Gradient Boosting

Noa Lubin & Lior Sidi

Influenced by Josh Starmer

Y-DATA
SCHOOL OF DATA SCIENCE

Agenda

- Motivation
- Gradient Boosting Trees Regression Example
- Gradient Boosting Trees Algorithm
- Gradient Boosting Trees Classification
- Improvements
- Code
- Summary

Motivation

History from Adaboost to Gradient Boost

- Adaboost invented, the first successful boosting algorithm

  [Freund et al., 1996, Freund and Schapire, 1997]

- Adaboost formulated as gradient descent with a special loss function

  [Breiman et al., 1998, Breiman, 1999]

- Adaboost generalized to Gradient Boosting in order to handle a variety of loss functions

  [Friedman et al., 2000, Friedman, 2001]

Y-DATA
SCHOOL OF DATA SCIENCE

How do we improve Adaboost?

- In Adaboost, “shortcomings” are identified by high-weight data points
- In Gradient Boosting, “shortcomings” are identified by gradients
- We want to build a new model on our residuals (errors)
- It’s easier to think first of a regression problem:

Given $(x_1,y_1), (x_2,y_2), \ldots$ we want to learn $F(x)$ that minimizes square loss

- We build a first weak model, and then try to learn has the mapping

$x_1 \rightarrow y_1\text{-}F_1(x_1)$

$x_2 \rightarrow y_2\text{-}F_1(x_2)$

…

- This process can be repeated!

Gradient Boosting Trees Regression Example

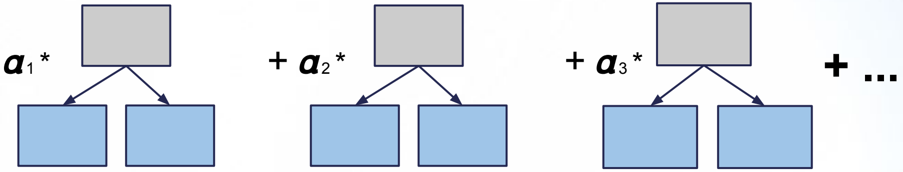
Recalling AdaBoost

Gender	Age	Source	Pay
Male	23	FB	56
Female	49	Search	32
Male	55	Search	45
Female	19	FB	23

Recalling AdaBoost

Gender	Age	Source	Pay
Male	23	FB	56
Female	49	Search	32
Male	55	Search	45
Female	19	FB	23

AdaBoost

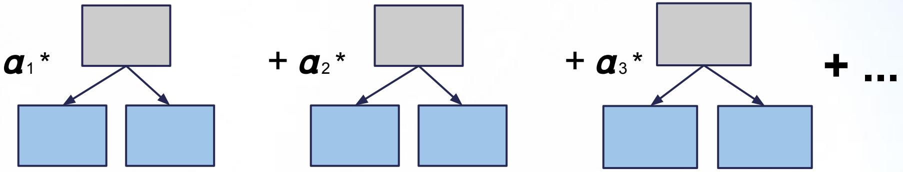


Train **Stump trees** based on the **error** of the previous tree, scale each tree **differently.**

Recalling AdaBoost

Gender	Age	Source	Pay
Male	23	FB	56
Female	49	Search	32
Male	55	Search	45
Female	19	FB	23

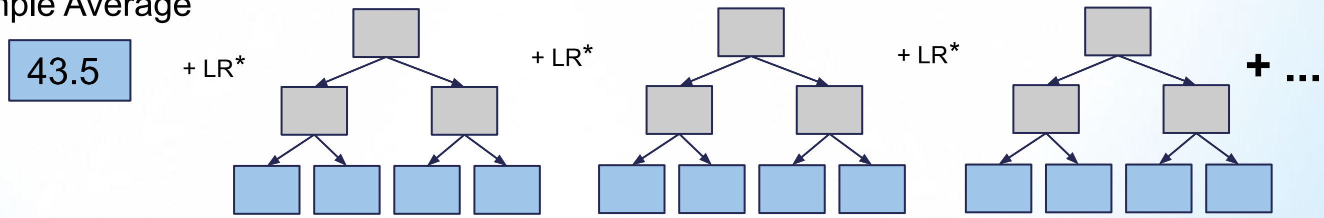
AdaBoost



Train **Stump trees** based on the **error** of the previous tree, scale each tree **differently.**

Gradient Boosting Trees

Simple Average



Train **trees** based on the **Pseudo Residual** of the previous tree, scale each tree **evenly**.

Y-DATA
SCHOOL OF DATA SCIENCE

Start with an Average

Gender	Age	Source	Pay
Male	23	FB	56
Female	49	FB	24
Male	55	Search	45
Male	19	FB	60
Male	43	FB	40
Female	20	FB	62
Female	41	Search	19
Female	36	FB	22

Average Pay

41

Y-DATA
SCHOOL OF DATA SCIENCE

Compute Residual

Gender	Age	Source	Pay	Residual (0)
Male	23	FB	56	15
Female	49	FB	24	
Male	55	Search	45	
Male	19	FB	60	
Male	43	FB	40	
Female	20	FB	62	
Female	41	Search	19	
Female	36	FB	22	

Average Pay

41

Pseudo Residual = (Observed - Predicted)
= (56-41) = 15

Y-DATA
SCHOOL OF DATA SCIENCE

Compute Residual

Gender	Age	Source	Pay	Residual (0)
Male	23	FB	56	15
Female	49	FB	24	-17
Male	55	Search	45	4
Male	19	FB	60	19
Male	43	FB	40	-1
Female	20	FB	62	21
Female	41	Search	19	-22
Female	36	FB	22	-19

Average Pay

41

Pseudo Residual = (Observed - Predicted)
= (56-41) = 15

Y-DATA
SCHOOL OF DATA SCIENCE

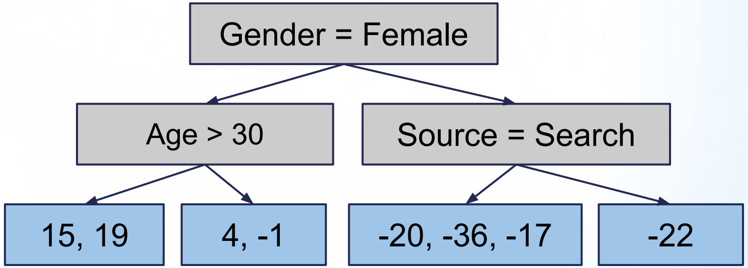
Fit to predict residuals

Gender	Age	Source	Pay	Residual (0)
Male	23	FB	56	15
Female	49	FB	24	-17
Male	55	Search	45	4
Male	19	FB	60	19
Male	43	FB	40	-1
Female	20	FB	62	21
Female	41	Search	19	-22
Female	36	FB	22	-19

Average Pay

41

Pseudo Residual = (Observed - Predicted)
= (41-56) = 15



Y-DATA
SCHOOL OF DATA SCIENCE

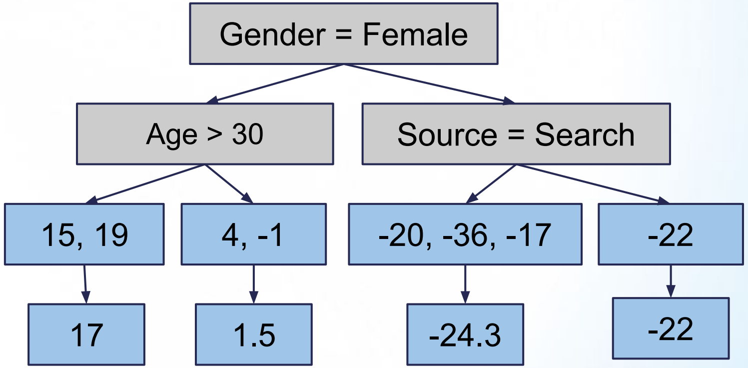
Fit to predict residuals

Gender	Age	Source	Pay	Residual (0)
Male	23	FB	56	15
Female	49	FB	24	-17
Male	55	Search	45	4
Male	19	FB	60	19
Male	43	FB	40	-1
Female	20	FB	62	21
Female	41	Search	19	-22
Female	36	FB	22	-19

