Decision Tree

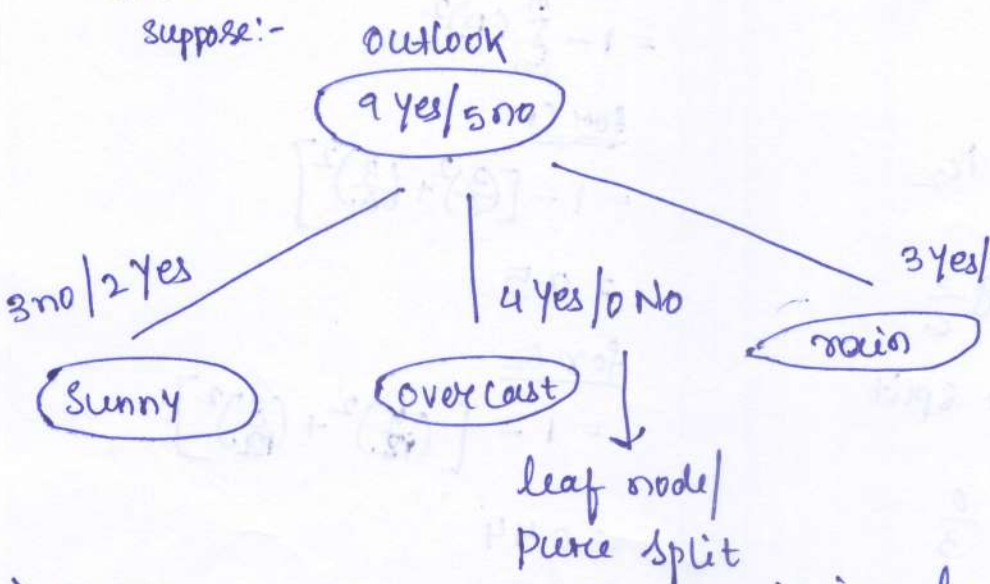
- ID3
- CART (classification & Regression Tree)

What is a leaf node?

⇒ No-more splitting happens if leaf node occurs

Ex:-

Suppose:-



→ Splitting happens until/ unless it is a leaf node.

How application will know whether it is a leaf node or not

→ purity of a leaf node is checked by using

- → Entropy
- → Gini Index

Entropy

→ used for small data set.

→ It ranges between (0-1)

Binarey classification

$$H(S) = -P_+ \log P_+ - P_- \log P_-$$

Multi class classification

$$H(S) = -P_{C_1} \log P_{C_1} - P_{C_2} \log P_{C_2}$$

Gini Index

→ for large dataset

→ It ranges between (0-0.5)

