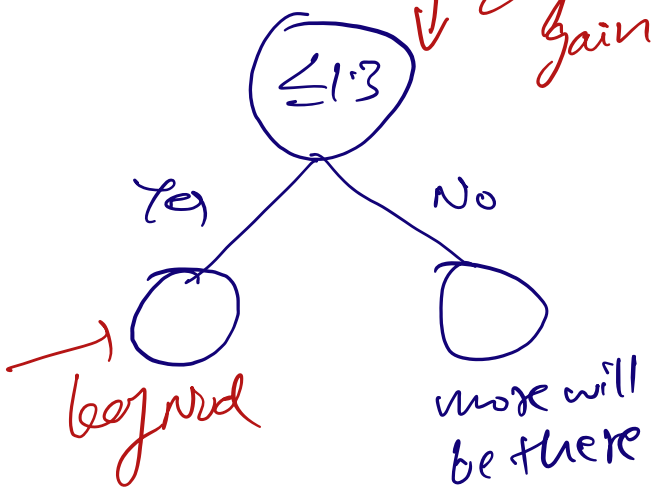
Note!

⇒ If we have 100's of features, then we use Gini Impurity

⇒ If we have few features, then we use Entropy....

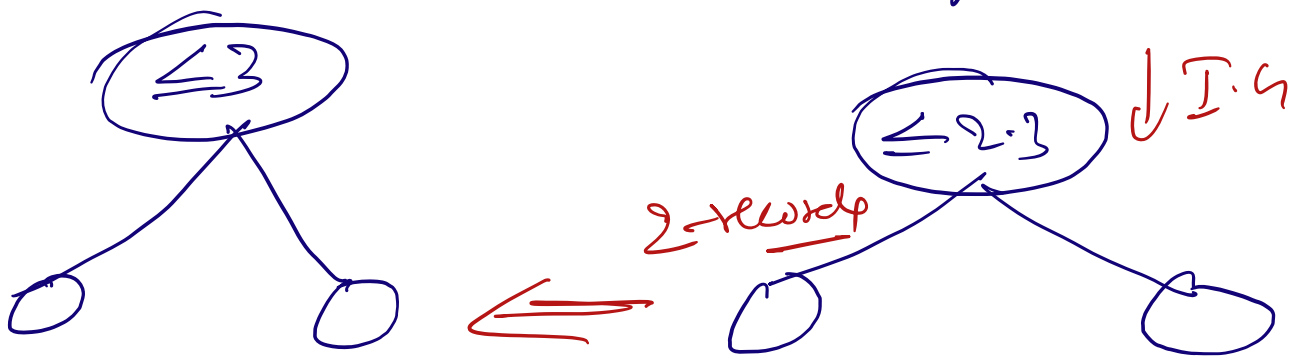
For continuous data

ex: f_1	op ⇒ f_1
2.3	1.3
1.3	2.3
4	3
5	4
7	5
3	7



next it will take

second feature



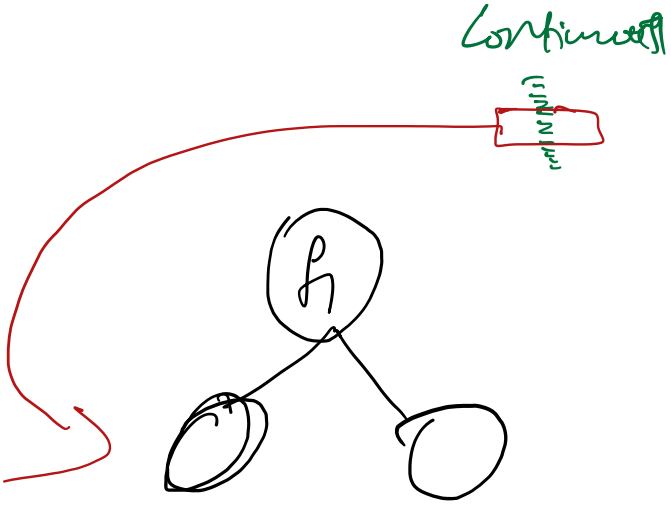
note: In this way they calculate for every feature making them root and Ch,

→ then we take the height value node and then split them further.

Decision Tree Regression

ex: let

mean value is calculated.