Average Pay

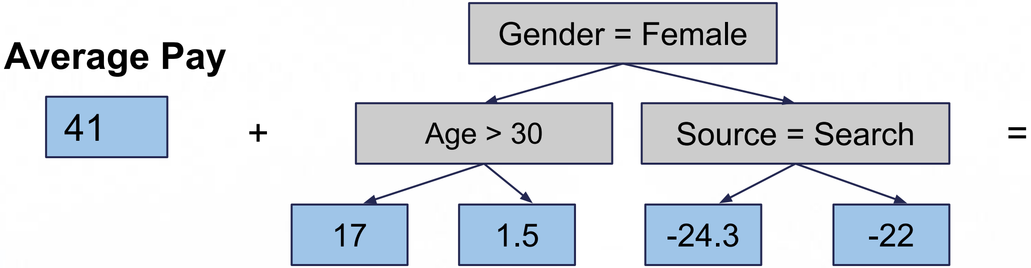
41

Pseudo Residual = (Observed - Predicted)
= (41-56) = 15



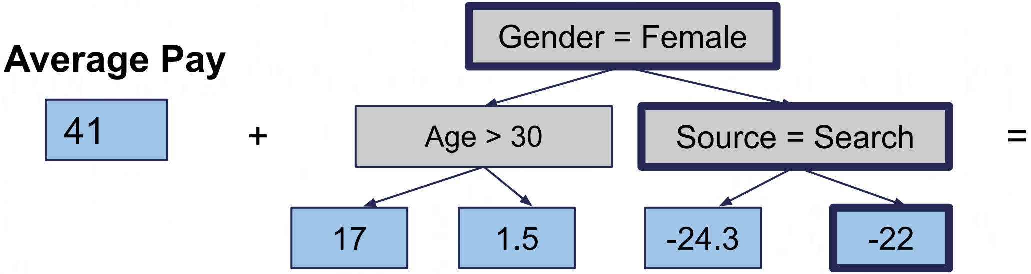
Predict with LR=1

Gender	Age	Source	Pay	Predicted
Female	41	Search	19	?



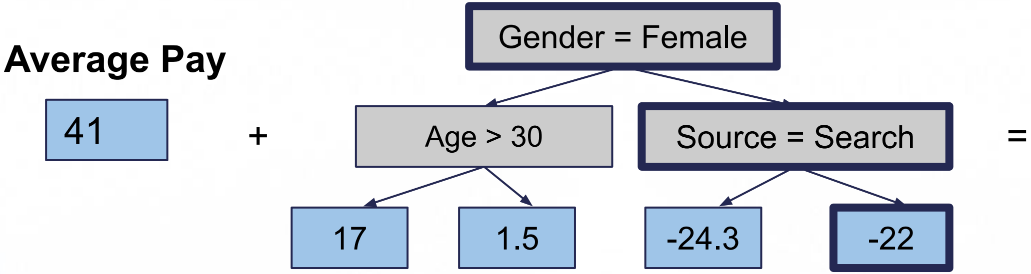
Predict

Gender	Age	Source	Pay	Predicted
Female	41	Search	19	?



Predict

Gender	Age	Source	Pay	Predicted
Female	41	Search	19	41 - 22 = **19**



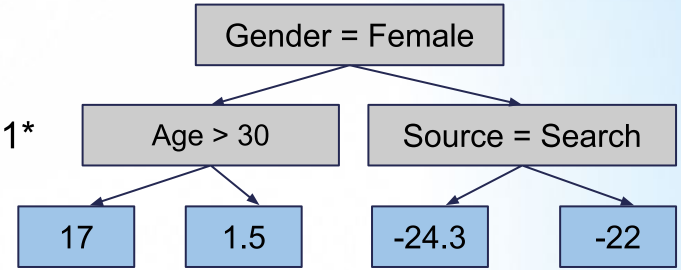
The model overfits the train data

Y-DATA
SCHOOL OF DATA SCIENCE

Compute the residual of the first tree

Gender	Age	Source	Pay	Predicted	Residual (1)
Male	23	FB	56	42.7	13.3
Female	49	FB	24		
Male	55	Search	45		
Male	19	FB	60		
Male	43	FB	40		
Female	20	FB	62		
Female	41	Search	19		
Female	36	FB	22		

41 + 0.1*

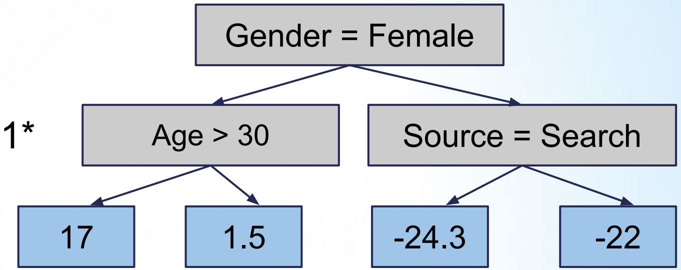


Y-DATA
SCHOOL OF DATA SCIENCE

Compute the residual of the first tree

Gender	Age	Source	Pay	Predicted	Residual (1)
Male	23	FB	56	42.7	13.3
Female	49	FB	24	38.57	-14.57
Male	55	Search	45	41.15	3.85
Male	19	FB	60	42.7	17.3
Male	43	FB	40	41.15	-1.15
Female	20	FB	62	38.57	23.43
Female	41	Search	19	38.8	-19.8
Female	36	FB	22	38.57	-16.57

41 + 0.1*



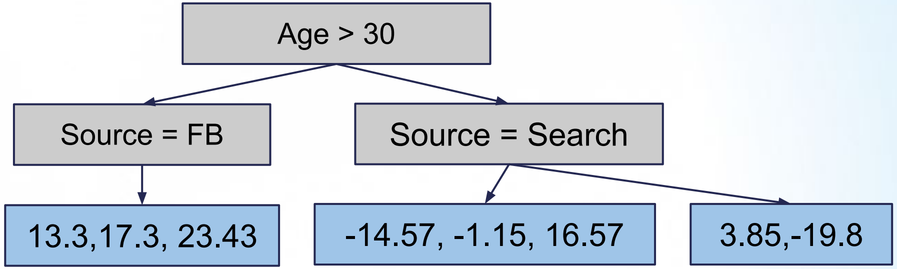
Residuals got improved from last time!

Residual (0)
15
-17
4
19
-1
21
-22
-19

Residual (1)
13.3
-14.57
3.85
17.3
-1.15
23.43
-19.8
-16.57

Fit model 2 to predict residuals

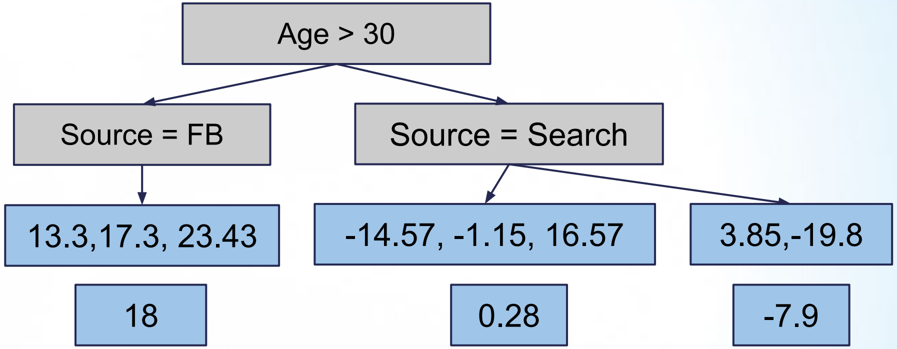
Gender	Age	Source	Pay	Residual
Male	23	FB	56	13.3
Female	49	FB	24	-14.57
Male	55	Search	45	3.85
Male	19	FB	60	17.3
Male	43	FB	40	-1.15
Female	20	FB	62	23.43
Female	41	Search	19	-19.8
Female	36	FB	22	-16.57



Y-DATA
SCHOOL OF DATA SCIENCE

Fit model 2 to predict residuals

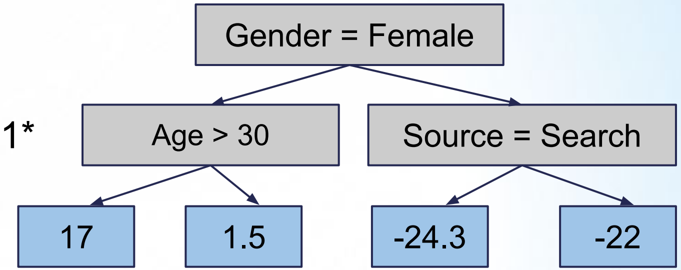
Gender	Age	Source	Pay	Residual
Male	23	FB	56	13.3
Female	49	FB	24	-14.57
Male	55	Search	45	3.85
Male	19	FB	60	17.3
Male	43	FB	40	-1.15
Female	20	FB	62	23.43
Female	41	Search	19	-19.8
Female	36	FB	22	-16.57