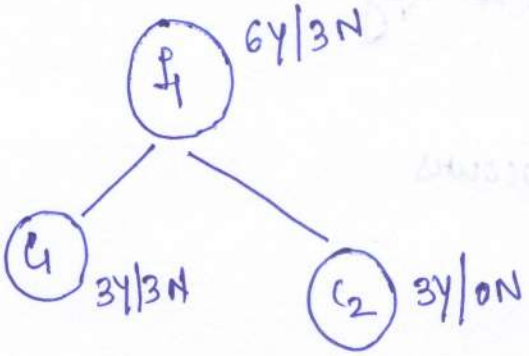
$$G.I = 1 - \sum_{i=1}^{n} (P)^2$$

for Binary

$$= 1 - \sum_{i=1}^{2} \left((P_+)^2 + (P_-)^2\right)$$

Entropy

Ex:-



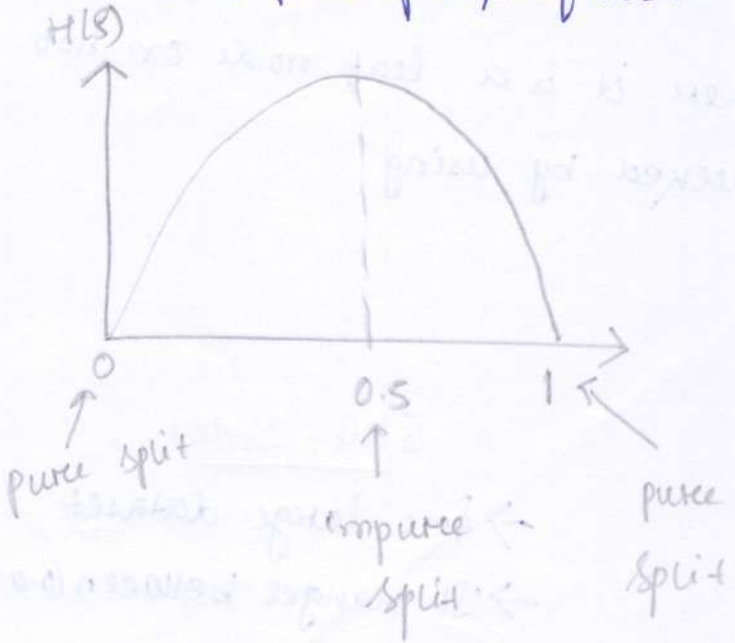
$$H(S) = -P_{C_1} \log P_{C_1} - P_{C_2} \log P_{C_2}$$

$$H(C_1) = -\frac{3}{6} \log \frac{3}{6} - \frac{3}{6} \log \frac{3}{6}$$

$= 1 \rightarrow$ very Impure split

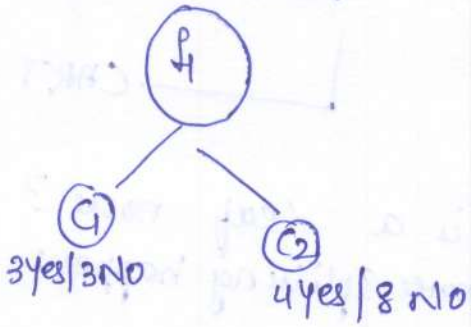
$$H(C_2) = -\frac{3}{0} \log \frac{3}{0} - \frac{0}{3} \log \frac{0}{3}$$

$= 0 \rightarrow$ pure split / leaf node.



G.I

Ex:-



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

for C_1

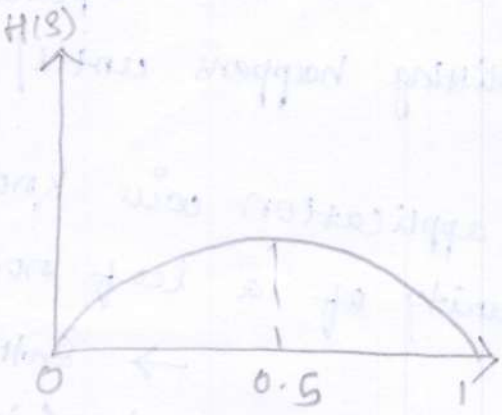
$$= 1 - \left[\left(\frac{3}{6}\right)^2 + \left(\frac{3}{6}\right)^2\right]$$

$= 0.5$

for C_2

$$= 1 - \left[\left(\frac{4}{12}\right)^2 + \left(\frac{8}{12}\right)^2\right]$$

$= 0.44$



~~How~~ features

which features to take the split?

→ It is decided by information Gain

Information Gain

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

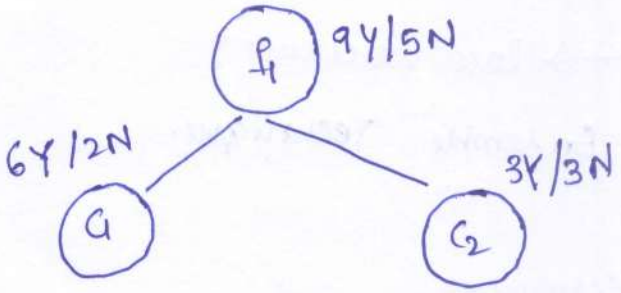
$H(S)$ = entropy / G.I

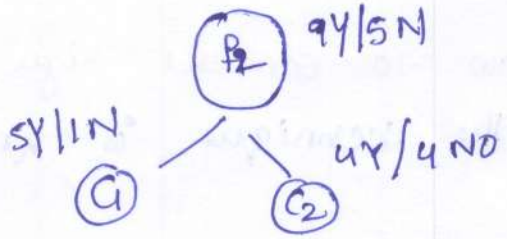
$H(S_v)$ = entropy of child node

$|S|$ = Total nodes

$|S_v|$ = child nodes.

