XgBoost

(Basic idea of all boosting tech is using errors of previous tree and improvising in present tree)

* Also called as extreme gradient boosting.
* It is an optimized gradient boosting ml library.
* Powerful algorithm with high speed & Performance
* Feasible to train on large dataset
* The core XgBoost algorithm is parallelizable that is it does parallelization within a single tree

XgBoost classifier:-

* Formation of trees were same as gradient boosting.
* It uses only Binary classification.

Dataset:-

Salary	credit	approval	Residual (~~Probability - approval~~) (approval - Probability)
<=50K	B	0	-0.5
<=50K	G	1	0.5
<=50K	G	1	0.5
>50K	B	0	-0.5
>50K	G	1	0.5
>50K	N	1	0.5
<=50K	N	0	-0.5

* Since it is classification (Binary) the probability for o/p is $1/2 = 0.5$
* There is a pseudo algorithm to follow Xgboost

① construct Tree with Root

② calculate Similarity weight

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

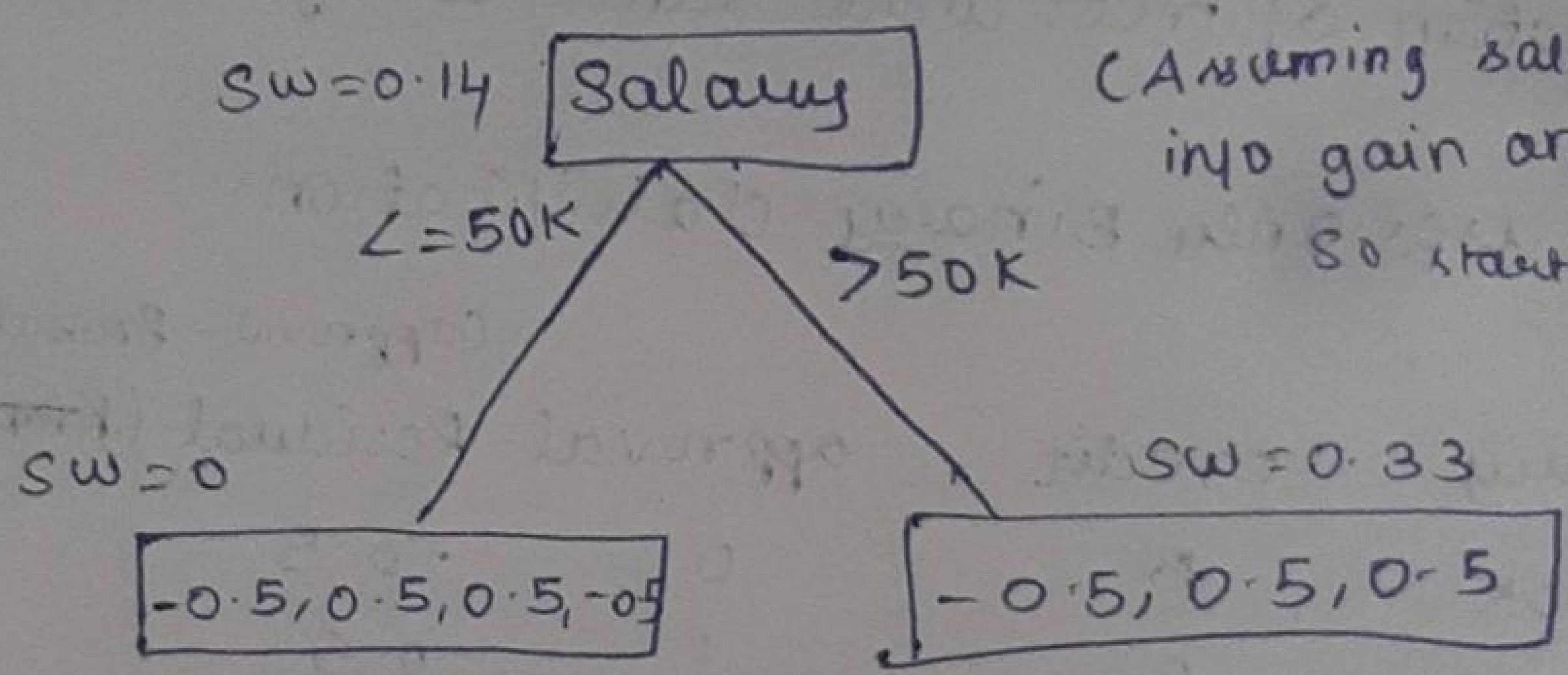
where λ is ~~an~~ a hyperparameter

③ calculate gain.

