statinfer

Boosting

statinfer.com

Contents

- What is boosting
- Boosting algorithm
- Building models using boosting

Boosting

- Boosting is one more famous ensemble method
- Boosting uses a slightly different techniques to that of bagging.
- Boosting is a well proven theory that works really well on many of the machine learning problems like speech recognition
- If bagging is wisdom of crowds then boosting is wisdom of crowds where each individual is given some weight based on their expertise

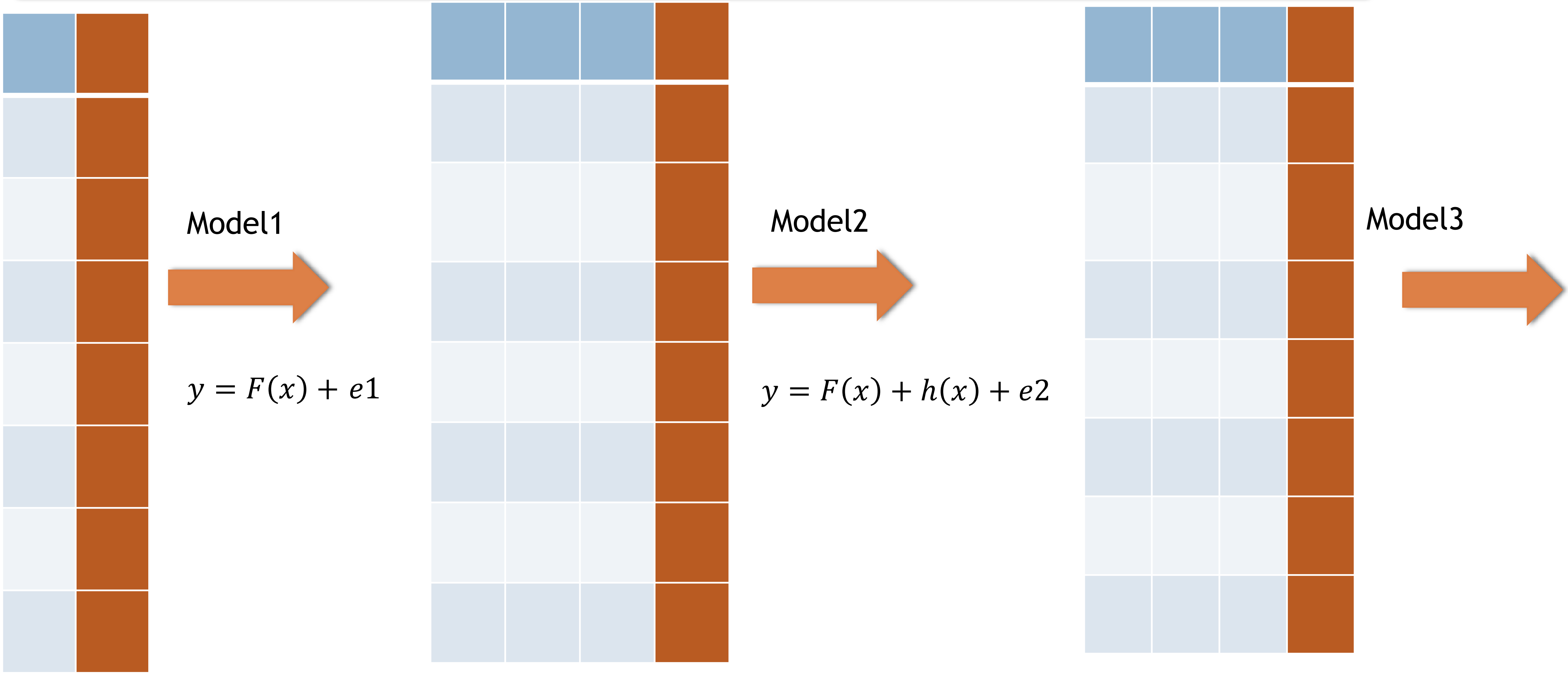
Boosting

- Boosting in general decreases the bias error and builds strong predictive models.
- Boosting is an iterative technique. We adjust the weight of the observation based on the previous classification.
- If an observation was classified incorrectly, it tries to increase the weight of this observation and vice versa.

statinfer

Gradient Boosting

Gradient Boosting Algorithm - illustration



Gradient Boosting Algorithm - illustration

X	Y
28	310
32	360
15	170
35	400
26	286
28	350
33	370

Model1

$$y = F(x) + e1$$

X	Y	$\hat{y}_i$	$e1_i$
28	310	300	10
32	360	400	-40
15	170	200	-30
35	400	350	50
26	286	300	-14
28	350	380	-30
33	370	350	20

Model2

$$y = F(x) + h(x) + e2$$

X	$e1_i$	$\widehat{e1}_i$	$e2_i$
28	10	12	-2
32	-40	-45	5
15	-30	-32	2
35	50	45	5
26	-14	-10	-4
28	-30	-34	4
33	20	18	2

Model3

iterations

- How many iterations shall we run?
- Too many iterations – Overfitting
- Too few iterations – Underfitting
- Iterations is a hyperparameter – need to be optimal