Ex:-





entropy of root

$$I(S) = -P_+ \log_2 P_+ - P_- \log_2 P_-$$

$$= -\frac{9}{14} \log_2 \frac{9}{14} - \frac{5}{14} \log_2 \frac{5}{14}$$

$$\approx 0.94$$

$$I(C_1) = -\frac{6}{8} \log_2 \frac{6}{8} - \frac{2}{8} \log_2 \frac{2}{8}$$

$$= 0.81$$

$$I(C_2) = 1$$

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

$$= 0.049$$

$$H(S) = -\frac{9}{14} \log_2 \frac{9}{14} - \frac{5}{14} \log_2 \frac{5}{14}$$

$$= 0.937825$$

$$H(C_1) = -\frac{5}{6} \log_2 \frac{5}{6} - \frac{1}{6} \log_2 \frac{1}{6}$$

$$= 0.6497$$

$$H(C_2) = 1$$

$$Gain(S, f_2) = 0.937825 - \left[0.649 \frac{7}{7} \times \frac{6}{14} + 1 \times \frac{8}{14}\right]$$

$$= 0.088425$$

$$I.G(f_2) > I.G(f_1)$$

⇓

So we use feature-2 for splitting

Why over-fitting happens in Decision Tree?

⇒ If decision tree is drawn to its full depth
- ↳ Training accuracy ↑↑ (low bias)
- ↳ Testing " ↓↓ (High Variance)

How to fix overfitting?

Pre-pruning
→ Initially max depth is choosen

Post-pruning
→ First the DT is built to depth then cutting is done.

How to convert high variance ⟶ low variance?

→ The technique is known as Ensemble Technique.

Ensemble Technique

Boosting
→ Each Algorithm are parallel
→ Majority voting classifier
→ Avg o/p in Regressor

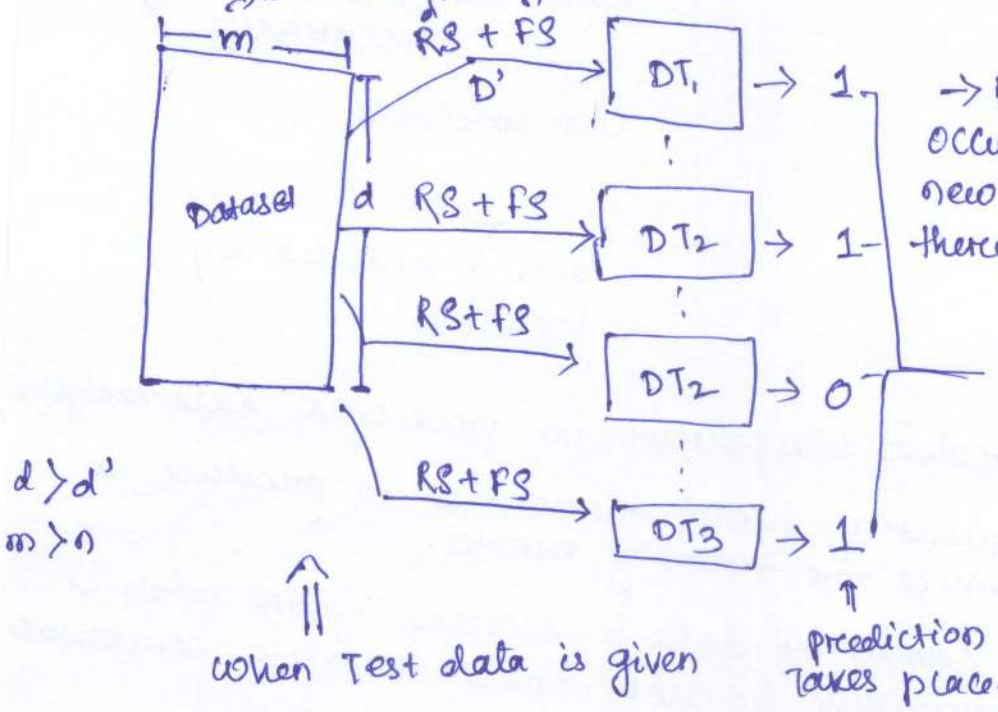
Ex:-
Random Boost forest classifier
- Regressor.