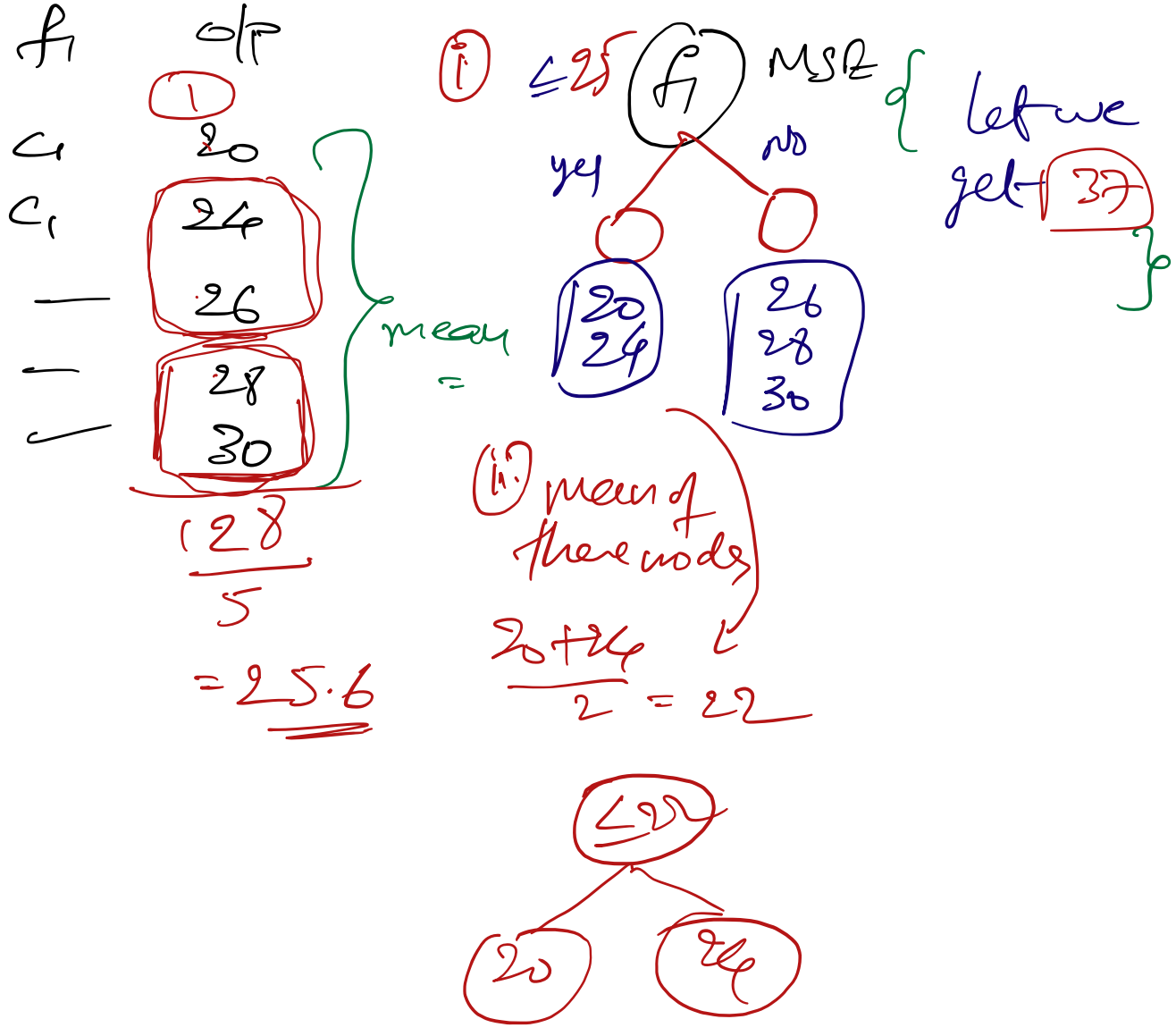
→ Compute MSE based on that, we split.

$$\frac{1}{2m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2$$

after take values and keep in a node and then again pick it for MSE again.

and this process continues, As the MSE value reduces it means we are nearing leaf node. and on the leaf node, whatever mean we have it is our o/p.

ex:



note: Decision Tree leads to overfitting, to deal it

1. post proning
2. pre proning

ex: 07yes/2no → so, it roughly say that it gives node

so, we don't do more pruning and stop here, called post pruning.

pre pruning:

→ it is calculated by hyperparameter
→ it tells abt max_depth, max_leaf

we use GridSearchCV

Agenda:

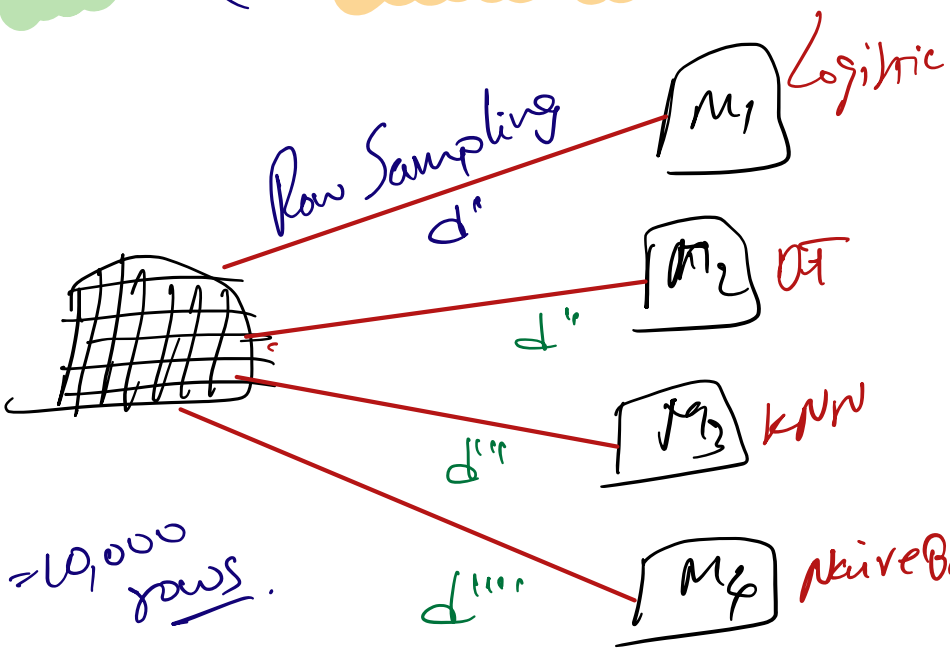
(1) Ensemble Techniques → Bagging
                        → Boosting
(2) Random Forest
(3) AdaBoost
(4) XG Boost

Ensemble Technique: we were discussing only single algo at a time, now we see how to use multiple algos together.

Bagging ↔ Boosting

Bagging: (for classification)

$[d' << D]$



$D = 10,000$ rows.

$d' =$ some.

note: when doing row sampling some of the rows might be repeated in other algos

How the inferencing will happen in test data:

M_1	give o/p =	0
M_2	give o/p =	1
M_3	give o/p =	1
M_4	give o/p =	1

0 → binary classification

all these o/p will get aggregated and the result will be acc to majority voting.

