study notes: XGBoost Regression

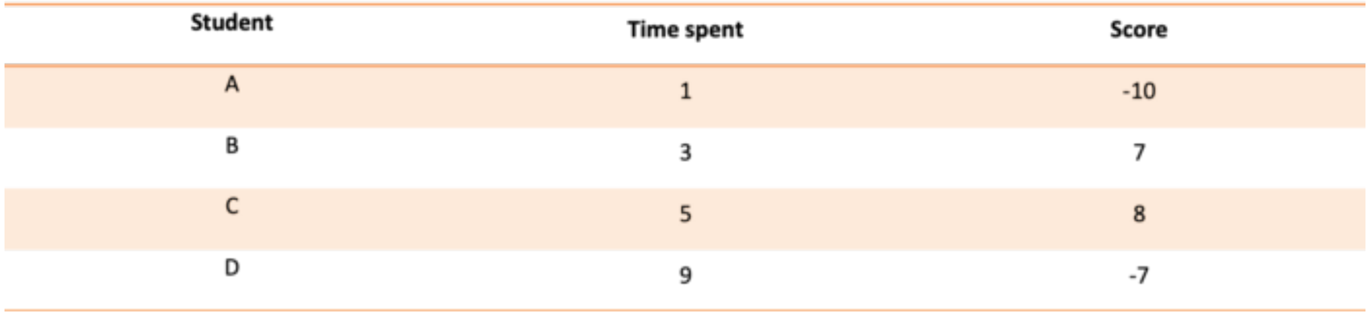
Cheryl Lai Jan 10, 2020 · 5 min read ★

> *“there is only one path to happiness, and that is in giving up all outside of your sphere of choice, regarding nothing else as your possession, surrendering all else to God and Fortune .”— EPICTETUS*

XGBoost stands for extreme gradient boosting. A regular boosting algorithm is an ensemble technique to train multiple weak learners sequentially to form a strong learner. A weak learner is usually a simple model (taking the mean or a small decision tree with few splits) that is highly interpretable. Each model learn from the error made by the previous model until no improvement can be made. The prediction on each model is then added to make up the final prediction.

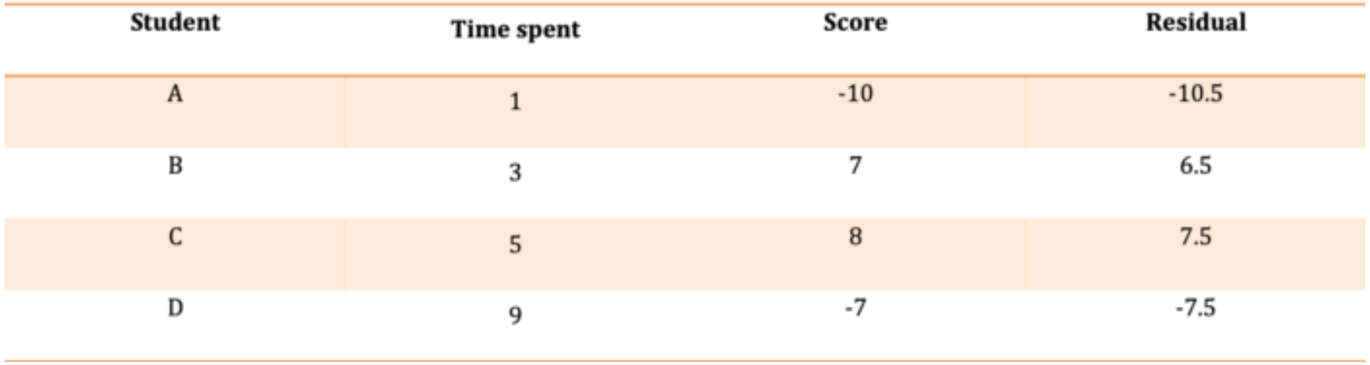
Compared to the regular boosting model, XGBoost contains an additional ***regularisation parameter*** that can improve the stability and accuracy of the model. The model is also **sparsity aware** and handles missing values efficiently by automating data imputation based on the training loss.

A sample dataset of time spent studying and the test result (target variable) for each student as shown below.



Student	Time spent	Score
A	1	-10
B	3	7
C	5	8
D	9	-7

With an initial prediction of 0.5 (ŷ = 0.5), the **residual** can be calculated as y-ŷ (table below).

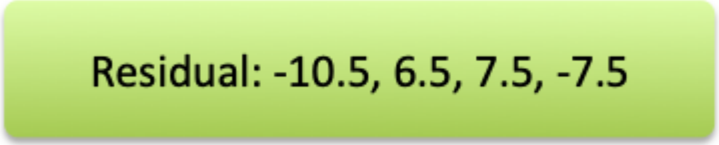


Student	Time spent	Score	Residual
A	1	-10	-10.5
B	3	7	6.5
C	5	8	7.5
D	9	-7	-7.5

Like regular gradient boosting, the XGBoost regression tree is trained on the residual.

XGBoost Regression Tree

1.Create a weak learner (DT) with a root node consisting of all the residual values.



Residual: -10.5, 6.5, 7.5, -7.5

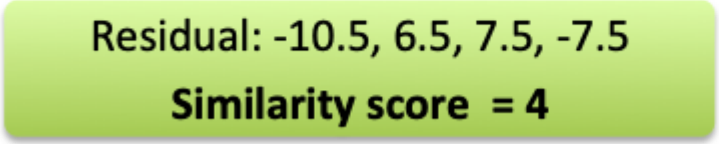
2.Calculate the similarity score within each node using the formula:

$$Similarity\ Score = \frac{(\Sigma Residual)^2}{\#\ of\ Residual +\ \lambda}$$

eqn 1

where λ is the Regularization parameter.

when λ = 0: similarity score = 4 for the residual in the root node.