—

Constructing the base model

Res ⇒ [−0.5, 0.5, 0.5, 0.5, 0.5, 0.5, −0.5]



(Assuming salary has high info gain among other features) So starting split with salary)

* calculating similarity weight leaf nodes and parent node and updating in above tree $\boxed{\text{taking } \lambda = 0}$

for <=50K ⇒

$$\frac{-0.5+0.5+0.5-0.5}{0.5(1-0.5)+0.5(1-0.5)+0.5(1-0.5)+0.5(1-0.5)+0}$$

$= 0$

for >50K

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

$$= \frac{0.25}{0.75} = 1/3 = 0.33$$

for root node:
~~ao~~ after ~~cal~~ same calculation
SW = 0.14

calculating gain

$= 0+0.33-0.14$

$=0.21$

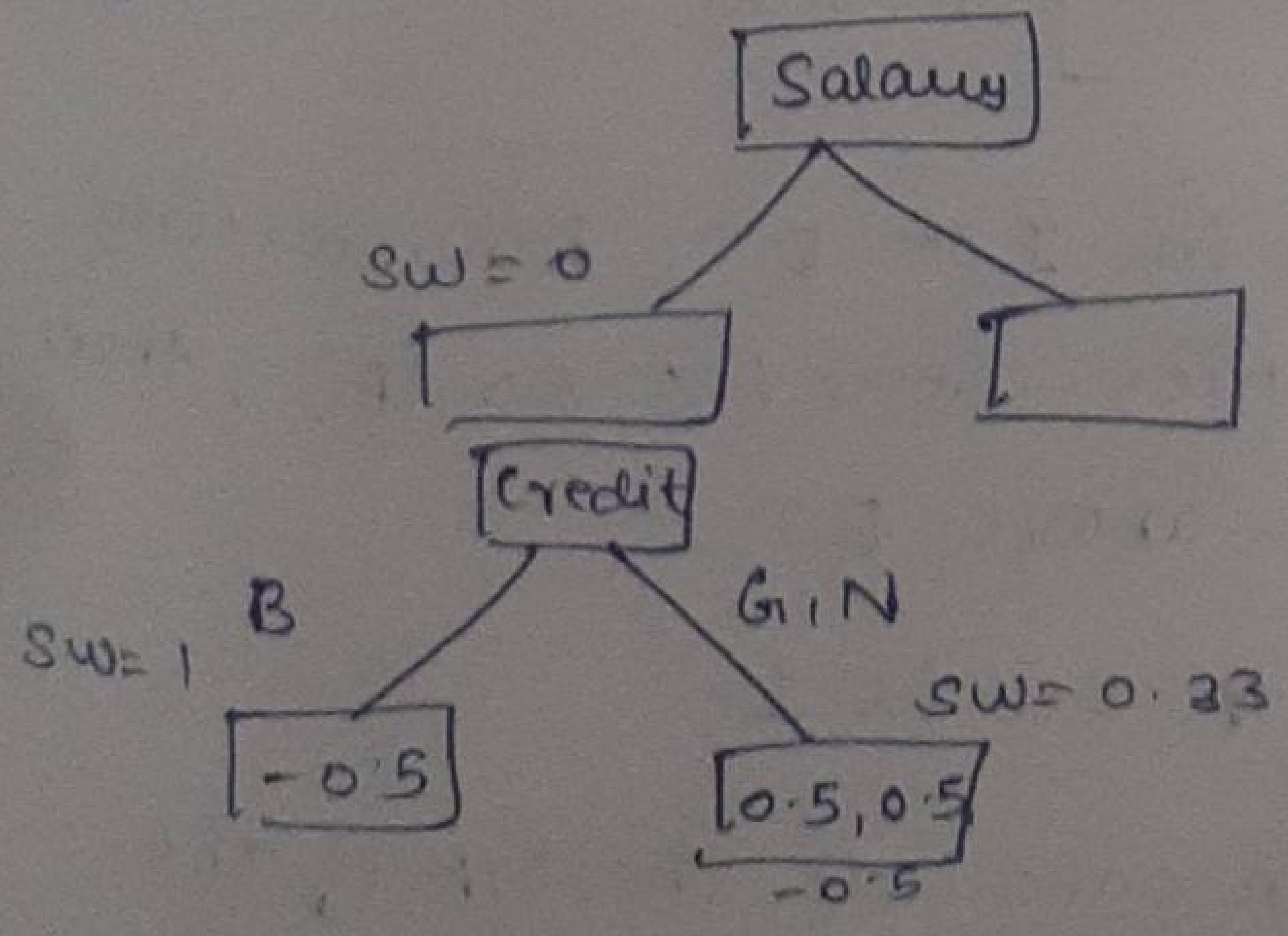
(considering salary has better gain than credit]