Bagging
→ Sequential Algorithm is applied.

Ex:-
Adaboost, Cathoost, Gradient boosting, XG-Boost.

Random forest classifier

RS → Row Sampling
FS → feature Sampling



→ Row sampling with Replacem occures in the step-① i.e. here Rows & dataset is pick there might be two repetetion

To aggregate the majorit vote is consider.
o/p = 1

when Test data is given

prediction takes place.

Advantages

① To make high variance → low variance we are using multiple decision Tree in parallel and doing majority voting

② If we are changing data in the dataset it will not affect much the accuracy, bcz the data are splitted properely within the decision tree not causing major impact.

if it is a Regression

→ we take mean or median of the output

How many Decision Tree to be used?

- It can be decided by hyper-parameter Tuning.

Bagging

① High variance + low bias

↓

Variance is reduced by Randomization + Aggregation

Boosting

② High bias (low training accuracy) + low variance

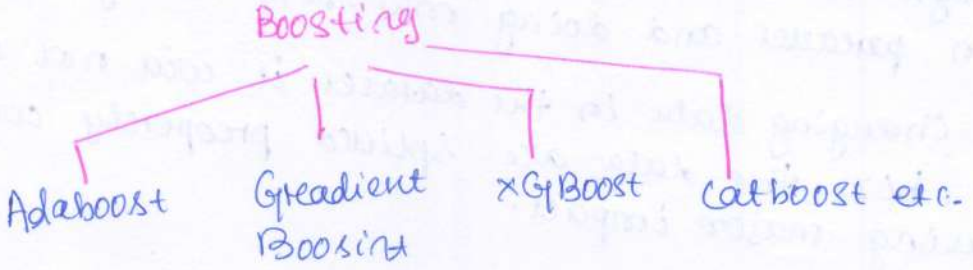
↓

bias is reduced by boosting

Boosting is a method to reduce bias/error in predictive data analysis

→ It combines set of sequential weak learners to produce a strong learner to minimize the training errors.

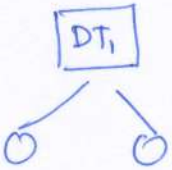
→ In boosting, a random sample of data is selected, fitted with a model and then trained sequentially i.e each model tries to compensate for weakness of its predecessor with each iteration.



Adaboost

→ combines weak learners in sequential to make a strong learner.

Stump = DT with one level & a binary o/p.



→ Each stump is made by taking the previous stumps mistakes into account.

f_1	f_2	f_3	O/P	weights	updated weights	New sample weig
			Yes	$\frac{1}{7}$	0.058	$\frac{0.058}{0.697} = 0.083$
			NO	$\frac{1}{7}$	0.058	0.0832
			–	$\frac{1}{7}$	0.058	0.0832
			–	$\frac{1}{7}$	0.349	0.5007
			–	$\frac{1}{7}$	0.058	0.0832
			–	$\frac{1}{7}$	0.058	0.0832
			–	$\frac{1}{7}$	0.058	0.0832
					0.697	0.9921

Step ① ⇒ Equal weights are assigned i.e $= \frac{1}{N} = \frac{1}{7}$

Step ② ⇒ Then we will create decision stump for each feature and calculate G.I of each tree. Let tree with lowest G.I will be our first stump.

Let 4th one is our 1st Stump.

Step ③ we will calculate the "Amount of say" or "Importance" or "influence" i.e performance of stump.

$$\frac{1}{2}\log_e\left(\frac{1-TE}{TE}\right) = \frac{1}{2}\log_e\left(\frac{1-1/7}{1/7}\right) \simeq 0.895$$

∴ Let one record is at fault so error $= \frac{1}{7}$

Step ④ :- Weights need to be updated i.e decrease the weight of correct record, increase the weight of incorrect record.