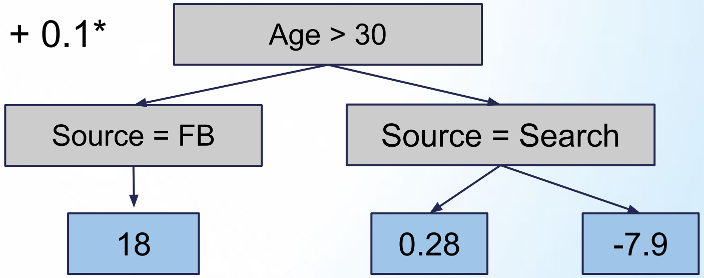
Predict with the new model

Gender	Age	Source	Pay	Predict (2)
Male	23	FB	56	41 +0.1*17+0.1*18
Female	49	FB		
Male	55	Search		
Male	19	FB		
Male	43	FB		
Female	20	FB		
Female	41	Search		
Female	36	FB		

41 + 0.1*



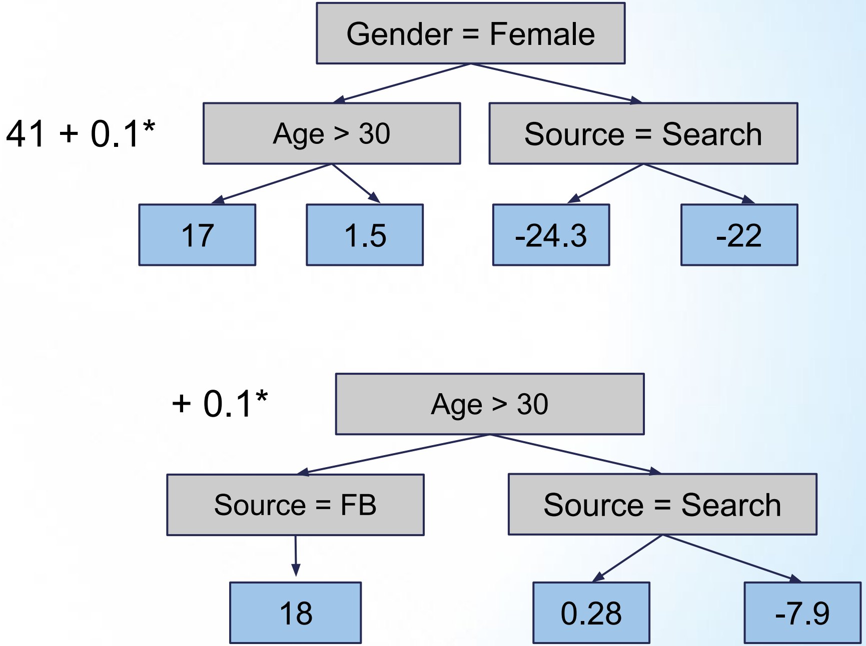
+ 0.1*



Y-DATA
SCHOOL OF DATA SCIENCE

Predict with the new model

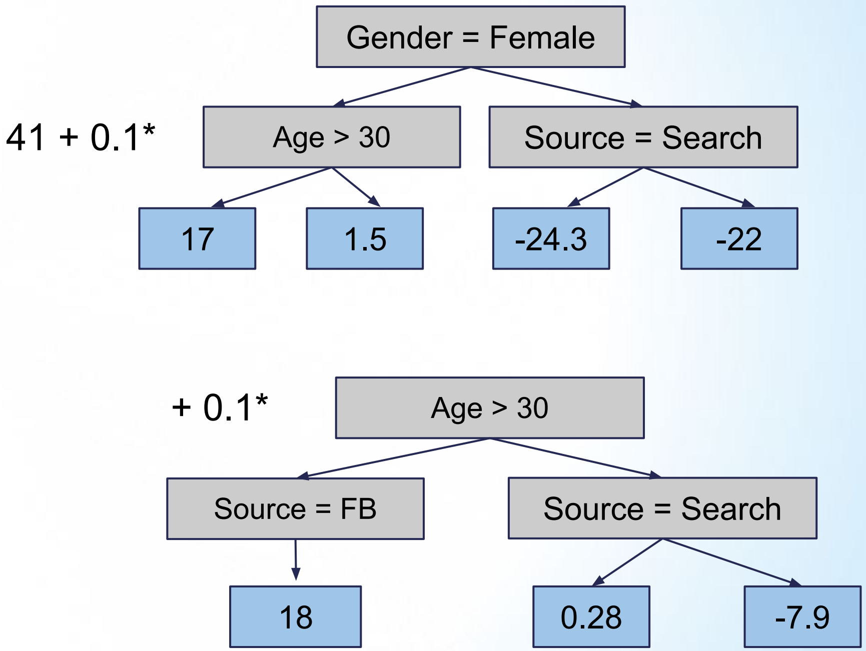
Gender	Age	Source	Pay	Predict (2)
Male	23	FB	56	44.5
Female	49	FB		
Male	55	Search		
Male	19	FB		
Male	43	FB		
Female	20	FB		
Female	41	Search		
Female	36	FB		



Y-DATA
SCHOOL OF DATA SCIENCE

Predict with the new model

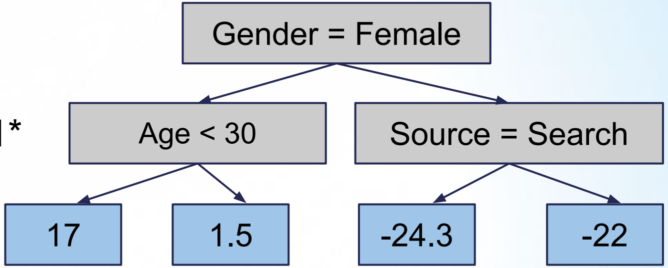
Gender	Age	Source	Pay	Predict (2)
Male	23	FB	56	44.5
Female	49	FB	24	38.1
Male	55	Search	45	40.36
Male	19	FB	60	44.5
Male	43	FB	40	41.178
Female	20	FB	62	40.37
Female	41	Search	19	38.01
Female	36	FB	22	38.598



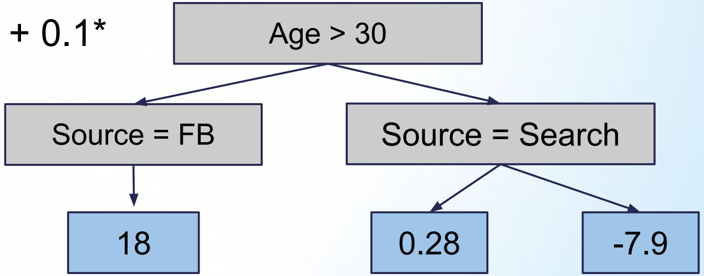
Compute residuals...

Gender	Age	Source	Pay	Predict (2)	Residual (2)
Male	23	FB	56	44.5	11.5
Female	49	FB	24	38.1	-14.1
Male	55	Search	45	40.36	4.64
Male	19	FB	60	44.5	15.5
Male	43	FB	40	41.178	-1.178
Female	20	FB	62	40.37	21.63
Female	41	Search	19	38.01	-19.01
Female	36	FB	22	38.598	-16.598

41 + 0.1*

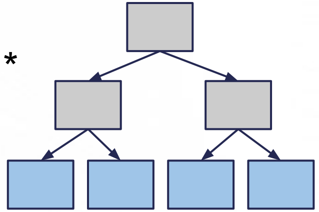


+ 0.1*

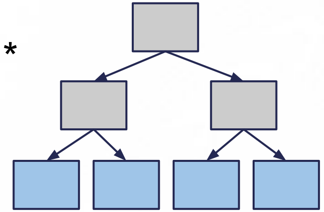


43.5

+ 0.1*

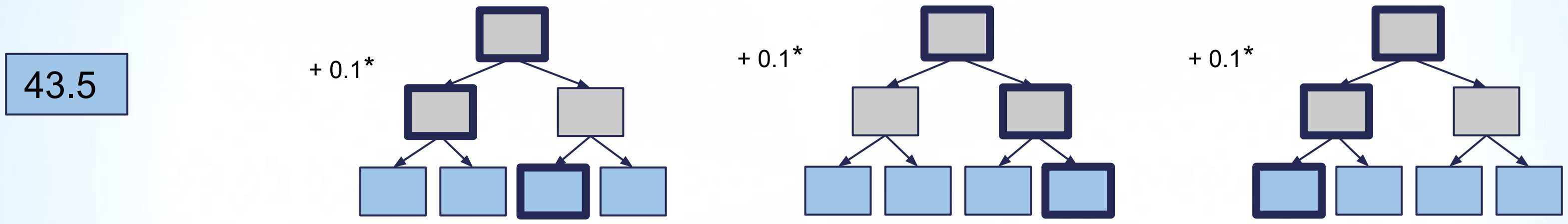


+ 0.1*



Residual (0)	Residual (1)	Residual (2)
15	13.3	11.5
-17	-14.57	-14.1
4	3.85	4.64
19	17.3	15.5
-1	-1.15	-1.178
21	23.43	21.63
-22	-19.8	-19.01
-19	-16.57	-16.598

Using the model for prediction



Gender	Age	Source	Pay
Male	23	FB	?

