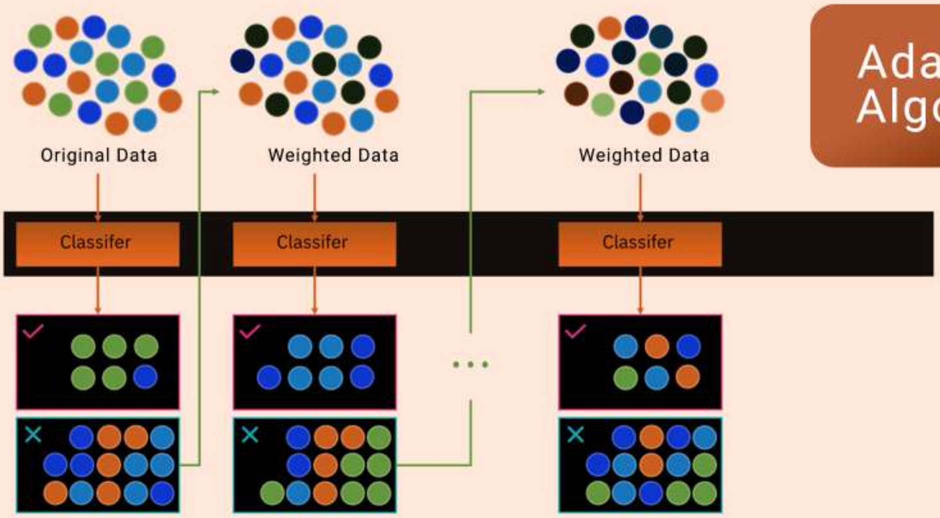
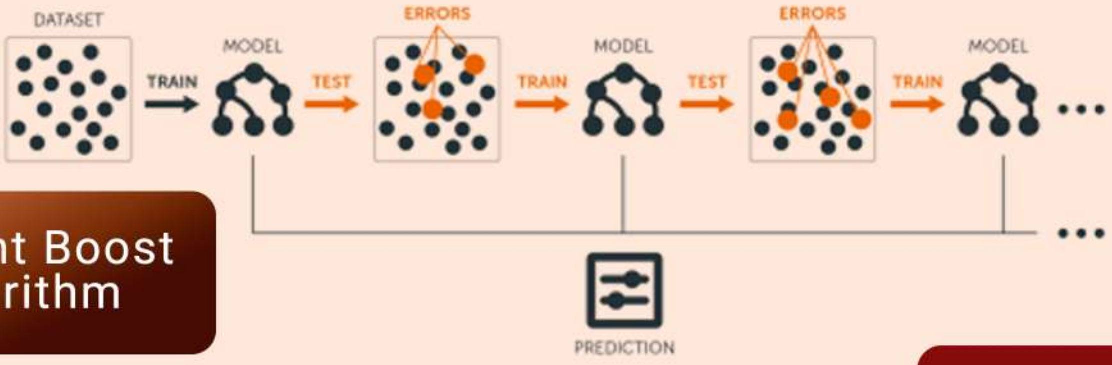
Comparision between AdaBoost and Gradient Boost Algorithm



AdaBoost Algorithm

S.No	Adaboost	Gradient Boost
1	An additive model where shortcomings of previous models are identified by high-weight data points.	An additive model where shortcomings of previous models are identified by the gradient.
2	The trees are usually grown as decision stumps.	The trees are grown to a greater depth usually ranging from 8 to 32 terminal nodes.
3	Each classifier has different weights assigned to the final prediction based on its performance.	All classifiers are weighed equally and their predictive capacity is restricted with learning rate to increase accuracy.
4	It gives weights to both classifiers and observations thus capturing maximum variance within data.	It builds trees on previous classifier's residuals thus capturing variance in data.



Gradient Boost Algorithm

BY SAHIL JOSAN

Machine Learning : Gradient Boost Algorithm

- Regression Problem.

Exp (x_1)	Degree (x_2)	Salary (y)
2	BE	50K
3	Masters	70K
5	Masters	80K
6	PHD	100K

⇒ **Steps for Gradient Boosting.**

Step 1. Create a Base model.

$\hat{y}$ = Average of Dependent Feature (y).

$$\therefore \hat{y} = \frac{50K + 70K + 80K + 100K}{4} = 75K.$$

75K will be the base model.