Gradient Boosting Algorithm - Theory

- Step1: Build model1 on original dataset
  - Data points are $(x_1, y_1), (x_2, y_2), (x_3, y_3) \dots (x_N, y_N)$ and the model for this data is $y = F(x)$.
  - Predicted values by the model are $\hat{y}_i = F(x_i)$
- Step2: Calculate the errors (residuals)
  - Calculate errors for the model
  - $e1_i = y_i - \hat{y}_i$; $(y_i = \hat{y}_i + e1_i)$; $(y = F(x) + e1)$
- Step3: Build a model for errors
  - $(x_1, e1_1), (x_2, e1_2), (x_3, e1_3) \dots (x_N, e1_N)$ and build a model on these residuals. Let this model be h(x)
  - $e1 = h(x) + e2.$
- Step4: Update final prediction and errors repeat step 2 and 3
  - $y = F(x) + h(x) + e2$
  - $y = F(x) + \rho * h(x) + e2$ (multiply the predictions from new model with ρ)
  - ρ *is Shrinkage or learning rate*

Shrinkage / Learning rate (ρ)

- $y = F(x) + h(x) + e2$
- $y = F(x) + \rho * h(x) + e2$
- ρ – usually between [0.0001 -0.1]
- Helps us in prolonging the algorithm
- Why prolonging is important ?

statinfer

With ρ=1

$y = F(x) + h(x) + e2$

X	Y
28	310
32	360
15	170
35	400
26	286
28	350
33	370

Model1 →

$y = F(x) + e1$

X	Y	$\hat{y}_i$	$e1_i$
28	310	300	10
32	360	400	-40
15	170	200	-30
35	400	350	50
26	286	300	-14
28	350	380	-30
33	370	350	20

Model2 →

X	$e1_i$	$\widehat{e1}_i$	$e2_i$
28	10	12	-2
32	-40	-45	5
15	-30	-32	2
35	50	45	5
26	-14	-10	-4
28	-30	-34	4
33	20	18	2

Model3 →

With ρ=0.1

$$y = F(x) + 0.1 * h(x) + e2$$

X	Y
28	310
32	360
15	170
35	400
26	286
28	350
33	370

Model1

$$y = F(x) + e1$$

X	Y	$\hat{y}_i$	$e1_i$
28	310	300	10
32	360	400	-40
15	170	200	-30
35	400	350	50
26	286	300	-14
28	350	380	-30
33	370	350	20

Model2

X	$e1_i$	$\widehat{e1}_i$	$e2_i$
28	10	1.2	8.8
32	-40	-4.5	-35.5
15	-30	-3.2	26.8
35	50	4.5	45.5
26	-14	-1.0	-13
28	-30	-3.4	-26.6
33	20	1.8	18.2

Model3

The need of “Learning –rate”

Iterations	Train Accuracy	Test Accuracy
1	10%	9%
2	15%	15%
3	30%	29%
4	45%	45%
5	65%	64%
6	78%	77%
7	84%	84%
8	95%	75%
9	100%	60%

How to stop at 90% train and test accuracy ?

With ρ=0.1

Iterations	Train Accuracy	Test Accuracy
10	10%	9%
20	15%	15%
30	30%	29%
40	45%	45%
50	65%	64%
60	78%	77%
70	84%	84%
80	95%	75%
90	100%	60%

statinfer

XGBoost - XGB

statinfer

XGBoost - XGB

1. Extreme Gradient Boosting – XGBoost – Faster version of GBM
2. Both XGBoost and gbm follows the principle of gradient boosting.
3. There are however, the difference in modelling & performance details.
4. XGBoost used a more regularized model formalization to control over-fitting, which gives it better performance.
5. Improved convergence techniques, vector and matrix type data structures for faster results
6. Unlike GBM, XGBoost package is available in C++, Python, R, Java, Scala, Julia with same parameters for tuning

XGBoost Advantages

1. Developers of XGBoost have made a number of important performance enhancements.
2. XGBoost and GBM have big difference in speed and memory utilization
3. Code modified for better processor cache utilization which makes it faster.
4. Better support for multicore processing which reduces overall training time - **You can use GPU**

Important Parameters and Tips

- Learning rate
  - Also known as eta and shrinkage
  - Keep it between [0.1-0.001]
- Number of trees
  - Also known as number of estimators or iterations or boosting rounds.
  - Keep it optimal [50-200] depending on the data and learning rate
  - If learning rate is low then number of trees should be high
- Tree depth
  - Max_depth. Keep it low- Try [4-5]

Products Sorting in E-commerce Warehouse

LAB: Boosting

- Otto Group Product Classification Challenge https://www.kaggle.com/c/otto-group-product-classification-challenge/overview
- Ecom products classification. Rightly categorizing the items based on their detailed feature specifications. More than 100 specifications have been collected.
- Data: Ecom_Products_Menu/train.csv
- Build a decision tree model and check the training and testing accuracy
- Build a boosted decision tree.
- Is there any improvement from the earlier decision tree

DT vs GBM vs XGB

- DT Accuracy
- GBM Accuracy
- GBM – Execution time
- XGB Model Accuracy
- XGB Execution time

statinfer

Thank You

statinfer.com