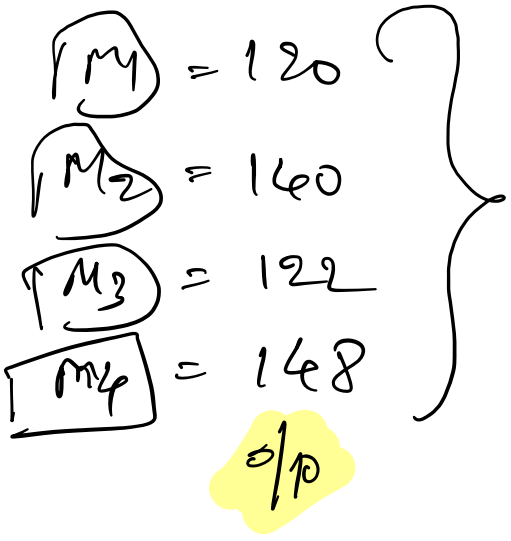
↓

Bootstrap Aggregator

note:

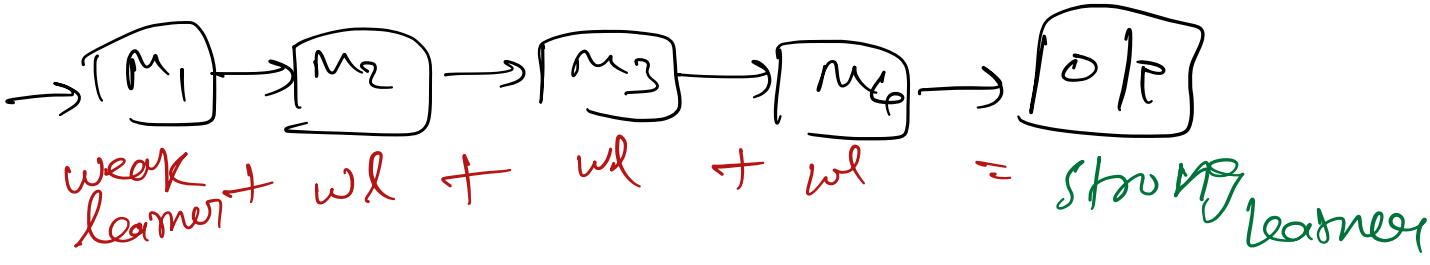
wt if we have equal no.of 1's and 0's, there is no chance for them to be tie....

(for Regression)



we take mean of these values, that will be the o/p of the model.

Boosting: it is a sequential combination of weak learning models.



Bagging

1. Random Forest classifier
2. Random Forest Regression

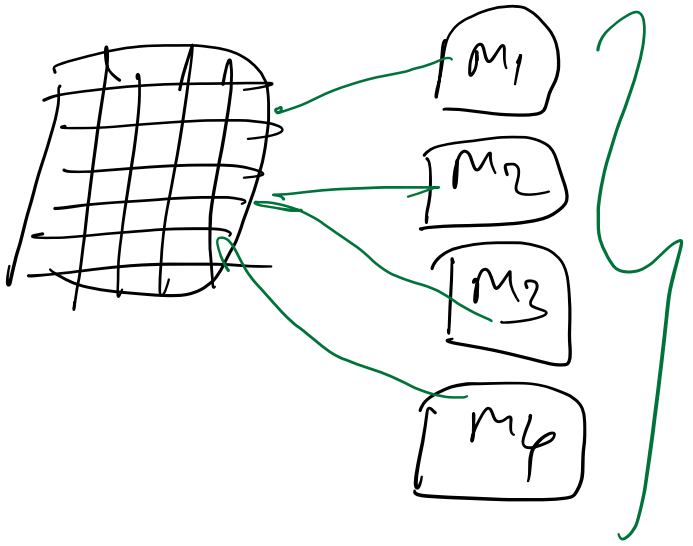
Boosting

1. Adaboost
2. Gradient
3. Xgboost

↳ :-

In DT we have problem of overfitting

{Low Bias, High Variance} ⇒ we convert it to {Low Bias, low Variance}



all are DT's

majority voting is followed.

⇒ in Regressor we take mean of o/p's

→ Normalization / ~~Standardization~~ is not required in DT

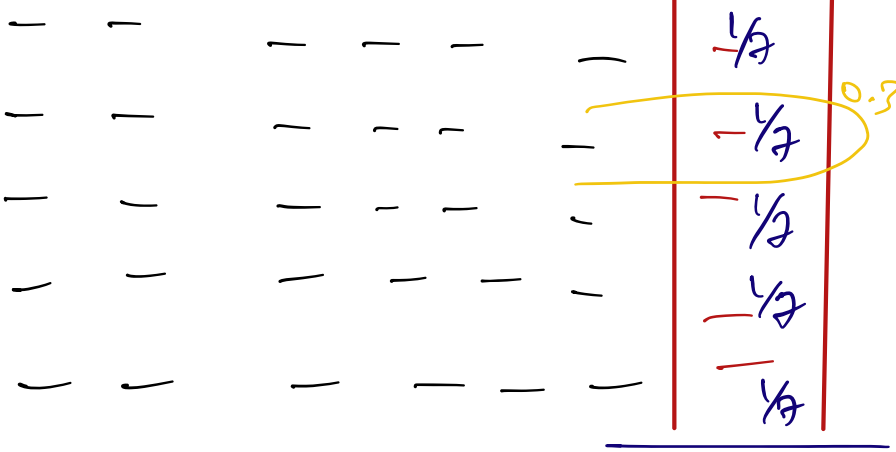
⇒ It is required in KNN

→ Random is not impacted by the outliers

KNN is impacted.

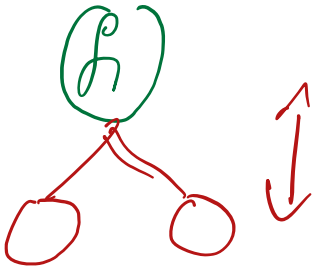
Adaboost:

f1	f2	f3	f4	O/P	wt	note:
—	—	—	—	Yes	1/7	0.05 If there is an
—	—	—	—	No	1/7	error, then the



value of that error = 1/7

(2) information gain



depth will be of only one level called stumps

[weak learner]

(1) Total error

Performance of stump :-

(2) $= \frac{1}{2} \log_e (1 - TE / TE)$

eg: $TE = 1/7 \Rightarrow Pz = 0.895$

'3' main things in AdaBoost

(3) update wts

New sample wt =

Correct Records

$(wt) \times e^{-Ps}$

$\Rightarrow \frac{1}{2} \times e^{-0.895} = 0.05$

In Correct wts:-

$(wt) \times e^{Ps} \Rightarrow \frac{1}{7} \times e^{0.895} = 0.349$

note: The cor record wts will be reduced, and the error records wt will be increased.