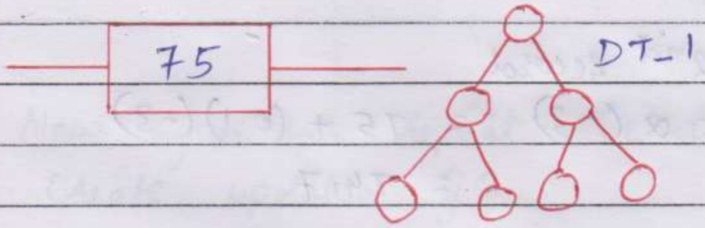
Exp(x_1)	Degree	Salary (y)	$\hat{y}$
2	BE	50K	75K
3	Masters	70K	75K
5	Masters	80K	75K
6	PHD	100K	75K

Step 2. Compute the Residuals or Errors.

$$R_1 = y - \hat{y}$$

Exp (x_1)	Degree (x_2)	Salary (y)	$\hat{y}$	R_1
2	BE	50K	75K	-25
3	Masters	70K	75K	-5
5	Masters	80K	75K	5
6	PHD	100K	75K	25

Step 3 We construct the next decision tree with inputs x_1 and x_2 and target feature R_1



I/P of DT-1 = x_1, x_2 and R_1
O/P of DT-1 = R_2

Salary (y)	$\hat{y}$	R_1 $(y-\hat{y})$	R_2 (O/P of 1st Decision Tree.)
50K	75K	-25	-23
70K	75K	-5	-3
80K	75K	5	3
100K	75K	25	20

Step 4 Now calculate predicted output. $[\hat{y} + R_2]$

$\therefore 75 + (-23) = 75 - 23 = 52$

But our intial output according to Salary feature was 50K and now updated predicted output comes 52K that means our model is overfitted. To solve this we have to include learning rate (α)

α = Learning Rate.

Predicted output $(\hat{y}) = 75 + \alpha(-23)$

Learning rate can be any value : 0.1, 0.01, 0.001 etc.
Here we assume $\alpha = 0.1$

$\therefore$ Predicted output $(\hat{y}) = 75 + (0.1)(-23)$
$= 72.7$

Similarly for 2nd record
$75 + \alpha(-3) = 75 + (0.1)(-3)$
$= 74.7$

Salary (y)	$\hat{y}$	R1	R2	Updated $\hat{y}$
50K	75K	-25	-23	72.7
70K	75K	-5	-3	74.7
80K	75K	5	3	75.3
100K	75K	25	20	77

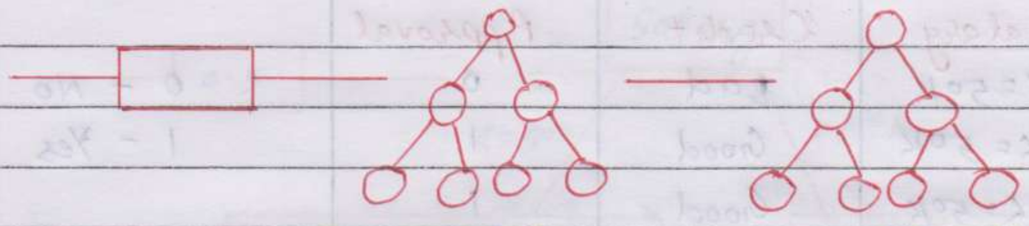
$\boxed{\text{Step5}}$ Now with the updated $\hat{y}$ again compute the Residuals or Errors.

$R3 = y - \text{Updated } \hat{y}$

Salary(y)	$\hat{y}$	R1	R2	Updated $\hat{y}$	R3
50K	75K	-25	-23	72.7	-22.7
70K	75K	-5	-3	74.7	-4.7
80K	75K	5	3	75.3	4.7
100K	75K	25	20	77	23

Step 6 Based on our new target feature R3, we will construct our next decision tree.

I/P [x_1, x_2, R_1] I/P [x_1, x_2, R_3]



O/P [R_2] O/P [R_4]

Now again we repeat the whole steps.

1. Create updated $\hat{y}2$
2. Create output feature R5 by substracting Salary (y) − Updated $\hat{y}2$
3. Ten again provide this R_5 as the input to decision tree [3] and R6 will be the output and so on.

This will keep on going until the value of R keeps on decreasing or the value of n-estimator is reached.

Final Formula :-

$$F(x) = h_0(x) + \alpha_1[h_1(x)] + \alpha_2[h_2(x)] \ldots\ldots \ldots\ldots \alpha_n[h_n(x)]$$

↑ [Base Learner]

$$\boxed{F(x) = \sum_{i=0}^{n} \alpha_i[h_i(x)]}$$ ⇒ Gradiant Boost Algorithm.

SAHIL JOSAN

XG Boost : Extreme Gradient Boost Algorithm

- Binary Classification Problem.