* Next we want to take credit feature that should be Binary classified like [B, (G, N) or (B, G), N or anyother]

and this will continue under any of the 2 leaf node of salary:

* The position and the combination of the spliting of feature will also be ~~con~~ done by calculating gain for each position (left or right) of each combination.

* Best will be selected.



=> DT - 1

Gain => ~~0.1~~ $1+0.33-0 = 1.33$

* After that if we want to continue split then it will be selected by post pruning using cover value.

Cover value ⇒ Prob(1 - Prob)

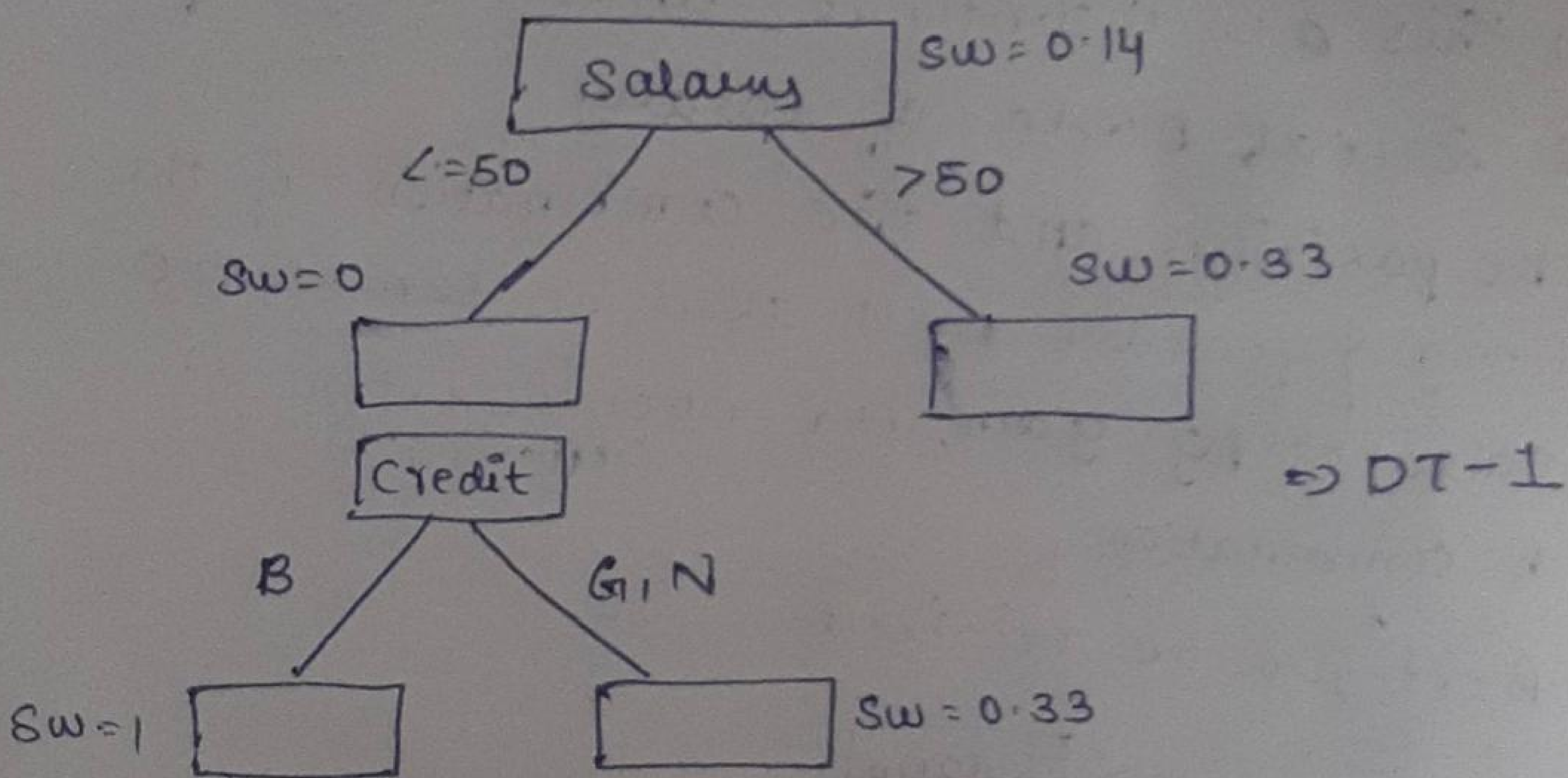
Here cover value is

$= 0.5(1 - 0.5)$

$= 0.25$

So, if my gain is less than 0.25 in any of branch then it will be pruned.

we can calc any no. of DT if residuals is calculated



* now training over in 1st DT, now testing data comes in, the entries were (<=50, B), then now follow the path in above DT...

* according to it the SW is 1

* with this we can calculate the o/p of the Base learner that is by the formula

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

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

$$= \log(1)$$

$$= 0$$

∴ The o/p for the base model is 0.

