Adaboost

* Adaptive Boosting
* Yoav Freund and Robert Scapire in 1996
* OG Boosting algorithm
* Godel Prize in 2003
* famous due to high performance
* No high-level mathematics
* Doesn't require much tuning unlike GB & XGBoost
* weights assign → rows & weak learner

* Different

GB/XGBoost

- focus on residuals / errors of previous weak learners

1st weak learner → 3
2nd weak learner → 2
3rd " " → 1

* Minimize error

Adaboost

focus on misclassified rows/samples/instances

* Correct them

10 rows → 5 predicted ✓
5 rows → 3 predicted ✓
2 rows → 1 predicted

General Working

Already discussed

Mathematical

(w) (e) (w_{new})

Index	X_1	X_2	Y	Initial weight	$\hat{Y}$	error	New weights
1	1	2	3.5	0.167	3.0	0.5	0.195
2	2	1	2.0	0.167	5.0	-3.0	0.055
3	1	3	4.0	0.167	3.0	1.0	0.153
4	3	2	6.0	0.167	5.0	1.0	0.153
5	2	3	5.0	0.167	5.0	0.0	0.250
6	3	1	4.5	0.167	5.0	-0.5	0.195

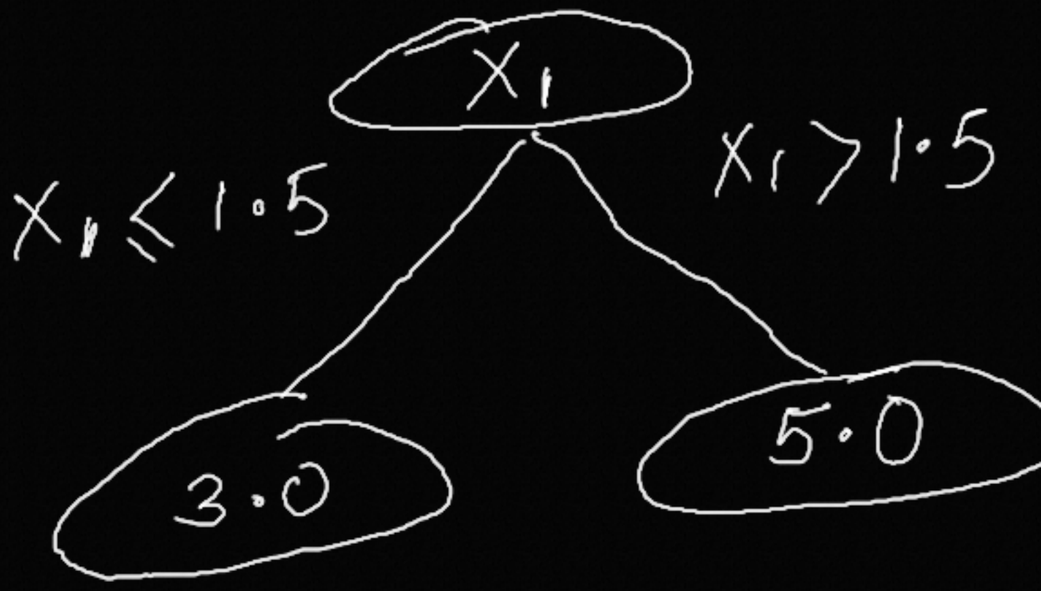
① Initialize weights to all rows

$$\text{Initial weight} = \frac{1}{n} = \frac{1}{6} = \underline{\underline{0.167}}$$

$n \rightarrow$ no of rows

② Train first weak learner

Do prediction ($\hat{y}$)



③ Calculate error

$$\text{error} = Y - \hat{Y}$$

(e) actual predicted

④ Calculate <u>weighted error</u> for weak learner

$$E_1 = \sum w \times |e|$$

$$E_1 = (0.167 \times 0.5) + (0.167 \times 3.0) + (0.167 \times 1.0) + (0.167 \times 1.0) + (0.167 \times 0.0) + (0.167 \times 0.5)$$

$$E_1 = 1.002$$

⑤ Calculate alpha (weight of weak learner)

$$\alpha_1 = \frac{1}{2} \ln\left(\frac{1-E}{E}\right)$$

$$\alpha_1 = \frac{1}{2} \ln\left(\frac{1-1.002}{1.002}\right)$$

$$\alpha_1 = -0.5$$

⑥ Update weights

$$w_{new} = w_{old} \times e^{\alpha_1 \times |e|}$$

error

↳ exponent

$$w_{1\,new} = 0.167 \times e^{-0.5 \times 0.5} = 0.130$$

$$w_{2\,new} = 0.167 \times e^{-0.5 \times 3.0} = 0.037$$

w_3 w_6.

