XGBoost

20 July 2020 19:55

Gradient boost is an algorithm that optimises a prediction model on a specified loss function. Optimisation occurs be evaluating the gradient (first derivative) of the fitted function and chooses to move in the direction that has a lowest gradient - i.e. Closer to the bottom in terms of loss.

Limitation - gradient based optimisation depends on us computing the gradient of the loss function, our loss function and model must be fully differentiable (they must be a smooth, continuous function).

Step 1 - set some "weights" or Beta values for model
Step 2 - estimate the loss function
Step 3 - Differentiate, check if positive/negative
Step 4 - If positive then decrease weights, if negative increase weights
Step 5 - repeat

What "step size" should weights be changed?
Shouldn't be constant, as this could mean that the minima is not reached when close (step too large). So step size should be large when far away from minima but small when it's really close.

Loss Function
Log Loss is used for binary because the value tends towards infinity when the results are incorrect, if MSE was used the penalty is much less so could be the preferred solution.

Stands for Extreme Gradient Boosting and is not regression. It is one algorithm within a set known as Gradient Boost algorithms. The system is super fast and can train a model much faster than any other competitors. In 2017 it was used for 17/29 of the winning solutions on Kaggle.

It is so fast for these reasons:

- a novel tree learning algorithm is for handling sparse data (lots of 0's)
- a theoretically justified weighted quantile sketch procedure enables handling instance weights in approximate tree learning
- Parallel and distributed computing makes learning faster which enables quicker model exploration
- Exploits "Out-of-Core" computation

References:

- https://arxiv.org/pdf/1603.02754.pdf
- https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d
- https://xgboost.readthedocs.io/en/latest/tutorials/model.html

Worked Examples:

- https://xgboost.readthedocs.io/en/latest/R-package/xgboostPresentation.html
- https://www.kaggle.com/rtatman/machine-learning-with-xgboost-in-r

How to Implement:

- Data must be split to test and train
- Must be a **Matrix** and not a dataframe

To get our dataframe ready to be fed into the xgboost function, we needed to:

- Remove information about the target variable from the training data
- Reduce the amount of redundant information
- Convert categorical information (like country) to a numeric format
- Split dataset into testing and training subsets
- Convert the cleaned dataframe to a Dmatrix

How decision trees work:

The tree ensemble model consists of a set of classification and regression trees (CART). Here's a simple example of a CART that classifies whether someone will like a hypothetical computer game X.