For correct record = weight $\times e^{-ps}$

$$= \frac{1}{7} \times e^{-0.895} = 0.058$$

For wrong record = weight $\times e^{ps}$

$$= \frac{1}{7} \times e^{0.895} = 0.349$$

We know that Total sum of updated weights must be equal to 1 but we are getting 0.697. To bring this sum equal to 1 we need to normalize these weights by dividing all the weights by the total sum of updated weights..

Then we will do bucketing, i.e max Bucket size is for wrong records i.e the wrong predicted is passed to next decision tree So that training of weak learners will be done on the next step

Gradient Boosting

→ The main idea behind this algorithm is to build models sequentially and these subsequent model try to reduce the errors of the previous model.

Ex:-

Exp	Degree	Salary (y)	(predicted value) $\hat{y}$	$y-\hat{y}$ (R1)	let suppose R_2	
2	BE	50K	75	-25	-23	75+(-23) = 52 (overfitting)
3	masters	70K	75	-5	-3	
5	masters	80K	75	5	3	75+ α(R2)
6	PHD	100K	75	25	20	75+0.1 (-23) = 73.7

Step 1:- Compute the Base model to get single o/p.

$$avg = \frac{50+70+80+100}{4}$$

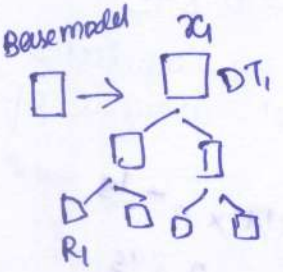
$$= 75$$

Step 2:- Compute Residuals Error | Pseudo Residuals (R_1)

actual − prediction

Step 3:- construct DT

$\{x_i, R_1\}$



$h_1(x)$ = o/p of DT_1

Step 4:- Let we pass independent features to DT then R_2 error comes

Here we say, predicted − R_2 = 52 it is close to y but overfitting

Does why?

→ Because though it has low bias but it will have high variance. new test data is added to the avoid this learning rate is introduced. ($\alpha = 0-1$)

Still 73.7 is quite large than y so next decision tree will be created based on this error.

$$F(x) = h_0(x) + \alpha_1 h_1(x) + \alpha_2 h_2(x) + \ldots = \sum_{i=1}^{n} \alpha_i h_i(x)$$

Machine learning is nothing but finding relationship between i/p and output data.

Additive Modelling

Adding functions / DT at every step and boosting Algorithm uses additive modelling format.

Pseudo Algorithm Explanation of Gradient Boosting

Ex:-

R&D Spend	Administration	Marketing Spend	(Y) profit
165	137	472	192
101	92	250	144
29	127	201	91

Step 1:- Initialize

Requirements

1) x_i, y_i → dependent features
↓ independent features.

2) differentiable Loss function $L(y, f(x))$ based on either Regression or classification problem.

$$L(Y, F(x)) \rightarrow L(Y, \hat{Y})$$

($\hat{y}$)

Here we are using MSE (Mean Square Error) $= \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

Step 1:- Initialize $f_0(x) = \text{argmin} \sum_{i=1}^{N} L(y_i, \gamma)$

Aim:- our aim is to establish relationship between $Y = F(x)$

Since, it is a boosting $F(x)$ is obtained adding small functions sequentially.

$$F(x) = F_0(x) + F_1(x) + F_2(x) \ldots\ldots + F_n(x)$$

$F_0(x)$ ← Base model
$F_1(x)$ ↳ DT

Step 1 :- Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1} L(y_i, \gamma)$ / Initialize Base Model.

$\gamma = \hat{y}$

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

Here our aim is to get "γ" value in such a way that the error is less.