* For the base learner the prob for all records were equal, but it will change in the upcoming trees.

* The probability for each record in training data can calculated using sigmoid activation function

* Finding prob for 1st training record ($\angle = 50$, B)

$$\sigma\left(\underbrace{0}_{\text{o/p of base model}} + \underbrace{\alpha}_{\text{Learning rate (0 to 1)}}\underbrace{(SW)}_{\text{SW according to } \angle = 50,\ B}\right)$$

$$= \sigma(0 + 0.1(1))$$

$$= \sigma(0 + 0.1)$$

$$= \sigma(0.1)$$

Sigmoid function $= \dfrac{1}{1+e^{-x}} = \dfrac{1}{1+e^{-0.1}} = 0.6$

∴ 0.6 will be the new probability for 1st training record

* Like wise probabilities will be calculated for all training records.

* Now as we done earlier, we can calculate new residuals of 1st DT by (approval - new probabilities)

* By taking new residuals as o/p we can construct successive decision trees.

XgBoost Regressor :-

* The tree building process were same as XgBoost classifier

$$\left.\begin{array}{c}\text{Similarity} \\ \text{weight}\end{array}\right\} = \frac{\Sigma(\text{residual})^2}{\text{No of Residual} + \lambda}$$

Dataset :-

Exp	gap	salary	Res
2	yes	40K	-11
2.5	yes	42K	-9
3	no	52K	1
4	no	60K	9
4.5	yes	62K	11

(Salary - {avg(salary) or o/p of Base })

