Gradient Boosting

This is work very well for classification and Regressor both problem.

Dataset

Regression Algorithms

Experience	Degree	Salary
2	B.E.	50k
3	master	70k
4	master	80k
5	Ph.D.	100k

In Gradient Boosting we create first a base model and this will be only a mean of target coloumn

(1) I/P → Base model → O/P

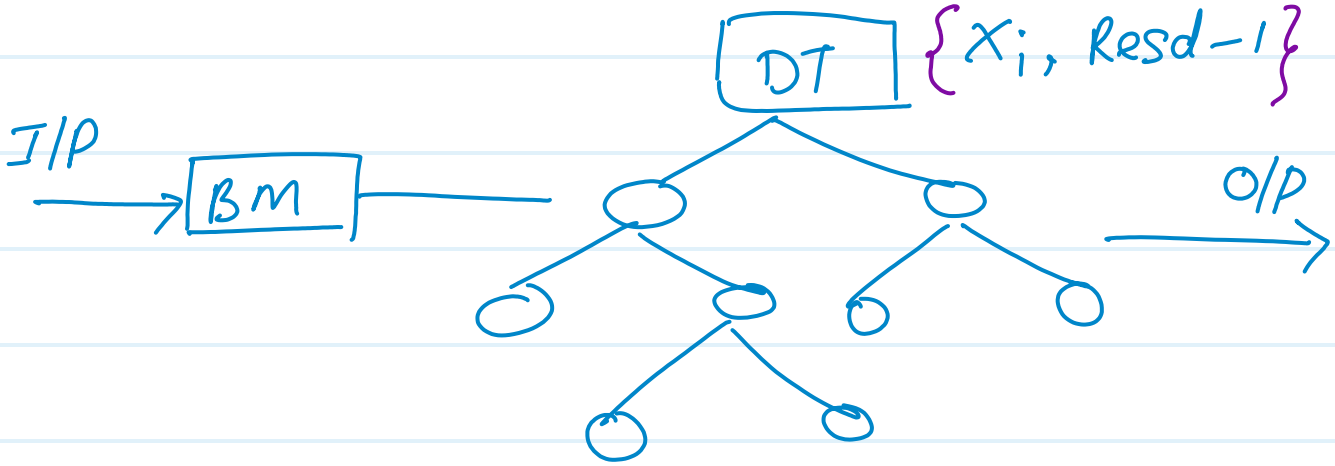
$$\Rightarrow \frac{50+70+80+100}{4} = 75$$

② Compute Residual, Error

exp	Deg.	salary	$\hat{y}$	$(y-\hat{y})$ pred-1	↓ Resd-1
2	B.G.	50	75	-25	-23
3	MS	70	75	-5	-3
4	MS	80	75	5	4
5	PhD	100	75	25	22

Assumed After DT

③ Decision tree, consider input x_i and o/p Res-1



I/P → BM — DT → O/P

2y/Exp → 75 + (-23) = 52

Decision tree predict for first point of dataset is 52 and it is very near to the actual value. It means we overfitted the model.

Due to this overfitting of this model, Gradiend boosting use sequanc of DT model with the learning α in every DT so it can be learning gradully for best model Range of α would be 0 to 1,