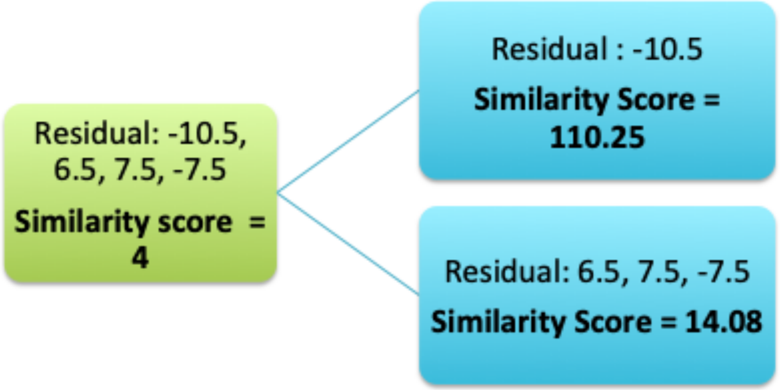


Residual: -10.5, 6.5, 7.5, -7.5

Similarity score = 4

3.Calculate the similarity score for each node from its first tree split.

- When values of the residual are very different, the values will cancel out and the similarity score will be low (eg. (+7.5–7.5)/2 results in 0).
- Similarity score will be high when $\#\ of\ Residual = 1$



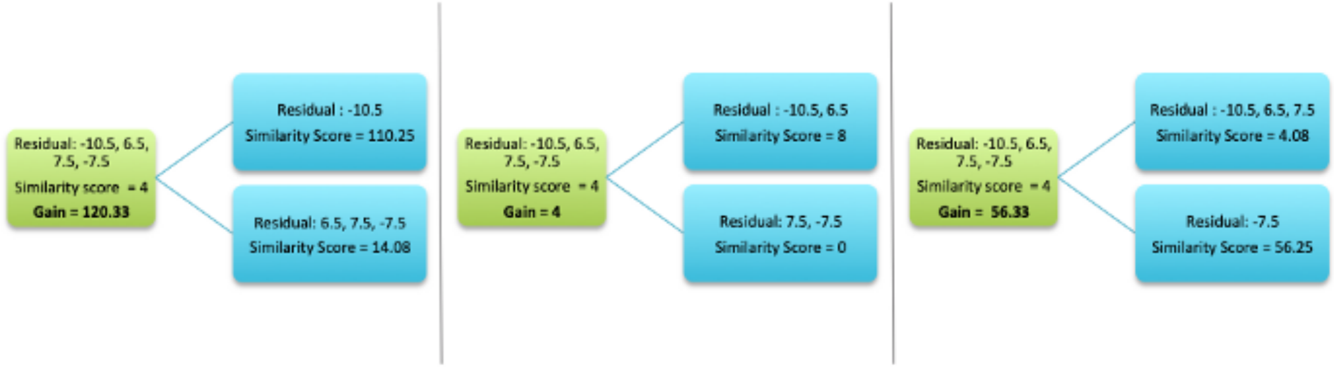
4.To calculate how much better the split is able to classify than the root node by using the formula

$$Gain = Similarity_{left\ leaf} + Similarity_{right\ left} - Similarity_{root}$$

eqn 2

The gain from the first split = 120.33.

5.The best split is obtained by comparing the gain across different ways of grouping the residuals.



Comparing the 3 different trees above, the tree on the left has the highest gain and is therefore the best split.

6.Using the tree with the highest gain, each node will split into further sub-nodes. The nodes will stop splitting when it has only 1 residual left or based on the user defined min number of sample data in each node, max iterations or tree depth.

In this example, the tree will stop at 2 splits.

7.Tree pruning prevents overfitting by comparing the gain value with the complexity parameter γ (defined by user, in this eg γ = 130). A branch containing the terminal node is prune when gain < γ (or gain-γ = negative).

Pruning will start from the lowest branch. If gain > γ, the branch and branches before it will not be pruned.

If the gain < γ, the lowest level branch is prune and the model will check the pruning conditions on the previous level.

Since the gain on the lowest branch is 140.17 and more than γ (= 130), it will not be removed.

8.The accuracy and stability of the model can be further improved by using the regularization parameter.

when λ = 1, the similarity score shrinks (higher # of residual, less impact of λ on the similarity score) .

Lower values of the similarity score will also decrease the value of the gain and lead to more frequent pruning with the same γ. This prevents overfitting (*note: too much pruning may lead to underfitting*).

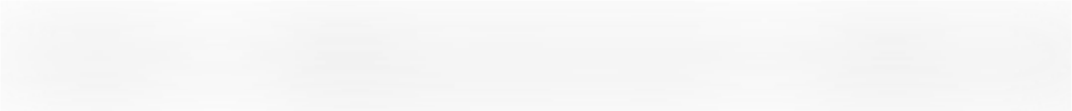
9.The output value from each model can be calculated using the formula

eqn 3

In regular GBM, the output value is simply the mean of all the residual value within each terminal node without λ.

A non-zero λ in eqn 3 will decrease the output value and its contribution to the final prediction.

10.The final prediction can be computed using the formula



eqn 4

where ε is the learning rate (default 0.3).

The output value (Prediction in the table below) is computed using eqn 4, with the initial prediction value of 0.5 and default ε of 0.3.

Since | Residual | > | Residual 1|, it shows that the prediction from the second model is better than the initial prediction and it will improve gradually with each iteration.

11.A second tree is built and fitted on the values of Residual 1 and the process will repeat until the residual values become very small, or when a maximum iteration is reached (user defined).

— The presence of a λ parameter shrinks the similarity score, gain and output value. The smaller values prevents overfitting and increases accuracy and stability (more pruning + smaller contribution from each tree)