w_4

w_5

(7) Normalization of weights

Total sum = 0.130 + 0.037 + 0.102 - - - -

= 0.668

Use normalized weights, w_3, w_4, w_5, w_6

$$w_{1\,new} = \frac{0.130}{0.668} = 0.195$$

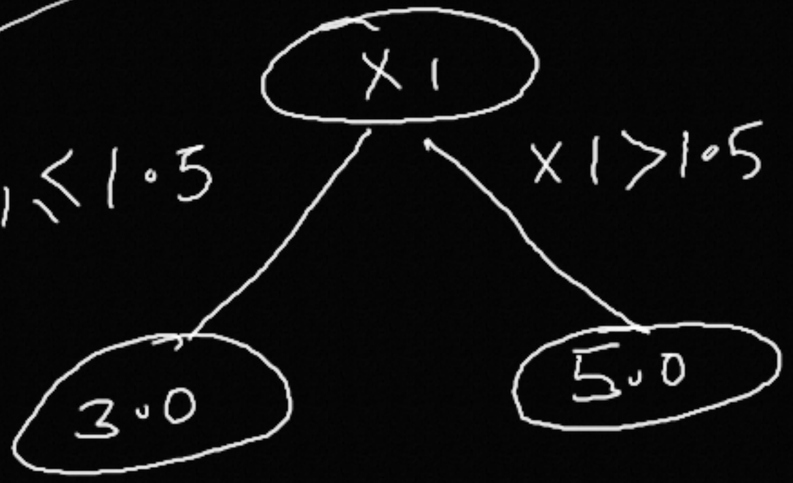
$$w_{2\,new} = \frac{0.037}{0.668} = 0.055$$

Repeat Steps 2 – 7 (untill misclassified rows become less)

Training complete

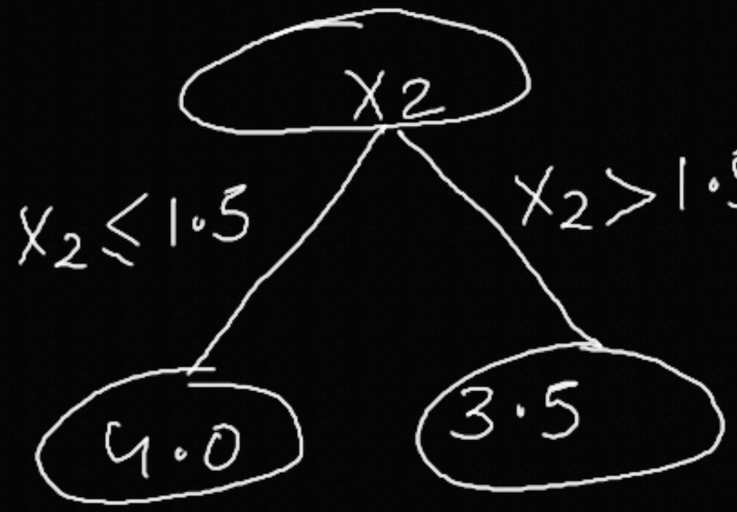
Prediction

WL1



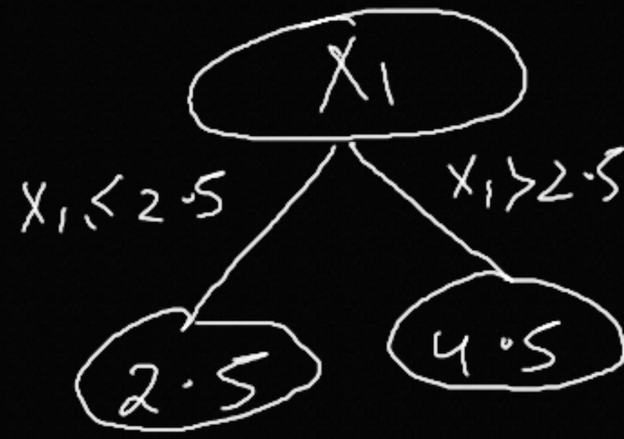
$\alpha_1 = 0.5$

WL2



$\alpha_2 = 0.3$

WL3



$\alpha_3 = 0.2$

User input $\Rightarrow X_1 = 2$

$X_2 = 2$

* Get prediction of each WL.

WL1 ⇒ $h_1 = 5.0$

WL2 ⇒ $h_2 = 3.5$

WL3 ⇒ $h_3 = 2.5$

* Calculate weighted sum

$$\hat{y} = \alpha_1 \times h_1 + \alpha_2 \times h_2 + \alpha_3 \times h_3$$

$$= 0.5 \times 5.0 + 0.3 \times 3.5 + 0.2 \times 2.5$$

$$= 4.05 \longrightarrow \text{output}$$

When to Use

Simpler & easy as compared to GB & XGBoost

Not much tuning reqd.

Simpler problems

When not to Use

Data has outliers Don't use.

Not Powerful as Compared to GB & XG Boost.

Adaboost → GB. → XGBoost.

OG algorithm

Moderate problem

Complex problem

Old is Gold

Boosting one of first algorithm

Simple problem