So

$$75 + \alpha(-23)$$

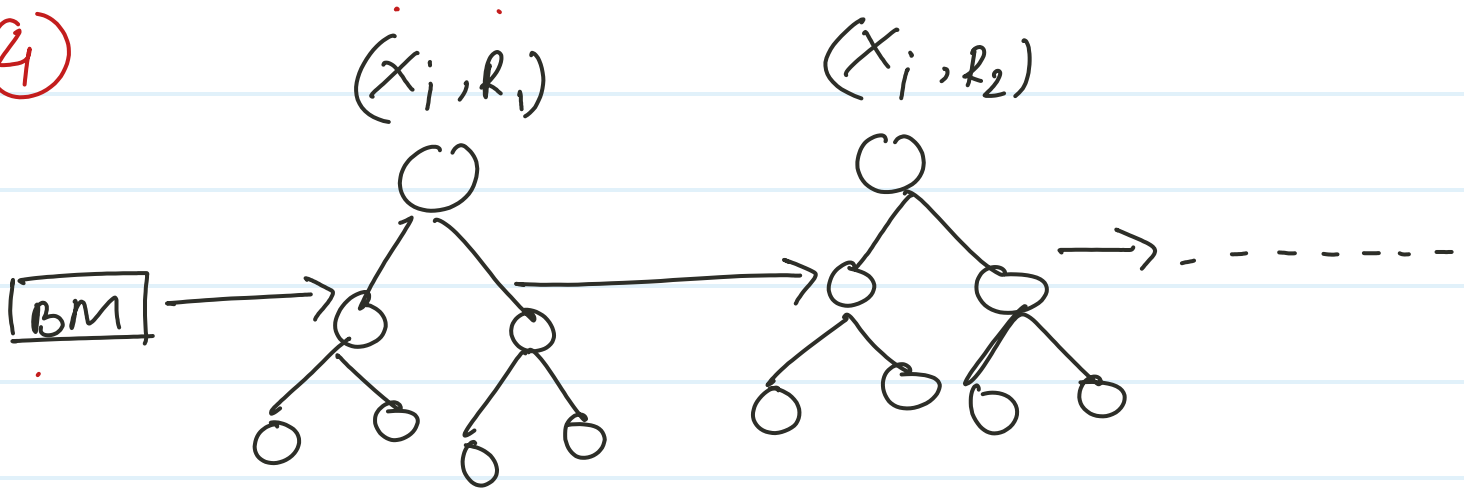
we just consider value of $\alpha = 0.01$

$$75 + (0.01)(-23)$$

2 → $75 - 0.23 = 74.77$
3 → $75 - (0.01)(-3) = 74.97$
4 →
5 →

	X_i	Y'	R_1	$\hat{Y}_2$	$(y-\hat{Y}_2)$ Pred-2	R_2
exp	Deg.	sal	pred-1	Red-1		
2	BE.	50	-25	-23	74.77	50 - 74.77	-24.77
3	MS	70	-5	-3	74.97	70 - 74.97	- 4.97
4	MS	80	5	4			
5	PhD	100	25	22			

(4)



math. function

$$F(x) = \alpha_0 h_0(x) + \alpha_1 h_1(x) + \alpha_2 h_2(x) + \cdots + \alpha_n h_n(x)$$

$[\alpha_0, \alpha, \alpha_2, n_\eta]$ This is mine learning rate

$$F(x) = \sum_{i=0}^{h} \alpha_i h_i(x)$$

At the final Answer we Add all of DT

Final o/p = $BM + \alpha DT_1 + \alpha DT_2 + \alpha DT_3 + \cdots DT_\eta$

Pros –

① Can handel missing value
② Resistant for overfitting
③ Handel non-linear dataset

Cons –

① Slower to train compared RF
② sensitive for outlier and noisy data
③ Required carefull hyperparameter.

Gradient Boosting Classification

Like popcorn	Age	Favorite color	Loves Trolls
Y	12	Blue	Y
Y	87	Green	Y
N	44	Blue	N
Y	19	Red	N
N	32	Green	Y
N	14	Blue	Y

We have consider Threshold 0.5 for binary class

For classification we need to use odds ratio

$$\text{odds ratio} = \frac{\text{Probability of success}}{\text{probability of failure}}$$

odds ratio = $\log\left(\frac{4}{2}\right) = \log(2)$

log naturally $\Rightarrow \ln(2) \approx 0.6931 \Rightarrow \boxed{0.7}$

$\boxed{0.7}$

output of first tree in sequence

After this calculate the probability of loving trolls

$$\text{probability of loving trolls} = \frac{e^{\text{odd ratio}}}{1 + e^{\text{odd ratio}}}$$

see gradient boosting classification uses logistic log function as above given

$$= \frac{e^{\log\left(\frac{4}{2}\right)}}{1 + e^{\log\left(4/2\right)}}$$

$$\Rightarrow \ln\left(\frac{4}{2}\right) = \ln(2)$$

$$\Rightarrow e^{\ln(4/2)} = e^{\ln(2)}$$

property of exponetial Algorithms

$$e^{\ln(x)} = x \quad \text{for } x > 0. \text{ Thus}$$

$$e^{\ln(2)} = 2$$

$$\frac{e^{\ln(2)}}{1 + e^{\ln(2)}} = \frac{2}{1+2} = \frac{2}{3} \approx 0.666$$