→ The wrong record wts will then go to the next weak learner

⇒ If we find the total sum of these new wts we find it is not equal to (1)

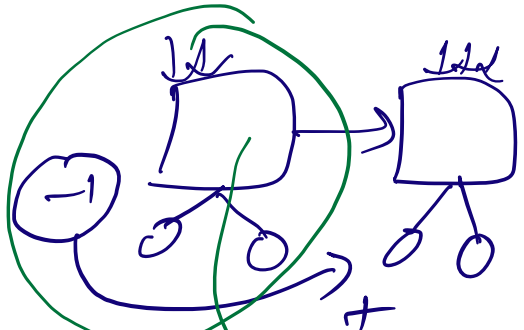
(1.) for this reason, we use Normalization wt:-

(i) we divide each new wt with the new wts summation.

ex:-

0.05 / 0.649 = 0.07

(2)



this bucket randomly creates num b/w 0 to 9

↳ first model we create, we got an error

↳ now along with that error, we pass some more rows into a new stump with the help of first stump.

↳ for this we have a concept

Bucket

Buckets

[0 to 0.07] → 1 getting added to next row

[0.07 to 0.14]

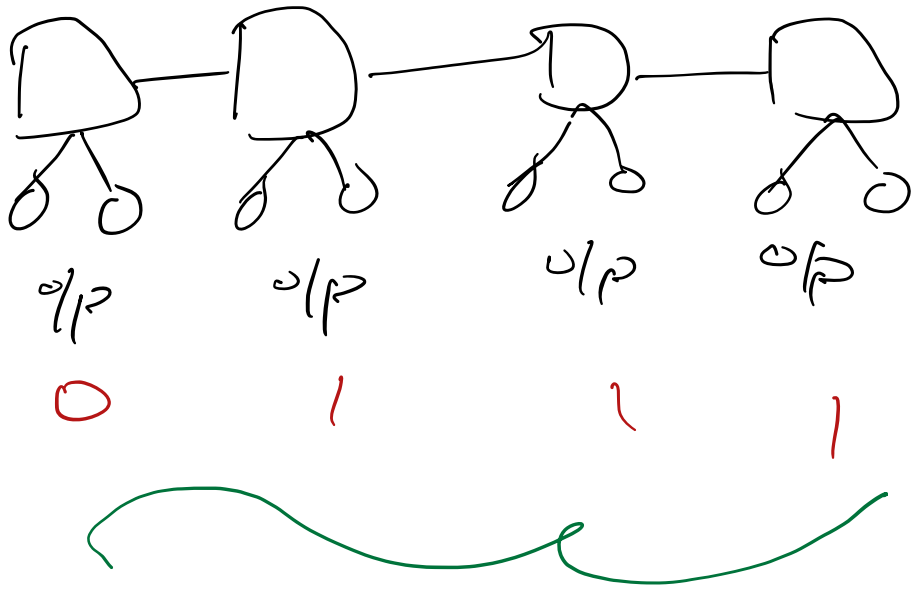
[0.14 to 0.21]

[0.21 to 0.747] ← this record is the biggest bucket size.

[0.747 to 0.751)

probability of that number being in any of these buckets, it will be more for the error bucket, so along with the buckets, this will also get passed.

After creating stumps of various DT's :



again we take majority voting = 1 (o/p)
(for classification)

Avg for regression.

Black box (Vs) white box:

Linear Regression → white Box
Rand Forest → Black Box
DT → white Box
ANN → Black Box

interview Qn.

Agenda:

1. Xgboost classifier
2. Xgboost Regressor
3. SVM
4. SVR

Xgboost Classifier: [Xtreme Gradient Boosting]

ex: {Dataset}

Salary	Credit	Approval
<=50	Bad	0
<=50	Good	1
<=50	G	1
>50	B	0
>50	G	1
>50k	normal	1

→ This Xgboost can be used for both classification and Regression problems

→ Internally we use DT in this Xgboost

→ when we use Xgboost, we first create a Base model with probability = 0.5

⟹ classification

Base model → Prob = 0.5

(weak learner)

[This is a dummy model]

→ Then we create a field in the original table called as Residual field (probably a new column)

Salary	Credit	Approval	Residual → Approval-base model prob
<=50	Bad	0	→ (-0.5)
<=50	Good	1	→ 0.5
<=50	A	1	→ 0.5
>50	B	0	→ (-0.5)
>50	G	1	→ 0.5
>50k	Nor	1	→ 0.5
<=50k	Normal	1	→ 0.5
<=50k	Normal	0	→ (-0.5)

Steps: After creating a base model

1. Creat a Binary Decision Tree using the features
2. Calculate the similarity weight

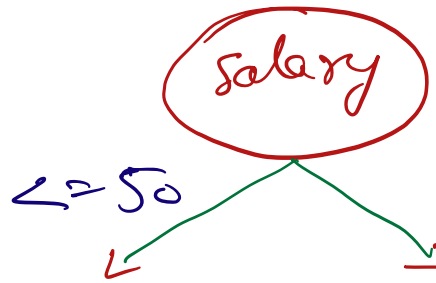
$$\frac{\sum (\text{Residual})^2}{(\quad)}$$

$\sum(p(1-p)+\lambda)$

③ Calculate the Information Gain.

we use these steps to create a Xgboost classifier.