$F_0(x) = \arg\min_\gamma \frac{1}{2} \sum_{i=1}^{n} (y_i - \gamma)^2$

↓ diff w.r.t γ

$$\frac{d F_0(x)}{d\gamma} = \frac{d}{d\gamma} \frac{1}{2} \sum_{i=1}^{n} (y_i - \gamma)^2 = \frac{1}{2} \sum_{i=1}^{n} \frac{d}{d\gamma} (y_i - \gamma)^2$$

$$= \sum_{i=1}^{n} (y_i - \gamma)' \frac{d}{d\gamma} (y_i - \gamma)$$

$$0 = -\sum_{i=1}^{n} (y_i - \gamma)$$

$$\sum_{i=1}^{n} (\gamma - y_i) = 0$$

$$\sum_{i=1}^{3} (\gamma - y_i) = 0$$

$$(\gamma - 192) + (\gamma - 144) + (\gamma - 91)) = 0$$

$$3\gamma = 192 + 144 + 91$$

$$\gamma = \frac{192 + 144 + 91}{3} \quad (\because \text{mean of the o/p column})$$

$$= \frac{427}{3}$$

$$= 142.33$$

Here, the base model value is 142.33

② For $m = 1$ to M:

$$F(x) = \underbrace{f_0(x)}_{\text{just a leaf / Base model}} + \underbrace{f_1(x)}_{DT} + \underbrace{f_2(x)}_{DT} + \cdots + f_n(x)$$

M = no. of Decision Tree

a) for $i = 1, 2, \ldots, N$ compute

$$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f = f_{m-1}}$$

i = no. of rows
m = on which DT we are working
r = residual / Pseudo Residual

Aim :- need to calculate Pseudo Residual for every Decisio Tree.

Let $m = 1$

$$r_{i1} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f = f_0}$$

You must have noticed in this Dataset we have Three rows
So, we need to calculate :-
r_{11}, r_{21}, r_{31} → 1st DT, 3 rows Pseudo residual.
↓
1st DT, 1 row Pseudo residual

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

$f(x_i) = \hat{y}_i$

$$r_{i1} = -\left[\frac{\partial L(y_i, \hat{y}_i)}{\partial \hat{y}_i}\right]_{f = f_0}$$

$$= -\left[\frac{\partial}{\partial \hat{y}_i}\frac{1}{2}(y_i - \hat{y}_i)^2\right]_{f = f_0}$$

$$= -\left[\frac{1}{2} \times 2(y_i - \hat{y}_i)\frac{\partial}{\partial \hat{y}_i}(y_i - \hat{y}_i)\right]_{f = f_0}$$

$$= -\left[(y_i - \hat{y}_i)(0 - 1)\right]_{f = f_0}$$

$$= \left[y_i - \hat{y}_i\right]_{f = f_0} = \left[y_i - f(x_i)\right]_{f = f_0} = \left[y_i - f_0(x_i)\right]$$

= observed − predicted

$r_{11} = y_1 - f_0(x_1) = 192 - 142 = 50$
$r_{12} = y_2 - f_0(x_2) = 144 - 142 = 2$
$r_{13} = y_3 - f_0(x_3) = 91 - 142 = -51$

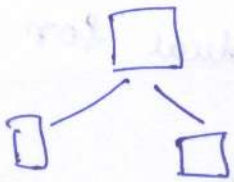
2(b) fit a regression tree to the targets Rim giving terminal regions R_{jm}, $j = 1, 2, \ldots, J_m$

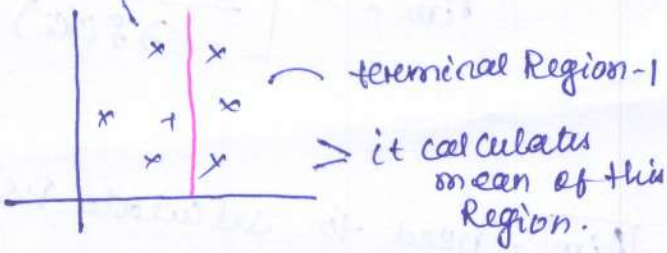
i.e

Considering three independent columns as input and Rim as o/p we need to fit a tree.

Here, what is terminal Region?