Salary	Credit	Approval
<=50K	Bad	0
<=50K	Good	1
<=50K	Good	1
>50K	Bad	0
>50K	Bad	1
>50K	Normal	1
<=50K	Normal	0

0 - No
1 - Yes

Step 1 Create a Base model:

It is a binary classification problem so the average output will be 0.5

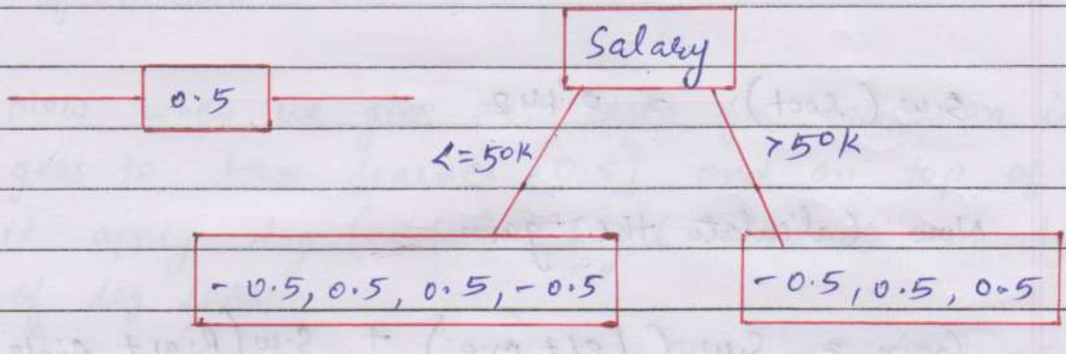
$\therefore \hat{y} = 0.5$

Step 2 Compute the Residuals or Errors.

$R_1 = y - \hat{y}$

Salary (x_1)	Credit (x_2)	Approval (y)	$\hat{y}$	R_1 ($y-\hat{y}$)
<=50K	Bad	0	0.5	-0.5
<=50K	Good	1	0.5	0.5
<=50K	Good	1	0.5	0.5
>50K	Bad	0	0.5	-0.5
>50K	Good	1	0.5	0.5
>50K	Normal	1	0.5	0.5
<=50K	Normal	0	0.5	-0.5

Step 3 We construct decision tree in sequence with input Salary (X_1), Credit (X_2) and o/p (R_1)



Step 4 We calculate Similarity weight (S.W)

$$\boxed{\text{Formula} = \frac{(\sum \text{Residuals})^2}{\sum Pr(1 - Pr)}}$$

S.W (<=50K) [Left Side]

$$S.W(\le 50K) = \frac{(-0.5 + 0.5 + 0.5 - 0.5)^2}{0.5(1-0.5)^2 + 0.5(1-0.5)^2 + 0.5(1-0.5)^2 + 0.5(1-0.5)^2}$$

$\boxed{Pr : \text{Base Learner o/p}}$

$\therefore S.W(\le 50K) = 0$

Similarly

S.W (>50K) [Right Side]

$$S.W(> 50K) = \frac{(-0.5 + 0.5 + 0.5)^2}{0.5(1-0.5) + 0.5(1-0.5) + 0.5(1-0.5)}$$

$$S.W(> 50K) = \frac{0.25}{0.75} = 0.33$$

$$S.w\,[Salary]\,(root) = \frac{(-0.5+0.5+0.5-0.5-0.5+0.5+0.5)^2}{0.5(1-0.5)+0.5(1-0.5)+0.5(1-0.5)+0.5(1-0.5)+0.5(1-0.5)+0.5(1-0.5)+0.5(1-0.5)}$$