⇒ we always create a binary decision Tree



[-0.5, 0.5, 0.5, -0.5]

[-0.5, 0.5, 0.5]

↳ My for this, we get Similarity wt = 0.33

(Step ① is done)

$$= \frac{(-0.5+0.5+0.5-0.5)^2}{0.5(1-0.5)+0.5(1-0.5)+0.5(1-0.5)+0.5(1-0.5)} \quad 0$$

Similarity wt = 0

we generally initialize $\lambda=0$

① ② ③

→ if we calculate similarity wt for root node
↳ take all residuals = 0.142

(step ② is done)

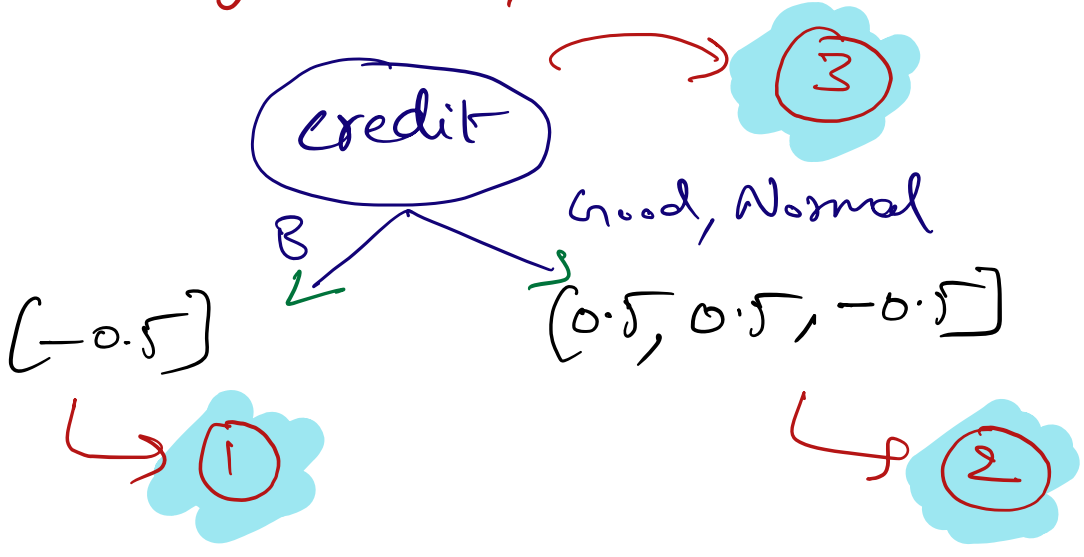
⇓

Information Gain = 0 + 0.33 − 0.14
= 0.19

note: we did [① + ② − ③]

→ after we get the information gain, we use it to do further split and select the exact feature, which helps us in finding the proper root node for the model.

→ after salary, we split credit



find similarity wt :-

1. 1
2. 0.33
3. 0

} $1 + 0.33 - 0 = 1.33$

↳ this "zero" came from (1) of salary tree with $\angle = 50$

Note:

and this process goes on.

Inferencing :-

(1) The first record will go to the base model and the probability is 0.5

(2) But, this is an assumed probability, how to find the actual probability???

(3) we use logs for that.

$\log(P/1-P)$ this is applied only for the base model.

↳ in this case $\log^{0.5/1-0.5} = \log^1 = 0$

now, when a record goes into this base model

we get the probability = 0

for 1st row = $[0 + \alpha(1)]$

$\Rightarrow \sigma[0 + \alpha(1)]$

actual prob

learning rate

↳ it can be any small value ace to the learning rate we choose

[just like gradient descent]

$\alpha = 0.01$

↳ In this case.

the first row goes into the Binary tree and reaches till Credit, Bad

Similarity wt = 1

Sigmoid function (activation function)

Since, this is a classification problem

based on the value of α, the o/p will be between 0 to 1

Hence,

$$\sigma[0 + \alpha_1(DT_1) + \alpha_2(DT_2) + \alpha_3(DT_3) + \cdots\cdots + \alpha_n(DT_n)]$$

Since, we are adding all the DT's it results us in Boosting

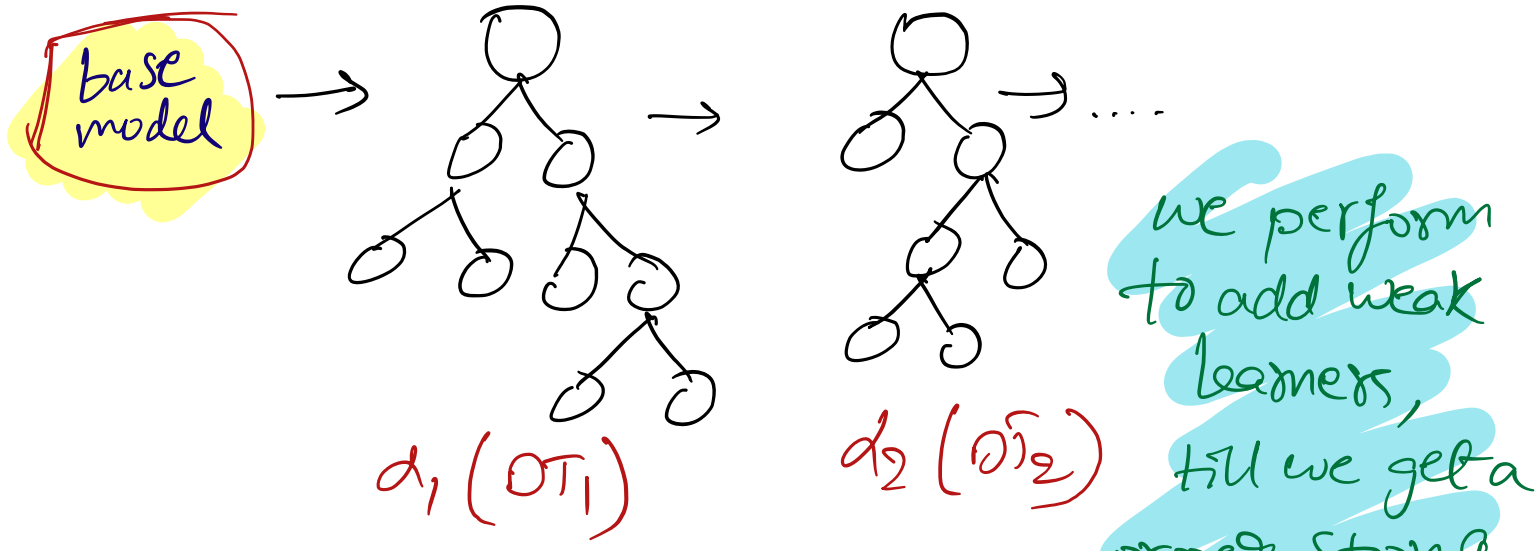
note:

1. we can split credit even more, but we prefer a node whose Information Gain is more