Suppose we have X | Y and DT is like this





It will cut Graph into two halves as the DT i.e one depth we have taken

R_{jm} = terminal Region

2(c) for $j = 1, 2, \ldots, J_m$, compute

$$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma)$$

Here, the aim is to calculate the o/p of terminal Regions based on above formula

$$\gamma_{11} = \arg\min_{\gamma} \sum_{x_i \in R_{11}} L(y_1, f_{m-1}(x_i) + \gamma)$$

$x_i \in R_{11} \Rightarrow R_{11}$ is terminal region

This means we need to calculate the points that are fallen into that terminal region only.

$$\gamma_{11} = \arg\min_{\gamma} \frac{1}{2} (y_i - (f_0(x_i) + \gamma))^2$$

$$\frac{dL}{d\gamma} = \frac{1}{2} \times 2 \times (y_i - f_0(x_i) + \gamma) \frac{d}{d\gamma} (y_i - f_0(x_i) + \gamma) = 0$$

$$= (y_i - (f_0(x) + \gamma))(-1)$$

$$\gamma_{11} = y_i - f_0(x) - \gamma$$

$$= 91 - 142 - \gamma = 0$$

$$\gamma = -51.00$$

XG-Boost/Extreme Gradient Boosting

> XGBoost improves upon the base of GBM framework through system optimization & algorithmic enhancements.

System optimization

1) parallelization.
2) Tree pruning
3) Hardware optimization

Algorithmic Enhancements

1) Regularization
2) Sparsity awareness
3) weighted quantile sketch
4) cross validation.

GBoost classifier

Salary	Credit	Approval	Res	New Pr
<=50K	Bad	0	0 − Pr = −0.5	0.6
<=50K	Good	1	1 − 0.5 = 0.5	-
<=50K	Good	1	0.5	-
>50K	Bad	0	−0.5	-
>50K	Good	1	0.5	-
>50K	Normal	1	0.5	-
<=50K	Normal	0	−0.5	-

Step 1:- Construct a Base model

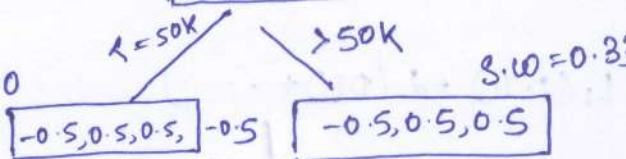
o/p will be 0,1 = $\frac{0+1}{2} = 0.5 = P_r$

Construct a tree

[−0.5, 0.5, 0.5, −0.5, 0.5, 0.5, −0.5]

→ $S.W = \frac{[+0.5]^2}{7 \times 0.25} = \frac{0.25}{7 \times 0.25} = 0.1428$

Salary



$\frac{0.5 + 0.5 + 0.5 - 0.5]^2}{5(1-0.5) + 0.5(1-0.5) + 0.5(1-0.5) + 0.5(1-0.5)]} = 0$

$S.W = \frac{[-0.5 + 0.5 + 0.5]^2}{(0.5(1-0.5)) \times 3} = \frac{0.25}{(0.25) \times (0.25) \times (0.25)} = 0.33$

Calculate the Similarity weight (S.W)

