XGBoost Algorithm Overview

Introduction

The XGBoost algorithm is a very popular and efficient method for running a Gradient Boosting Machine (GBM). It improves upon the standard GBM with better local interactions, allows for more regularization, handles missing values, and has many computational optimizations in modern packages. This overview will build from Decision Tree to GBM to XGBoost.

Decision Tree

Decision trees find signal through the following process:

1) Evaluate all possible (or roughly all) splits of data across all features.
2) Choose the split which yielded the largest reduction of loss (largest signal); typically mean squared error for regression tasks.
3) Split the data into 2 groups based on that split.
4) Repeat this process on the 2 subgroups until stopping rule is triggered. Stopping rules can include:
    a. Minimum gain threshold
    b. Maximum depth
    c. Minimum sample size

Sample Toy Data (Aggregated):

Food	Place	Company	Mean Joy Index
Pizza	Home	Alone	0.65
Pizza	Home	Family	0.83
Pizza	Home	Romantic	1.83
Pizza	Restaurant	Alone	0.81
Pizza	Restaurant	Family	1.51
Pizza	Restaurant	Romantic	3.86
Sushi	Home	Alone	0.33
Sushi	Home	Family	0.77
Sushi	Home	Romantic	1.29
Sushi	Restaurant	Alone	0.81
Sushi	Restaurant	Family	0.94
Sushi	Restaurant	Romantic	2.16
Curry	Home	Alone	0.37
Curry	Home	Family	0.71
Curry	Home	Romantic	1.59
Curry	Restaurant	Alone	0.62
Curry	Restaurant	Family	0.88
Curry	Restaurant	Romantic	2.61

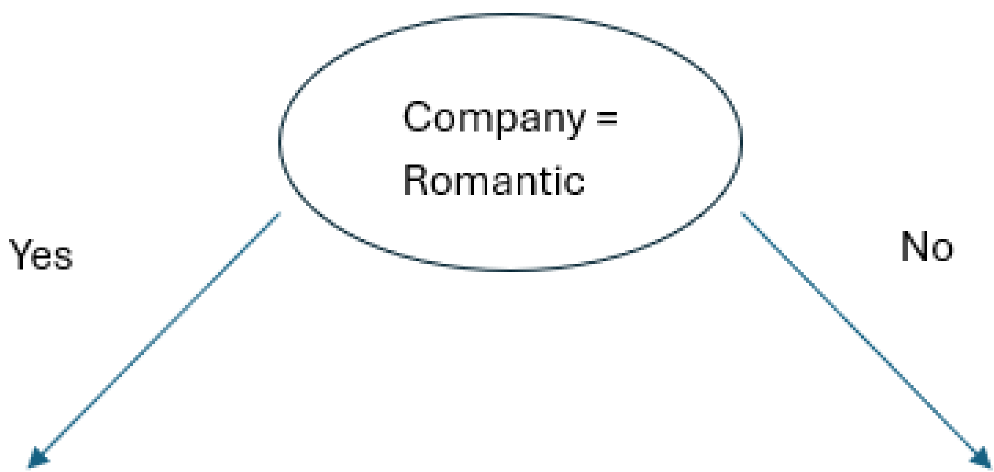
Potential Splits:

Food	Average of Mean Joy Index
Curry	1.13
Pizza	1.58
Sushi	1.05
Grand Total	**1.25**

Place	Average of Mean Joy Index
Home	0.93
Restaurant	1.58
Grand Total	**1.25**

Company	Average of Mean Joy Index
Alone	0.60
Family	0.94
Romantic	2.22
Grand Total	**1.25**

We can see that Company = Romantic is the most powerful split we have. So we construct the first split of the tree:



And this continues for each subgroup.

Gradient Boosting Machine

To greatly oversimplify, GBM models build sequential decision trees on the residuals of prior decision trees. Each tree has it's prediction power reduced by the learning rate before the next tree is built based on the residual.

More precisely, the model computes the gradient (1st derivative) of the loss function with respect to the current predictions. It then fits a tree to those gradients and adds that tree to the model and updates the predictions.

Key points & benefits:

1) Each tree fixes the errors of the prior trees.
2) Weak learners combine into strong learners.
3) Bias decreases while variance stays controlled due to shrinkage (learning rate) and depth limits.
4) No statistical significance to worry about.
5) Works well for reasonably small/thin data.

XGBoost Algorithm

XGBoost improves upon the standard GBM by using the Hessian (2nd derivative) along with the Gradient. This yields a better approximation to how the loss function behaves locally which improves split quality, and convergence speed.

Additionally, GBM's tend to overfit and XGBoost allows additional regularization including L1/L2 penalty and more shrinkage controls. Missing values can be left in the data, which can greatly help deployment pipelines.

Computational improvements include:

1) Histogram based split finding (fits faster)
2) Parallel tree construction
3) Out-of-core computation for big data
4) Can run on GPU if desired