2. Since, there is lot of math involved, we call Xgboost as a Black Box model

3. we tend to do pre-proning in this total process

Diagramitically:-



we perform to add weak learners, till we get a proper strong model

AKA a good boosting <= model

4. (DT's) get constructed depending on Independent features

5.

we basically setup the hyperparameter (d) by the help of Cross Validation

Xgboost Regressor :-

Exp	Lap	Salary → o/p Reg
2	Yes	40k → −11k
2.5	Yes	49k → 9k
3	No	52k → 1k
4	No	60k → 9k
4.5	Yes	62k → 11k

→ we create **base model** = avg of o/p's

$$= \frac{\Sigma \text{Salary}}{6}$$

$$= 51k$$

→ Residual

⇒ Salary − base model

DT

(3) Exp → [−11, −9, 1, 9, 11]

≤2 → [−11] (1)

>2 → [−9, 1, 9, 11] (2)

similarity wt =

$$\frac{\Sigma (\text{Residual})^2}{\text{No. of Residuals} + (\lambda)}$$

⇒ let $\lambda = 1$

1. 60.5
2. 28.8
3. 0.16

} $I.h = 60.5 + 28.8 - 0.16$

$= 98.13$

⇒ we got [98.34] acc to the first row with [exp = 2], we will get diff [DT's] for diff rows

⇒ we pick the split which has the better Information gain.

Inferencing :- [o/p]

$$51 + \alpha_1 (DT_1) + \alpha_2 (DT_2) + \cdots + \alpha_n (DT_n)$$

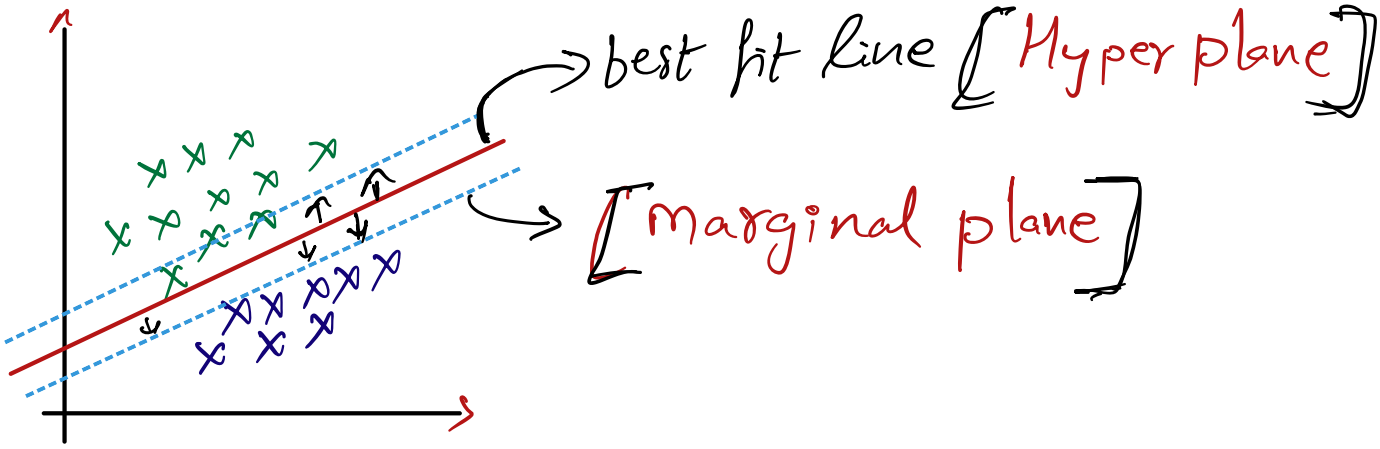
↓ [base model]

↘ acc to wt path it is going in a [DT], we take the avg of those residuals of that particular [DT] split.

⇒ This is also a Black Box model

SVM :- (it is like Linear regression)
[Support Vector Machine]

DON'T FORGET



note: (1.) If there are no overlaps (or) if there are no points/values in b/w the hyperplane and Marginal plane.

⇒ then we call it **Hard Marginal plane.**

(2.) if there are overlaps (or) if there are points in b/w the hyperplane and marginal plane

⇒ then we call it **Soft Marginal plane**

⇒ we focus on creating the **Marginal plane** with **Maximum Distance**, Even though there are some **Errors**, we consider them by solving it with the help of a **Hyperparamette.**

How to construct those lines?