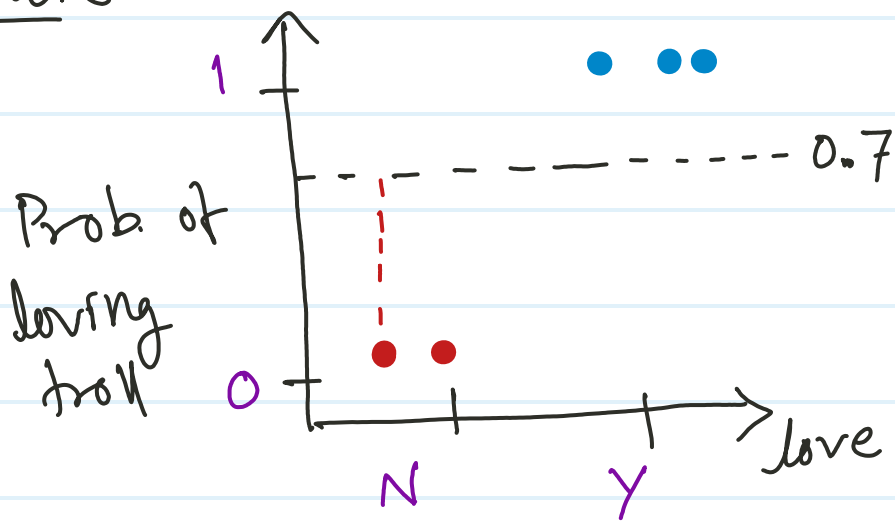
// Approximately we can write 0.7

Pseudo Residual => Residual = (observed - Predicted value)

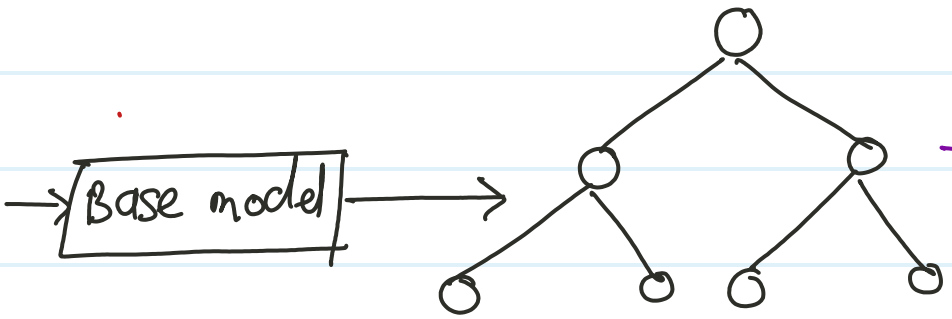
now probability of loving troll is 0.7

LP	Age	FC	Loves troll	R_1 Residual	Predi Prob.	R_2 Residual
Y	12	B	Y	1 - 0.7 = 0.3	0.9	1 - 0.9 = 0.1
Y	87	G	Y	1 - 0.7 = 0.3	0.5	1 - 0.5 = 0.5
N	44	B	N	0 - 0.7 = -07	0.5	0 - 0.5 = -05
Y	19	R	N	0 - 0.7 = -0.7	0.1	0 - 0.1 = -0.1
N	32	G	Y	1 - 0.7 = 0.3	0.9	1 - 0.9 = 0.1
N	14	B	Y	1 - 0.7 = 0.3	0.9	1 - 0.9 = 0.1

Residual calculate



In the above table we calculate Residual.



— Predicted prob. after making tree. given in table.

for classification boosting probability formula is-

$$\frac{\sum \text{Residual}}{\sum [\text{Previous Prob}_i \times (1 - \text{Previous Prob}_i)]}$$

Let's consider

Likepop	age	f.c.	Love bolls
Y	25	4	??

$\Rightarrow BM + \alpha(m_1) + \alpha_2(m_2) + \alpha_3(m_3)$

$\Rightarrow 0.7 + 0.8(1.4) + 0.8(0.6) = 2.3$

$$\text{probability} = \frac{e^{2.3}}{1+e^{2.3}} = 0.9$$

since we have threshold 0.5

$0.9 > 0.5$

This is love to bolls class.

In same way need to build many tree and it will give us final result.

usually Gradient boosting use tree between 8 to 32 leaves for training.

sklearn

XGBoost

Xtream Gradient Boosting.