Gradient Boosting Trees Algorithm

SCHOOL OF DATA

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.
2. For $m = 1$ to M:
   (a) For $i = 1, 2, \ldots, N$ compute

   $$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$

   (b) Fit a regression tree to the targets r_{im} giving terminal regions $R_{jm},\ j = 1, 2, \ldots, J_m$.
   (c) For $j = 1, 2, \ldots, J_m$ compute

   $$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$

   (d) Update $f_m(x) = f_{m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.
3. Output $\hat{f}(x) = f_M(x)$.

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.

2. For $m = 1$ to M:

   (a) For $i = 1, 2, \ldots, N$ compute

   $$r_{im} = -\left[\partial\right.$$

   (b) Fit a regression tree to the targets r_{im} giving terminal regions R_{jm}, $j = 1, 2, \ldots, J_m$.

   (c) For $j = 1, 2, \ldots, J_m$ compute

   $$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$

   (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.

3. Output $\hat{f}(x) = f_M(x)$.

MSE Loss = 1/2(observed - predicted)^2
d(Loss)/d(predicted) = -(observed - predicted)
r = **observed - predicted (Pseudo residual)**

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.

2. For $m = 1$ to M:

   (a) For $i = 1, 2, \ldots, N$ compute

   $r_{im} = -\left[\partial\right.$

Loss = 1/2(observed - predicted)^2
d(Loss)/d(predicted) = **-(observed - predicted)**
r = **observed - predicted (Pseudo residual)**

minal regions

Sum(d(Loss)/d(gamma)) = 0
-56+gamma - 24 + gamma - 45 + gamma = 0
3*gamma = 56 + 24 + 45
Gamma = (56 + 24 + 45) /3 = **average(Observed)**

$\gamma)$.

Gender	Age	Source	Pay
Male	23	FB	56
Female	49	FB	24
Male	55	Search	45

(d $\sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.

3. Ou

Y-D
SCHOOL OF DATA

Algorithm 10.3 *Gradient Tree Boosting Algor*

Loss = 1/2(observed - predicted)^2
d(Loss)/d(predicted) = **-(observed - predicted)**

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$
2. For $m = 1$ to M:
   (a) For $i = 1, 2, \ldots, N$ compute

$$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}$$

Just the Residual
This is where the Gradient appears

   (b) Fit a regression tree to the targets r_{im} giving terminal regions R_{jm}, $j = 1, 2, \ldots, J_m$.
   (c) For $j = 1, 2, \ldots, J_m$ compute

$$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$

   (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.
3. Output $\hat{f}(x) = f_M(x)$.

Y-D
SCHOOL OF DATA

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.
2. For $m = 1$ to M:

   (a) For $i = 1, 2, \ldots, N$ compute

   $$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$

   (b) Fit a regression tree to the targets r_{im} giving terminal regions $R_{jm},\ j = 1, 2, \ldots, J_m$.

   (c) For $j = 1, 2, \ldots, J_m$ compute

   $$\gamma_{jm} = \arg\min_\gamma \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$

   (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.

3. Output $\hat{f}(x) = f_M(x)$.

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.
2. For $m = 1$ to M:
   (a) For $i = 1, 2, \ldots, N$ compute
   $$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$
   (b) Fit a regression tree to the targets r_{im} giving terminal regions $R_{jm},\ j = 1, 2, \ldots, J_m$.
   (c) For $j = 1, 2, \ldots, J_m$ compute
   $$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$
   (d) Update $f_m(x) =$
3. Output $\hat{f}(x) = f_M(x)$.

For leaves with multiple samples → Average

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.
2. For $m = 1$ to M:
   (a) For $i = 1, 2, \ldots, N$ compute
   $$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$
   (b) Fit a regression tree to the targets r_{im} giving terminal regions $R_{jm},\ j = 1, 2, \ldots, J_m$.
   (c) For $j = 1, 2, \ldots, J_m$ compute
   $$\gamma_{jm} = \arg\min_\gamma \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$
   (d) Update $f_m(x) = f_{m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.
3. Output $\hat{f}(x) = f_M(x)$.

Gradient Boosting Trees Classification

Y-DATA
SCHOOL OF DATA SCIENCE

Reminder: Odds

Odds = $$\frac{p(X)}{1-p(X)}$$

$$\frac{p(X)}{1-p(X)} = e^{\beta^T X} = e^{\beta_1 X_1 + \ldots + \beta_p X_p}$$

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta^T X$$

$$p(X) = \frac{\exp(\beta^T X)}{1+\exp(\beta^T X)}$$

Y-DATA
SCHOOL OF DATA SCIENCE

Gender	Age	Source	Pay
Male	23	FB	Yes
Female	49	FB	Yes
Male	55	Search	Yes
Male	19	FB	Yes
Male	43	FB	No
Female	20	FB	Yes
Female	41	Search	No
Female	36	FB	No

Log(odds) to Pay

Odds = #True/#False

Log(5/3) = 0.73 ~0.7

Gender	Age	Source	Pay
Male	23	FB	Yes
Female	49	FB	Yes
Male	55	Search	Yes
Male	19	FB	Yes
Male	43	FB	No
Female	20	FB	Yes
Female	41	Search	No
Female	36	FB	No

Log(odds) to Pay

Log(5/3) = 0.73 ~0.7

Classify by Probability to Pay: Logistic function

Prob = e^log(odds)/(1+e^log(odds)) = 0.7

Y-DATA
SCHOOL OF DATA SCIENCE

Gender	Age	Source	Pay	Residual (0)
Male	23	FB	Yes	0.3
Female	49	FB	Yes	
Male	55	Search	Yes	
Male	19	FB	Yes	
Male	43	FB	No	
Female	20	FB	Yes	
Female	41	Search	No	
Female	36	FB	No	

Log(odds) to Pay

Log(5/3) = 0.73 ~0.7

Classify by Probability to Pay: Logistic function

Prob = e^log(odds)/(1+e^log(odds)) = 0.7

If the threshold is 0.5

Predict all as “Yes”

Residual = (Observed - prob(pay))
= (1-0.7) = 0.3

Gender	Age	Source	Pay	Residual (0)
Male	23	FB	Yes	0.3
Female	49	FB	Yes	0.3
Male	55	Search	Yes	0.3
Male	19	FB	Yes	0.3
Male	43	FB	No	-0.7
Female	20	FB	Yes	0.3
Female	41	Search	No	-0.7
Female	36	FB	No	-0.7

Log(odds) to Pay

Log(5/3) = 0.73 ~0.7

↓

Classify by Probability to Pay: Logistic function

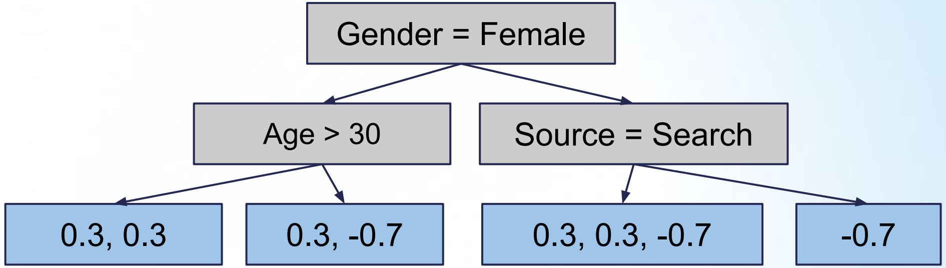
Prob = e^log(odds)/(1+e^log(odds)) = 0.7

↓

Predict all as "Yes"

Residual = (Observed - prob(pay))
= (1-0.7) = 0.3

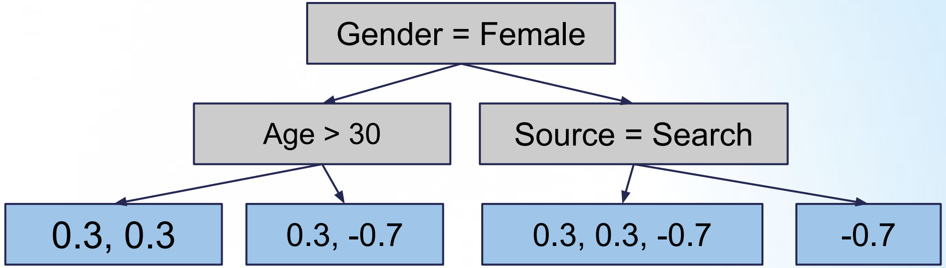
Gender	Age	Source	Pay	Residual (0)
Male	23	FB	Yes	0.3
Female	49	FB	Yes	0.3
Male	55	Search	Yes	0.3
Male	19	FB	Yes	0.3
Male	43	FB	No	-0.7
Female	20	FB	Yes	0.3
Female	41	Search	No	-0.7
Female	36	FB	No	-0.7



We can't simply average!