$$y = mx + c$$

$$\Downarrow$$

$$ax + by + c = 0$$

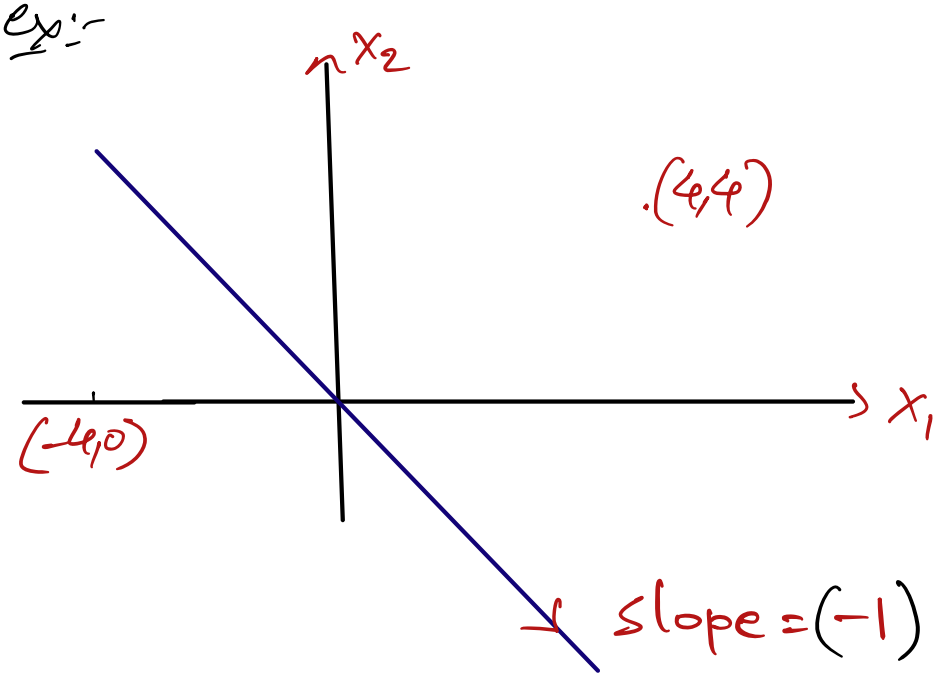
$$y = -\frac{c}{b} - \frac{ax}{b}$$

$$m = -a/b$$

$$y = w_1 x + w_2 x_2 + \cdots + b$$

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

ex:-



let, the eqn is passing through origin

$\Rightarrow b = 0$ and

$w = -1$

$$\Rightarrow y = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$$

lets, take the $(-4,0)$ point

$$\Rightarrow \quad y = \begin{bmatrix} -1 \\ 0 \end{bmatrix} \begin{bmatrix} -4 & 0 \end{bmatrix}$$

$$= 4 \Rightarrow \text{+ve Value}$$

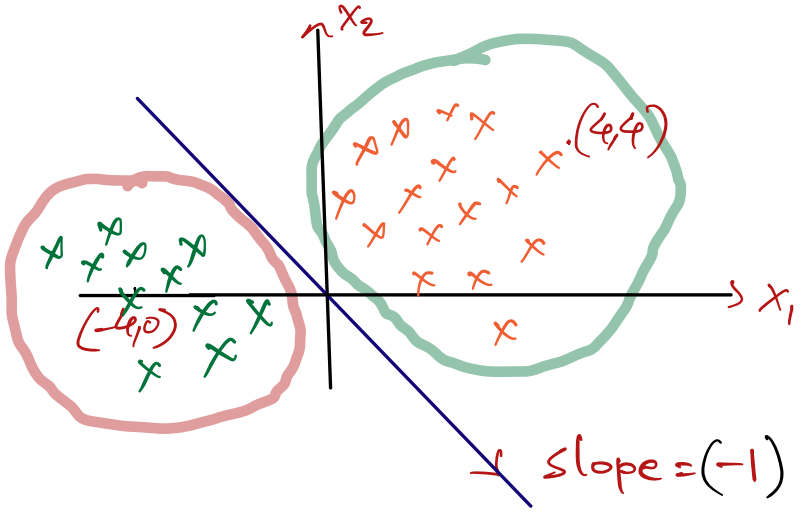
So, all the values, below the line (or) in the area of $(-4,0)$ point are considered to be +ve values

lets, take the $(+4,4)$ point

$$\Rightarrow \quad y = \begin{bmatrix} -1 \\ 0 \end{bmatrix} \begin{bmatrix} 4 & 4 \end{bmatrix}$$

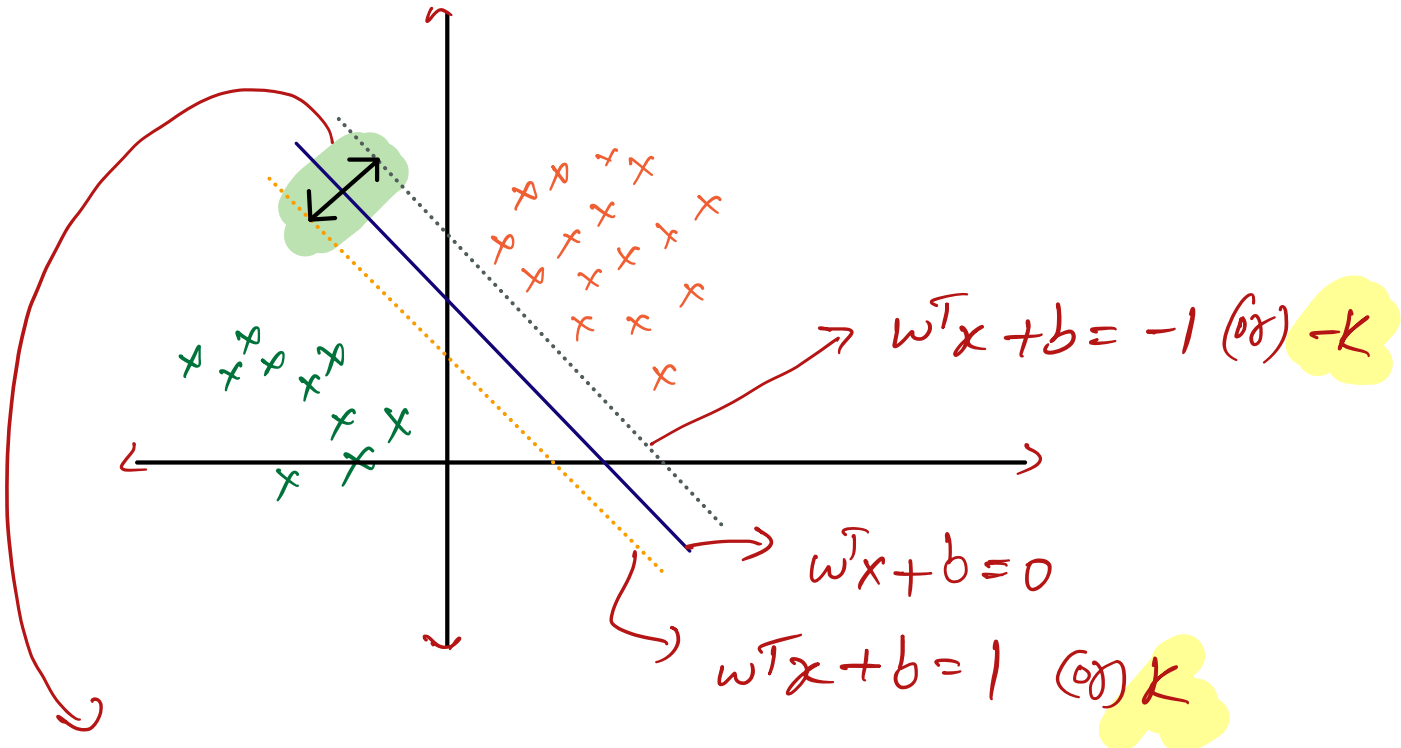
$$= -4 \Rightarrow \text{-ve Value}$$

So, all the values, above the line (or) in the area of $(4,4)$ point are considered to be -ve values



we divided them,

How to create marginal plane?