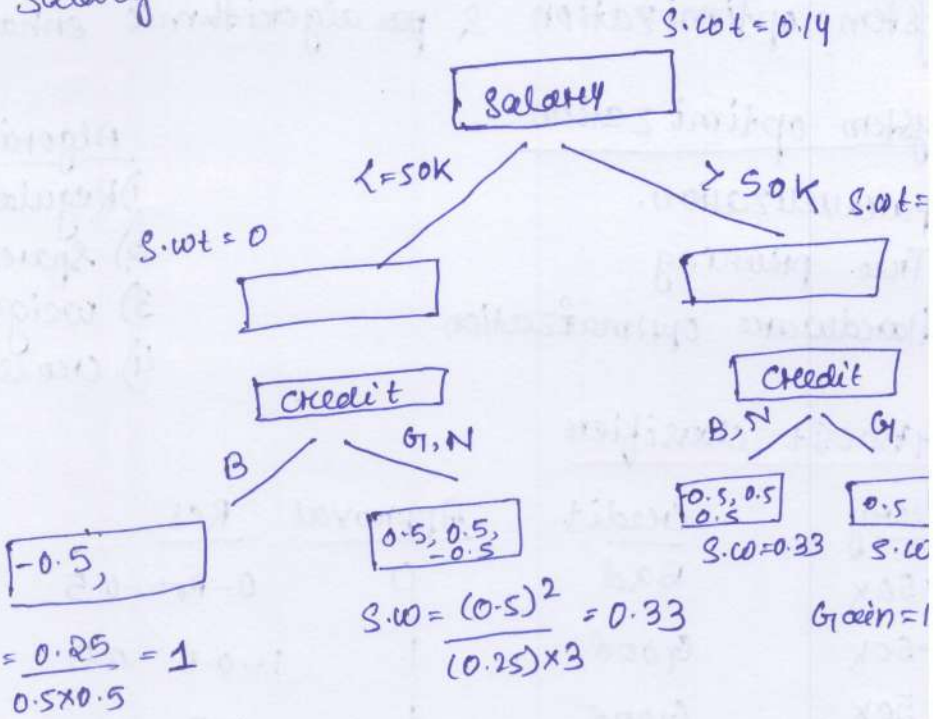
$= \frac{(\Sigma \text{ Residual})^2}{\Sigma \; Pr(1-P) + \lambda}$

Calculate Gain = (S.W of left hand side + S.W of RHS) − S.W of Root node

= [0 + 0.33) − 0.1428 = 0.1872

Which ever has more gain takes that split.

Suppose Gain is more for Salary so it takes the first split.



Here also we need to check which split it is taking and as per the gain we do the split.

$$\text{Gain} = (1 + 0.33) - 0 = 1.33$$

$$S.w = \frac{0.25}{0.5 \times 0.5} = 1$$

$$S.w = \frac{(0.5)^2}{(0.25) \times 3} = 0.33$$

→ Calculate the o/p of Base Model.

$$\log(\text{odds}) = \log\left(\frac{p}{1-p}\right) = \log(1)$$

$$= 0$$

$$= \sigma\left(0 + \underset{\uparrow}{0.1}\,\lambda_1 (\text{similarity rate of } DT_1)\right) = \sigma(0 + 0.1 \times 1) = \sigma(0.1)$$

↓ Basemodel

→ Sigmoid Activation function $= \frac{1}{1+e^{-x}} = \frac{1}{1+e^{-0.5}} = 0.6$

$$\text{XGBOOST} = \underbrace{\text{Base model}}_{\log\left(\frac{p}{1-p}\right)} + \sigma\left(\lambda_1 (DT_1) + \lambda (DT_2) + \cdots\right)$$

XG-Boost Regressor

Ex:-

Exp	GAP	Salaray	Res
2	Yes	40K	40-51.2=-11
2.5	Yes	42K	-9
3	NO	52K	1
4	No	60K	9
4.5	Yes	62K	11

① Create the Base model

$$= \frac{40+42+52+60+62}{5}$$

$$= 51.2K$$

$$S.W = \frac{\sum (Residual)^2}{No.of\ Residual + \lambda}$$

S.W = 1/(5+1) = 0.166

[-11, -9, 1, 9, 11]

EXP

<1=2 | >2

-11 | -9, 1, 9, 11

o/p = 11

let λ = 1

$S.W = \frac{(-11)^2}{1+1} = \frac{121}{2} = 60.5$

$S.W = \frac{(12)^2}{4+1} = \frac{144}{5} = 28.8$

Gap

Yes | NO

[-9, 11] o/p = 10 | [1, 9] o/p = 5

$Gain = (60.5 + 28.8) - 0.166$

$= 89.134$

∴ Like way splitting happens until unless best Gain is found.

$$o/p = 51 + \underset{\lambda}{(0.5)} \underset{T_1}{[-11]} + \alpha_2 T_2 + \alpha_3 T_3$$

↓ Base model ↓ avg of Decision Tree.

=