Y-DATA
SCHOOL OF DATA SCIENCE

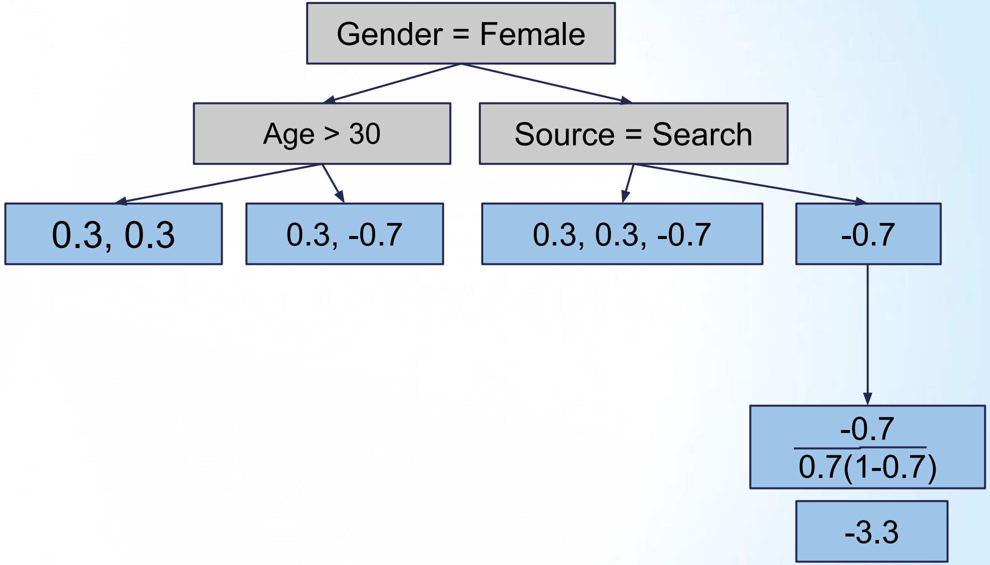
Gender	Age	Source	Pay	Residual (0)
Male	23	FB	Yes	0.3
Female	49	FB	Yes	0.3
Male	55	Search	Yes	0.3
Male	19	FB	Yes	0.3
Male	43	FB	No	-0.7
Female	20	FB	Yes	0.3
Female	41	Search	No	-0.7
Female	36	FB	No	-0.7



$$\frac{\sum \text{Residual}_i}{\sum \left[\text{Previous Probability}_i \times (1 - \text{Previous Probability}_i)\right]}$$

Y-DATA
SCHOOL OF DATA SCIENCE

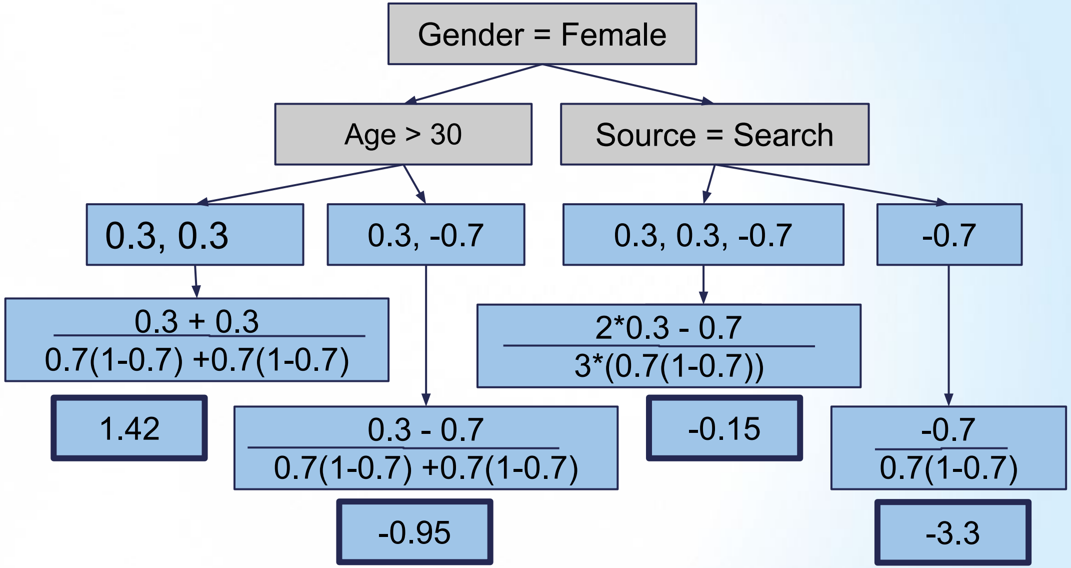
Gender	Age	Source	Pay	Residual (0)
Male	23	FB	Yes	0.3
Female	49	FB	Yes	0.3
Male	55	Search	Yes	0.3
Male	19	FB	Yes	0.3
Male	43	FB	No	-0.7
Female	20	FB	Yes	0.3
Female	41	Search	No	-0.7
Female	36	FB	No	-0.7



$$\frac{\sum \text{Residual}_i}{\sum \left[\text{Previous Probability}_i \times (1 - \text{Previous Probability}_i)\right]}$$

Y-DATA
SCHOOL OF DATA SCIENCE

Gender	Age	Source	Pay	Residual (0)
Male	23	FB	Yes	0.3
Female	49	FB	Yes	0.3
Male	55	Search	Yes	0.3
Male	19	FB	Yes	0.3
Male	43	FB	No	-0.7
Female	20	FB	Yes	0.3
Female	41	Search	No	-0.7
Female	36	FB	No	-0.7



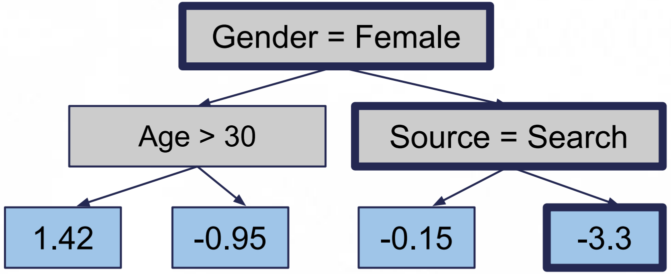
$$\frac{\sum \text{Residual}_i}{\sum \left[\text{Previous Probability}_i \times (1 - \text{Previous Probability}_i)\right]}$$

Compute the new log(odds)

Gender	Age	Source	Pay	Log(odds)	Predicted
Female	41	Search	No	?	?

Log(odds) Pay

0.7 + LR 0.8 *

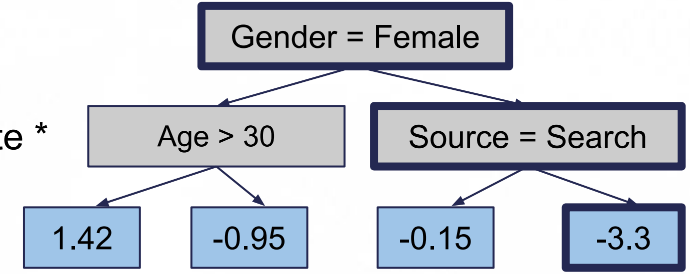


Compute the log(odds)

Gender	Age	Source	Pay	Log(odds)	Predicted
Female	41	Search	No	0.7 +0.8*(-3.3)= **-1.96**	?