our aim is to increase this distance, if we can increase this dist, it means our model is performing well.

$$w^T x_1 + b = 1$$
$$w^T x_2 + b = -1$$

(-) (-) (+)

$$w^T (x_1 - x_2) = 2$$

$$\frac{w^T (x_1 - x_2)}{\|w\|} = \frac{2}{\|w\|}$$

Maximize cost func^n

To remove the magnitude from a vector, we divide it with its unit vector

note: our aim is to maximize the
→ value of $\frac{2}{\|w\|}$ by updating (w, b) value

→ if we can maximize, then our marginal plane will spread and dist↑

Such that

$$y_i \begin{cases} +1 & w^T x + b \geq 1 \\ -1 & w^T x + b \leq -1 \end{cases}$$

⇓

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

⇓

for correct point

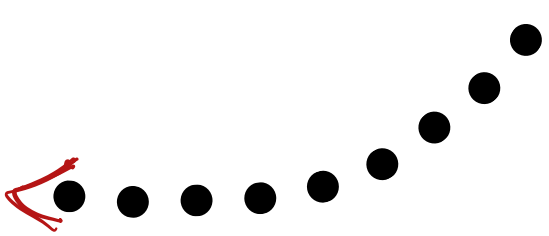
for minimize ⇒

$$\min_{(w,b)} \frac{||w||}{2} + c_i \sum_{i=1}^{n} \xi_i$$

ξ_i → zeta of i

it says, how many errors we can have.
(like points in opposite class)

Summation of the distance of the wrong data points.



↳ distance from its corr marginal line to the location where the point is.

i.e, distance from related marginal line to the other class location of that point

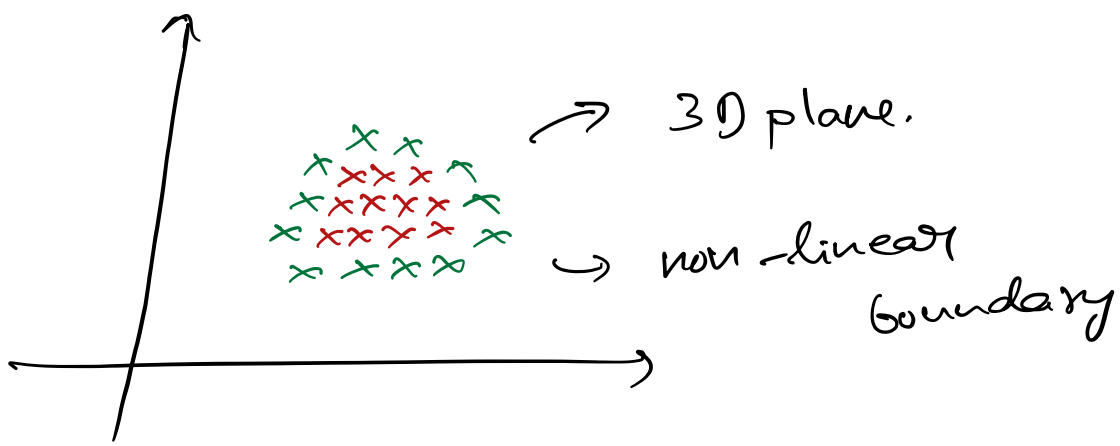
$$C \sum_{i=1}^{n} \xi_i$$

↳ this is a hyperparametre used to minimize the errors.

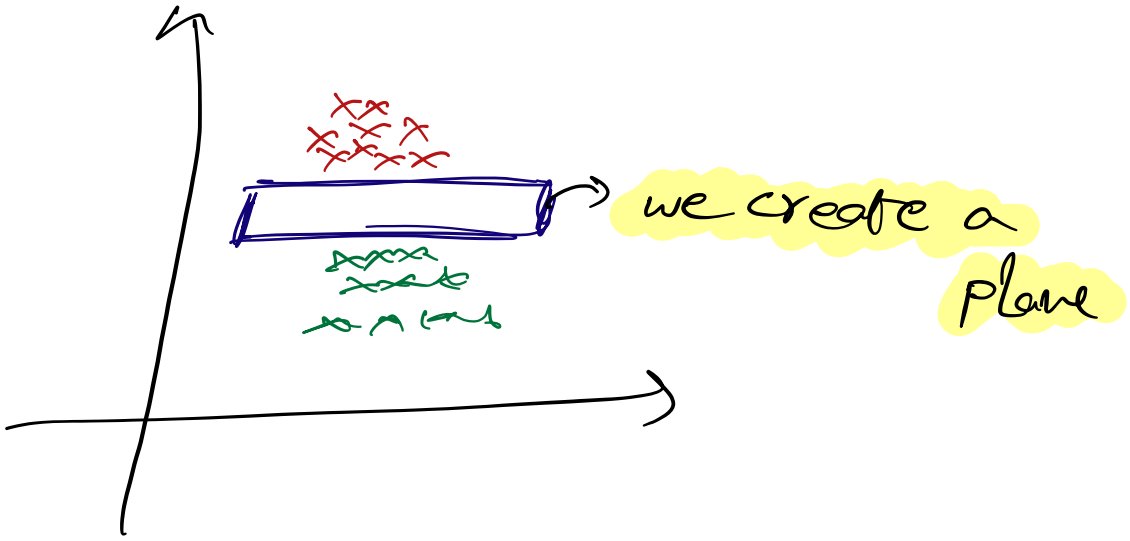
⇒ we can't remove all the errors, hence we are ok with few points in wrong location.

SVM Kernel :-

more info in sep notes



we convert it by pushing the points, to seperate them.



SVR: gives us the flexibility to define how much error is acceptable in our model and will find an appropriate line to fit the data

Minimize:

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