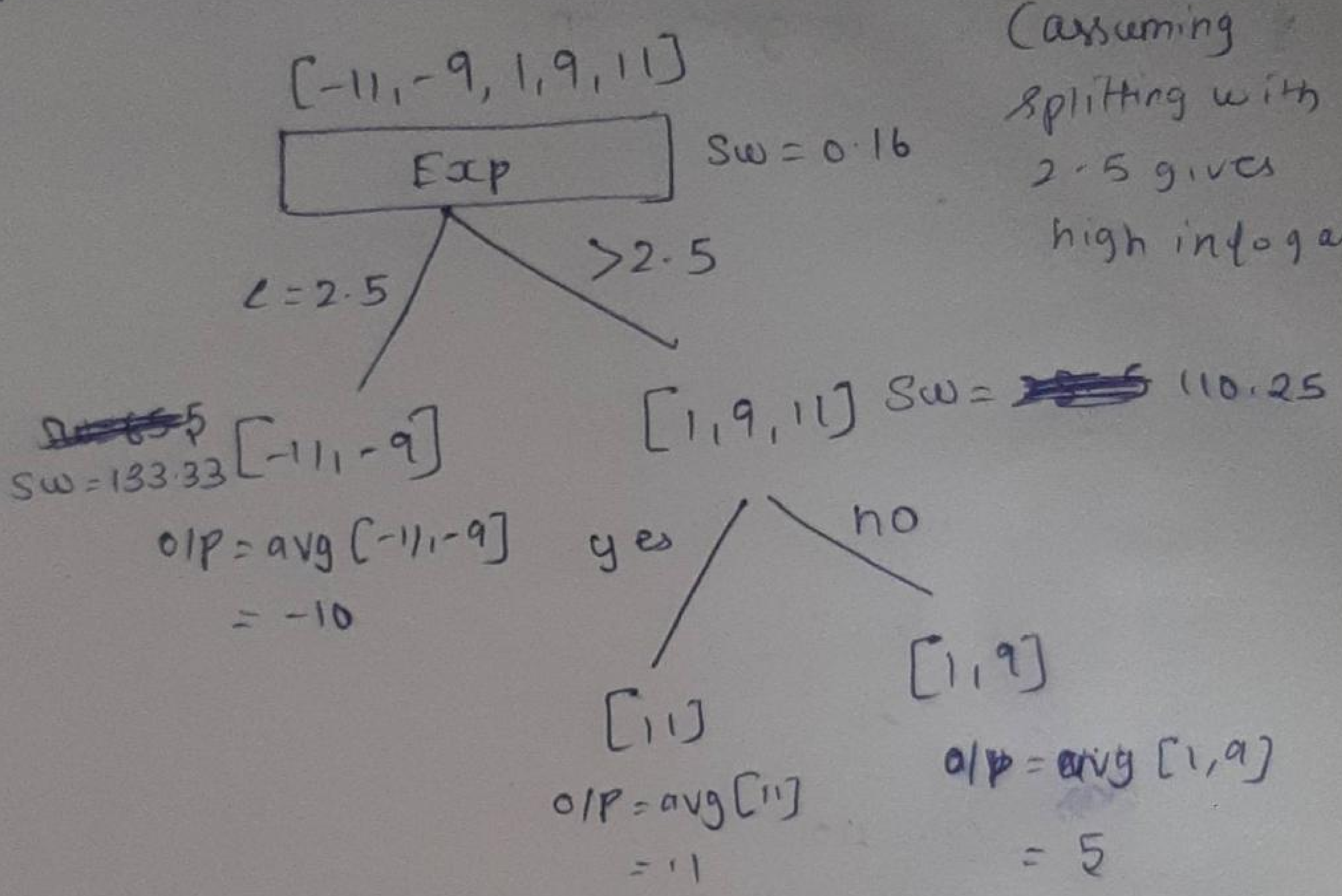
Z

* First we want to calculate o/p of base model that is avg(salary), here that is 51k

* Now (Salary - o/p of base model)

* So we will get res.

* Tree will constructed as the same process and respective sw will be calculated.



* So now we want to calculate original o/p of train data.

for ex take 1st record (2, yes) with value 2

~~So 50k + α [20]~~

So, 50 + α [-10] $\quad \alpha = 0.5$

o/p of base model

Learning rate

50 + 0.5 [-10]

50 - 5

= 46

* So like this for each record the O/P will be calculated.

* so then we can calculate ~~the~~ residuals of 1st DT by ~~[illegible]~~

(salary - new output)

* Likewise it goes on.

* In regressor we have ν which is like cover value in classifier, which is basically ~~to~~ used for post pruning.