Log(odds) Pay

0.7 + Learning Rate * 0.8



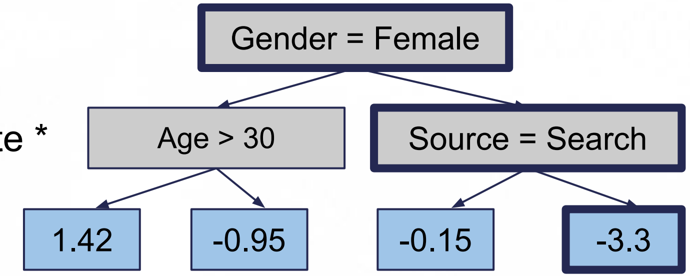
Y-DATA
SCHOOL OF DATA SCIENCE

Compute the log(odds)

Gender	Age	Source	Pay	Log(odds)	Predicted
Female	41	Search	No	0.7 +0.8*(-3.3)= **-1.96**	e^**-1.96**/(1+e^**-1.96**) =0.12

Log(odds) Pay

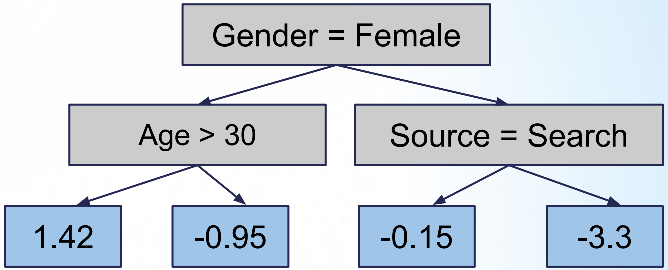
0.7 + Learning Rate * 0.8



Compute the residual of the first tree

Gender	Age	Source	Pay	Predicted	Residual (1)
Male	23	FB	Yes	0.87	0.13
Female	49	FB	Yes	0.64	0.36
Male	55	Search	Yes	0.49	0.51
Male	19	FB	Yes	0.87	0.13
Male	43	FB	No	0.48	-0.48
Female	20	FB	Yes	0.64	0.36
Female	41	Search	No	0.12	-0.12
Female	36	FB	No	0.63	-0.63

0.7 + 0.8*



Y-DATA
SCHOOL OF DATA SCIENCE

Residuals got improved from last time!

Residual (0)
0.3
0.3
0.3
0.3
-0.7
0.3
-0.7
-0.7

Residual (1)
0.13
0.36
0.51
0.13
-0.48
0.36
-0.12
-0.63

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^N L(y_i, \gamma)$.
2. For $m = 1$ to M:
   (a) For $i = 1, 2, \ldots, N$ compute
   $$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$
   (b) Fit a regression tree to the targets r_{im} giving terminal regions $R_{jm},\ j = 1, 2, \ldots, J_m$.
   (c) For $j = 1, 2, \ldots, J_m$ compute
   $$\gamma_{jm} = \arg\min_\gamma \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$
   (d) Update $f_m(x) = f_{m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.
3. Output $\hat{f}(x) = f_M(x)$.

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^N L(y_i, \gamma)$.
2. For $m = 1$ to M:
   (a) For $i = 1, 2, \ldots, N$ compute

   $$r_{im} = -\Big[\partial$$

   (b) Fit a regression tree to the ta
   R_{jm}, $j = 1, 2, \ldots, J_m$.

   (c) For $j = 1, 2, \ldots, J_m$ compute

   $$\gamma_{jm} = \arg\min_\gamma \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma).$$

   (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.
3. Output $\hat{f}(x) = f_M(x)$.

Log likelihood: $\sum_{i=1}^{N} \boldsymbol{y_i} \times \log(\boldsymbol{p}) + (1 - \boldsymbol{y_i}) \times \log(1 - \boldsymbol{p})$

$-\Big[$ **Observed** $\times \log(\boldsymbol{p}) + (1 -$ **Observed**$) \times \log(1 - \boldsymbol{p})\Big]$

5) -**Observed** $\times$ log(odds) + $\log(1 + e^{\log(\text{odds})})$

Y-DATA
SCHOOL OF DATA SCIENCE

Loss Function Derivative

$$\frac{d}{d\,log(odds)}\ \text{-}\mathbf{Observed} \times \log(\text{odds}) + \log(1 + e^{\log(\text{odds})}) =$$

$$= \text{-}\mathbf{Observed} + \frac{e^{\log(\text{odds})}}{1 + e^{\log(\text{odds})}}$$

$$= \text{-}\mathbf{Observed} + \boldsymbol{p}$$

ONY-D
SCHOOL OF DATA

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.

F_0 = log(odds)

2. For $m = 1$ to M:

   (a) For $i = 1, 2, \ldots, N$ compute

   $$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}$$

   (b) Fit a regression tree to the targets r_{im} giving terminal regions $R_{jm},\ j = 1, 2, \ldots, J_m$.

   (c) For $j = 1, 2, \ldots, J_m$ compute

   $$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$

   (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.

3. Output $\hat{f}(x) = f_M(x)$.

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.
2. For $m = 1$ to M:
   (a) For $i = 1, 2, \ldots, N$ compute

$$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$

= **observed - predicted** again!
(Pseudo residual)

   (b) Fit a regression tree to the targets r_{im} giving terminal regions R_{jm}, $j = 1, 2, \ldots, J_m$.
   (c) For $j = 1, 2, \ldots, J_m$ compute

$$\gamma_{jm} = \arg\min_\gamma \sum_{x_i \in R_{jm}} L($$

$$= -\textbf{Observed} + \frac{e^{\log(\text{odds})}}{1 + e^{\log(\text{odds})}}$$

$$= -\textbf{Observed} + p$$

   (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m}$

3. Output $\hat{f}(x) = f_M(x)$.

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.
2. For $m = 1$ to M:
   (a) For $i = 1, 2, \ldots, N$ compute
   $$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$
   (b) Fit a regression tree to the targets r_{im} giving terminal regions $R_{jm},\ j = 1, 2, \ldots, J_m$.
   (c) For $j = 1, 2, \ldots, J_m$ compute
   $$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$
   (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.
3. Output $\hat{f}(x) = f_M(x)$.

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.
2. For $m = 1$ to M:
   (a) For $i = 1, 2, \ldots, N$ compute
   $$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$
   (b) Fit a regression
   $R_{jm},\ j = 1, 2, \ldots$
   (c) For $j = 1, 2, \ldots,$
   $$\gamma_{jm} = \arg\min_\gamma \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$
   (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.
3. Output $\hat{f}(x) = f_M(x)$.

$$\frac{\sum \text{Residual}_i}{\sum \left[\text{Previous Probability}_i \times (1 - \text{Previous Probability}_i)\right]}$$

$$\gamma_{jm} = \underset{\gamma}{\operatorname{argmin}} \sum_{x_i \in R_{ij}} L(y_i, F_{m-1}(x_i) + \gamma)$$

$$\gamma = \frac{\textbf{Residual}}{\boldsymbol{p} \times (1 - \boldsymbol{p})}$$

For simplicity let's look at 1 sample

StatQuest

$$\gamma_{jm} = \underset{\gamma}{\operatorname{argmin}} \sum_{x_i \in R_{ij}} L(y_i, F_{m-1}(x_i) + \gamma)$$

$$\gamma = \frac{\textbf{Residual}}{\boldsymbol{p} \times (1 - \boldsymbol{p})}$$

$$L(y_1, F_{m-1}(x_1) + \gamma) = -y_1 \times [F_{m-1}(x_1) + \gamma] + \log(1 + e^{F_{m-1}(x_1)+\gamma})$$

$$\gamma_{jm} = \underset{\gamma}{\operatorname{argmin}} \sum_{x_i \in R_{ij}} L(y_i, F_{m-1}(x_i) + \gamma)$$

$$\gamma = \frac{\textbf{Residual}}{\boldsymbol{p} \times (1 - \boldsymbol{p})}$$

$$L(y_1, F_{m-1}(x_1) + \gamma) = -y_1 \times [F_{m-1}(x_1) + \gamma] + \log(1 + e^{F_{m-1}(x_1)+\gamma})$$

Approx the Loss function using second order Taylor polynomial

$$L(y_1, F_{m-1}(x_1) + \gamma) \approx L(y_1, F_{m-i}(x_1)) + \frac{d}{dF()}(y_1, F_{m-1}(x_1))\gamma + \frac{1}{2}\frac{d^2}{dF()^2}(y_1, F_{m-1}(x_1))\gamma^2$$