$$S.w\,(root) = 0.142$$

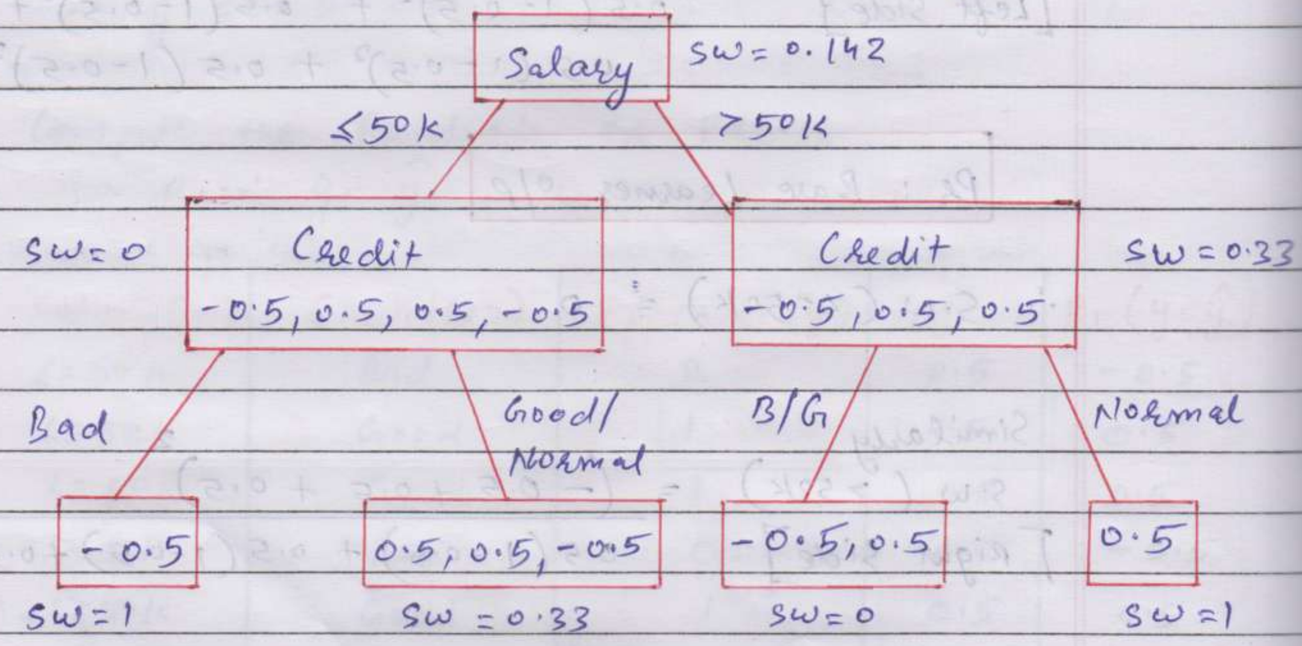
Now Calculate the gain

$$Gain = S.w\,(\text{Left one}) + S.w\,(\text{Right side}) - S.w\,(root)$$

$$= 0 + 0.33 - 0.142$$

$$Gain = 0.19$$

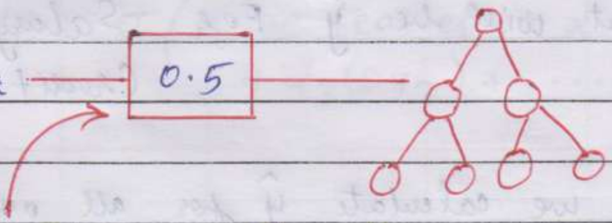
Now Break DT_1 into more depth using Credit Feature



SW: Similarity weight.

we can calculate information Gain by using Similarity weight. and the best one will be selected which will have the maximum information Gain.

⇒ Now when we give test data for validation it goes to base learner [0.5] and on top of it, it apply log (odds). Log (odds) is one type of log loss.

Test data 

$$\text{Log (odds)} = \log\left(\frac{P}{1-P}\right) = \log\left(\frac{0.5}{1-0.5}\right)$$

$$= \log 1 = 0$$

∴ Model output $= \sigma\left[0 + \alpha(1)\right]$

Here σ : Sigmoid Activation Function.

0 : Log (odds) on Base Learner [0.5]

(1) : Because Similarity weight = 1 For Salary ≤ 50K and Credit (Bad)

$$\text{Sigmoid Activation Function } (\sigma) = \frac{1}{1+e^{-z}}$$

Here $z = 0 + \alpha(1)$

where assume $\alpha = 0.1$

$\therefore$ Model output $= \sigma(0 + (0.1)1)$

$= \sigma(0.1)$

$= \dfrac{1}{1 + e^{-0.1}}$

Model output = 0.52.

This output will be $\hat{y}$ For Salary = ≤ 50K
Credit = Bad

Similarly we calculate $\hat{y}$ for all our records using similarity weight.

Salary	Credit	Approval	R_1	Updated $\hat{y}$	R_2
<=50K	Bad	0	-0.5	0.52	-0.52
<=50K	Good	1	0.5	0.58	0.42
<=50K	Good	1	0.5	0.58	0.42
>50K	Good	0	-0.5	0.5	-0.5
>50K	Bad	1	0.5	0.5	0.5
>50K	Normal	1	0.5	~~0.73~~ 0.52	0.48
<=50K	Normal	0	-0.5	0.58	-0.58

R_2 = Approval (y) − $\hat{y}$ (Updated)

Now next decision tree will be trained with salary and credit as the independent feature and R_2 as the dependent feature.