StatQuest

Y-DATA
SCHOOL OF DATA SCIENCE

$$\gamma_{jm} = \underset{\gamma}{\operatorname{argmin}} \sum_{x_i \in R_{ij}} L(y_i, F_{m-1}(x_i) + \gamma)$$

$$\gamma = \frac{\textbf{Residual}}{\boldsymbol{p} \times (1 - \boldsymbol{p})}$$

$$L(y_1, F_{m-1}(x_1) + \gamma) = -y_1 \times [F_{m-1}(x_1) + \gamma] + \log(1 + e^{F_{m-1}(x_1)+\gamma})$$

Approx the Loss function using second order Taylor polynomial

$$L(y_1, F_{m-1}(x_1) + \gamma) \approx L(y_1, F_{m-i}(x_1)) + \frac{d}{dF()}(y_1, F_{m-1}(x_1))\gamma + \frac{1}{2}\frac{d^2}{dF()^2}(y_1, F_{m-1}(x_1))\gamma^2$$

$$\frac{d}{d\gamma}L(y_1, F_{m-1}(x_1) + \gamma) \approx \frac{d}{dF()}(y_1, F_{m-1}(x_1)) + \frac{d^2}{dF()^2}(y_1, F_{m-1}(x_1))\gamma = 0$$

StatQuest

Y-DATA
SCHOOL OF DATA SCIENCE

$$\gamma_{jm} = \underset{\gamma}{\operatorname{argmin}} \sum_{x_i \in R_{ij}} L(y_i, F_{m-1}(x_i) + \gamma)$$

$$\gamma = \frac{\textbf{Residual}}{\boldsymbol{p} \times (1 - \boldsymbol{p})}$$

$$L(y_1, F_{m-1}(x_1) + \gamma) = -y_1 \times [F_{m-1}(x_1) + \gamma] + \log(1 + e^{F_{m-1}(x_1)+\gamma})$$

Approx the Loss function using second order Taylor polynomial

$$L(y_1, F_{m-1}(x_1) + \gamma) \approx L(y_1, F_{m-i}(x_1)) + \frac{d}{dF()}(y_1, F_{m-1}(x_1))\gamma + \frac{1}{2}\frac{d^2}{dF()^2}(y_1, F_{m-1}(x_1))\gamma^2$$

$$\frac{d}{d\gamma}L(y_1, F_{m-1}(x_1) + \gamma) \approx \frac{d}{dF()}(y_1, F_{m-1}(x_1)) + \frac{d^2}{dF()^2}(y_1, F_{m-1}(x_1))\gamma = 0$$

$$\gamma = \frac{-\frac{d}{dF()}(y_1, F_{m-1}(x_1))}{\frac{d^2}{dF()^2}(y_1, F_{m-1}(x_1))}$$

StatQuest

$$\gamma = \frac{-\frac{d}{dF()}(y_1, F_{m-1}(x_1))}{\frac{d^2}{dF()^2}(y_1, F_{m-1}(x_1))} = \frac{\textbf{Observed} - \frac{e^{\log(\text{odds})}}{1 + e^{\log(\text{odds})}}}{\frac{d^2}{dF()^2}(y_1, F_{m-1}(x_1))} = \frac{\textbf{Residual}}{\frac{d^2}{dF()^2}(y_1, F_{m-1}(x_1))} = \frac{\textbf{Residual}}{\boldsymbol{p} \times (1 - \boldsymbol{p})}$$

Remember me?

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^N L(y_i, \gamma)$.
2. For $m = 1$ to M:
   (a) For $i = 1, 2, \ldots, N$ compute
   $$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$
   (b) Fit a regression tree to the targets r_{im} giving terminal regions $R_{jm},\ j = 1, 2, \ldots, J_m$.
   (c) For $j = 1, 2, \ldots, J_m$ compute
   $$\gamma_{jm} = \arg\min_\gamma \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$
   (d) Update $f_m(x) = f_{m-1}(x) +^{\nu} \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.
3. Output $\hat{f}(x) = f_M(x)$.

Y-DATA
SCHOOL OF DATA SCIENCE

Improvements

Y-DATA
SCHOOL OF DATA SCIENCE

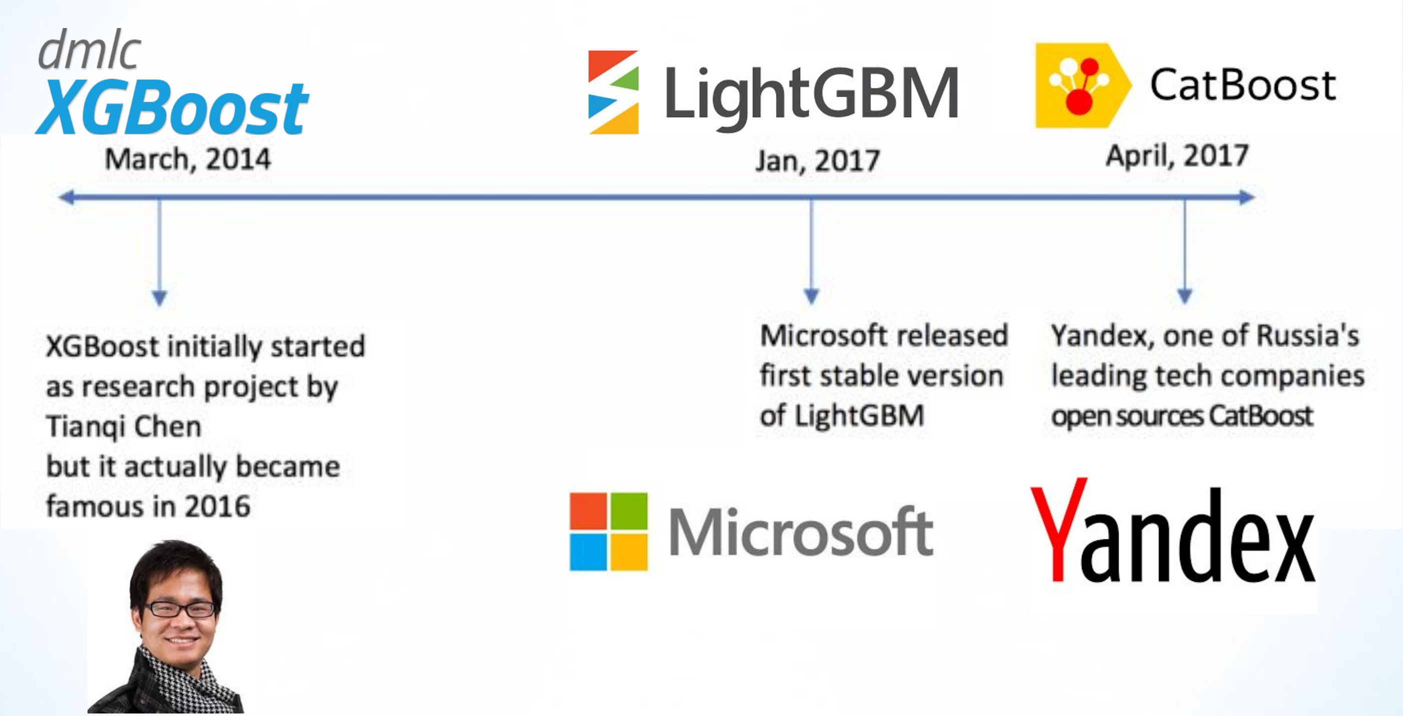
Implementations

dmlc
XGBoost

blog

Implementations

Implementations



Y-DATA SCHOOL OF DATA SCIENCE

XGBoost - E**x**treme **G**radient **Boost**ing

- Award winning algorithm - in 2015, 17 out of 29 Kaggle competitions.

Advancement

- Computing second-order gradients, i.e. second partial derivatives of the loss function (similar to Newton's method), which provides more information about the direction of gradients and how to get to the minimum of our loss function.
- Advanced regularization (L1 & L2), which improves model generalization.
- XGBoost has additional advantages: training is very fast and can be parallelized / distributed across clusters.

paper

dmlc XGBoost

LightGBM

Improvements:

- Gradient-based One-Side Sampling (GOSS)
  - Keeps all the instances with large gradients and performs random sampling on the instances with small gradients - AdaBoost?
- Exclusive Feature Bundling (EFB)
  - bundle mutually exclusive features, an NP problem but computed once before training.

[paper]

LightGBM

CatBoost

Improvements

- Handels Categorical values better
  - Instead of one hot encoding, compute the average of a categorical value for each label (using laplace smoothing)
- Handels Overfitting better
  - You never calculate the residuals on the data you trained on
- Faster
  - Symmetric trees as base predictors. In such trees the same splitting criterion is used across an entire level of the tree. Such trees are balanced and less prone to overfitting.
  - Allow them to use GPU to compute the best splits

[paper]

CatBoost

Function	XGBoost	CatBoost	Light GBM
Important parameters which control overfitting	1. **learning_rate or eta** – optimal values lie between 0.01-0.2 2. **max_depth** 3. **min_child_weight:** similar to min_child leaf; default is 1	1. **Learning_rate** 2. **Depth** - value can be any integer up to 16. Recommended - [1 to 10] 3. No such feature like min_child_weight 4. **l2-leaf-reg**: L2 regularization coefficient. Used for leaf value calculation (any positive integer allowed)	1. **learning_rate** 2. **max_depth**: default is 20. Important to note that tree still grows leaf-wise. Hence it is important to tune **num_leaves** (number of leaves in a tree) which should be smaller than 2^(max_depth). It is a very important parameter for LGBM 3. **min_data_in_leaf**: default=20, alias= min_data, min_child_samples

Function	XGBoost	CatBoost	Light GBM
Important parameters which control overfitting	1. **learning_rate or eta** – optimal values lie between 0.01-0.2 2. **max_depth** 3. **min_child_weight:** similar to min_child leaf; default is 1	1. **Learning_rate** 2. **Depth** - value can be any integer up to 16. Recommended - [1 to 10] 3. No such feature like min_child_weight 4. **l2-leaf-reg**: L2 regularization coefficient. Used for leaf value calculation (any positive integer allowed)	1. **learning_rate** 2. **max_depth**: default is 20. Important to note that tree still grows leaf-wise. Hence it is important to tune **num_leaves** (number of leaves in a tree) which should be smaller than 2^(max_depth). It is a very important parameter for LGBM 3. **min_data_in_leaf**: default=20, alias= min_data, min_child_samples
Parameters for categorical values	Not Available	1. **cat_features**: It denotes the index of categorical features 2. **one_hot_max_size**: Use one-hot encoding for all features with number of different values less than or equal to the given parameter value (max – 255)	1. **categorical_feature:** specify the categorical features we want to use for training our model

Function	XGBoost	CatBoost	Light GBM
Important parameters which control overfitting	1. **learning_rate or eta** – optimal values lie between 0.01-0.2 2. **max_depth** 3. **min_child_weight:** similar to min_child leaf; default is 1	1. **Learning_rate** 2. **Depth** - value can be any integer up to 16. Recommended - [1 to 10] 3. No such feature like min_child_weight 4. **l2-leaf-reg**: L2 regularization coefficient. Used for leaf value calculation (any positive integer allowed)	1. **learning_rate** 2. **max_depth**: default is 20. Important to note that tree still grows leaf-wise. Hence it is important to tune **num_leaves** (number of leaves in a tree) which should be smaller than 2^(max_depth). It is a very important parameter for LGBM 3. **min_data_in_leaf**: default=20, alias= min_data, min_child_samples
Parameters for categorical values	Not Available	1. **cat_features**: It denotes the index of categorical features 2. **one_hot_max_size**: Use one-hot encoding for all features with number of different values less than or equal to the given parameter value (max – 255)	1. **categorical_feature:** specify the categorical features we want to use for training our model
Parameters for controlling speed	1. **colsample_bytree**: subsample ratio of columns 2. **subsample**: subsample ratio of the training instance 3. **n_estimators**: maximum number of decision trees; high value can lead to overfitting	1. **rsm:** Random subspace method. The percentage of features to use at each split selection 2. No such parameter to subset data 3. **iterations**: maximum number of trees that can be built; high value can lead to overfitting	1. **feature_fraction:** fraction of features to be taken for each iteration 2. **bagging_fraction:** data to be used for each iteration and is generally used to speed up the training and avoid overfitting 3. **num_iterations:** number of boosting iterations to be performed; default=100

Y-DATA
SCHOOL OF DATA SCIENCE

Flight Delays

5M samples from 2015:

- MONTH, DAY, DAY_OF_WEEK: data type int
- AIRLINE and FLIGHT_NUMBER: data type int
- ORIGIN_AIRPORT and DESTINATION_AIRPORT: data type string
- DEPARTURE_TIME: data type float
- DISTANCE and AIR_TIME: data type float
- ARRIVAL_DELAY: this will be the target and is transformed into boolean variable indicating delay of more than 10 minutes

	XGBoost	Light BGM	CatBoost
Parameters Used	max_depth: 50 learning_rate: 0.16 min_child_weight: 1 n_estimators: 200	max_depth: 50 learning_rate: 0.1 num_leaves: 900 n_estimators: 300	depth: 10 learning_rate: 0.15 l2_leaf_reg= 9 iterations: 500 one_hot_max_size = 50

	XGBoost	Light BGM		CatBoost	
Parameters Used	max_depth: 50 learning_rate: 0.16 min_child_weight: 1 n_estimators: 200	max_depth: 50 learning_rate: 0.1 num_leaves: 900 n_estimators: 300		depth: 10 learning_rate: 0.15 l2_leaf_reg= 9 iterations: 500 one_hot_max_size = 50	
		Without passing indices of categorical features	Passing indices of categorical features	Without passing indices of categorical features	Passing indices of categorical features
Training AUC Score	0.999	0.992	0.999	0.842	0.887
Test AUC Score	0.789	0.785	0.772	0.752	0.816

	XGBoost	Light BGM		CatBoost	
Parameters Used	max_depth: 50 learning_rate: 0.16 min_child_weight: 1 n_estimators: 200	max_depth: 50 learning_rate: 0.1 num_leaves: 900 n_estimators: 300		depth: 10 learning_rate: 0.15 l2_leaf_reg= 9 iterations: 500 one_hot_max_size = 50	
Training AUC Score	0.999	Without passing indices of categorical features	Passing indices of categorical features	Without passing indices of categorical features	Passing indices of categorical features
		0.992	0.999	0.842	0.887
Test AUC Score	0.789	0.785	0.772	0.752	0.816
Training Time	970 secs	153 secs	326 secs	180 secs	390 secs

	XGBoost	Light BGM		CatBoost	
Parameters Used	max_depth: 50 learning_rate: 0.16 min_child_weight: 1 n_estimators: 200	max_depth: 50 learning_rate: 0.1 num_leaves: 900 n_estimators: 300		depth: 10 learning_rate: 0.15 l2_leaf_reg= 9 iterations: 500 one_hot_max_size = 50	
Training AUC Score	0.999	Without passing indices of categorical features	Passing indices of categorical features	Without passing indices of categorical features	Passing indices of categorical features
		0.992	0.999	0.842	0.887
Test AUC Score	0.789	0.785	0.772	0.752	0.816
Training Time	970 secs	153 secs	326 secs	180 secs	390 secs
Prediction Time	184 secs	40 secs	156 secs	2 secs	14 secs
Parameter Tuning Time (for 81 fits, 200 iteration)	500 minutes	200 minutes		120 minutes	

Code

Y-DATA
SCHOOL OF DATA SCIENCE

Let’s Think About This Together

1. What are the main hyper-parameters?
2. Can it work for multi-class data?
3. How does it handle categorical data?
4. How does it handle missing data?
5. Is it sensitive to outliers?
6. What if some features are correlated?
7. Is it prone to overfitting?
8. Is it Interpretable?
9. Can it be parallelized?
10. Speed of training
11. Speed of prediction

Y-DATA
SCHOOL OF DATA SCIENCE

Let's Think About This Together

1. What are the main hyper-parameters? Iterations (trees), LR, all trees hyper params
2. Can it work for multi-class data? yes
3. How does it handle categorical data? yes
4. How does it handle missing data? yes (same as trees)
5. Is it sensitive to outliers? no
6. What if some features are correlated? Handles well
7. Is it prone to overfitting? yes
8. Is it Interpretable? not off shelf, but there are explainability methods
9. Can it be parallelized? no
10. Speed of training - slow
11. Speed of prediction - medium

Summary

Gradient Boosting Pros & Cons

Pros	Cons
• Works very well “out of the box” • Doesn’t overfit (good results) • Non linear	• Not interpretable • Relatively slow to train (but much faster than NN) • Many hyper-params

Y-DATA
SCHOOL OF DATA SCIENCE

Keep In Touch

NOA_LUBIN

NOALU

NOA_LUBIN