LEC004 Demand Forecasting

VG441 SS2021

Cong Shi
Industrial & Operations Engineering
University of Michigan

Ensemble Learning

"The wisdom of the crowd is the collective opinion of
a group of individuals rather than that of a single expert."

relies on a group of weak predictors

"A group of predictors is called an ensemble. Therefore this Machine Learning technique is known as Ensemble Learning. Voilá!"

"Ensemble methods work best when the predictors are as independent of one another as possible. One way to get diverse classifiers is to train them using very different algorithms. This increases the chance that they will make very different types of errors, improving the ensemble's accuracy."

Ensemble Learning Techniques

- Hard voting classifier (for classification)

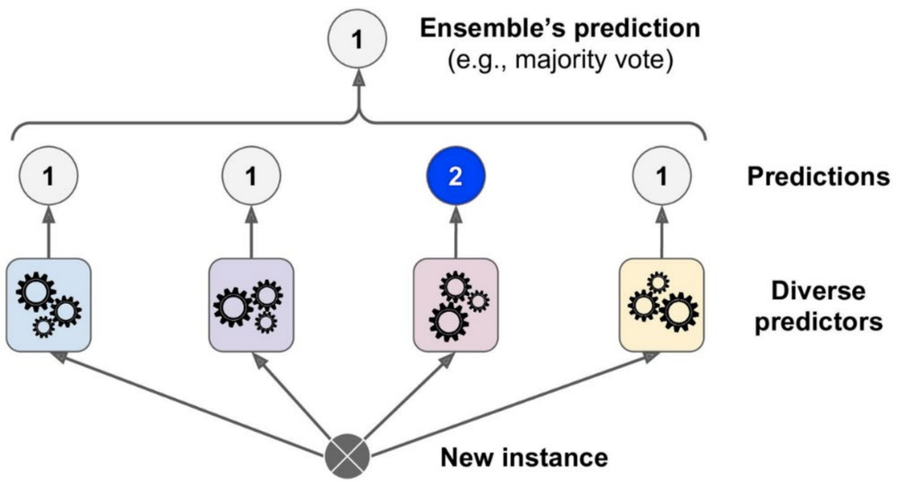
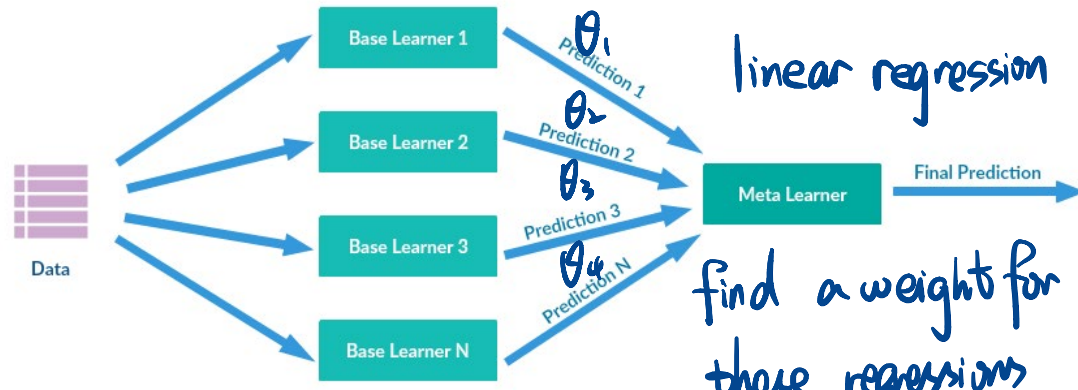


Figure 7-2. Hard voting classifier predictions

- Averaging or weighted averaged (for regression)

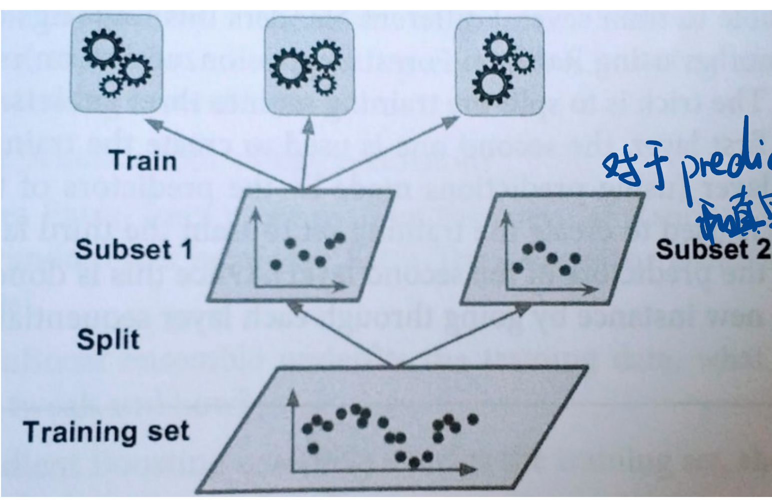
Ensemble Learning Techniques

- Stacking

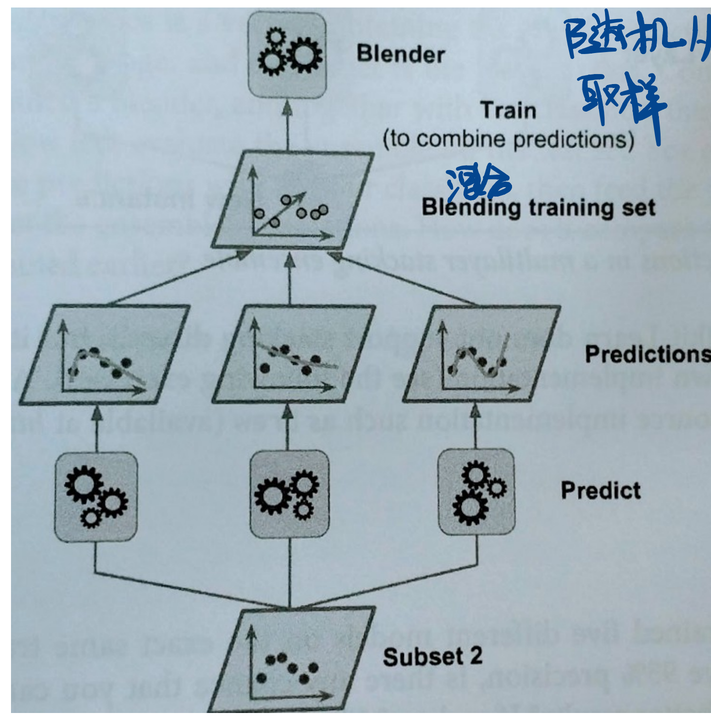


linear regression

find a weight for each of these regressions



对于prediction 和真实



随机从subset 1,2,3 取样

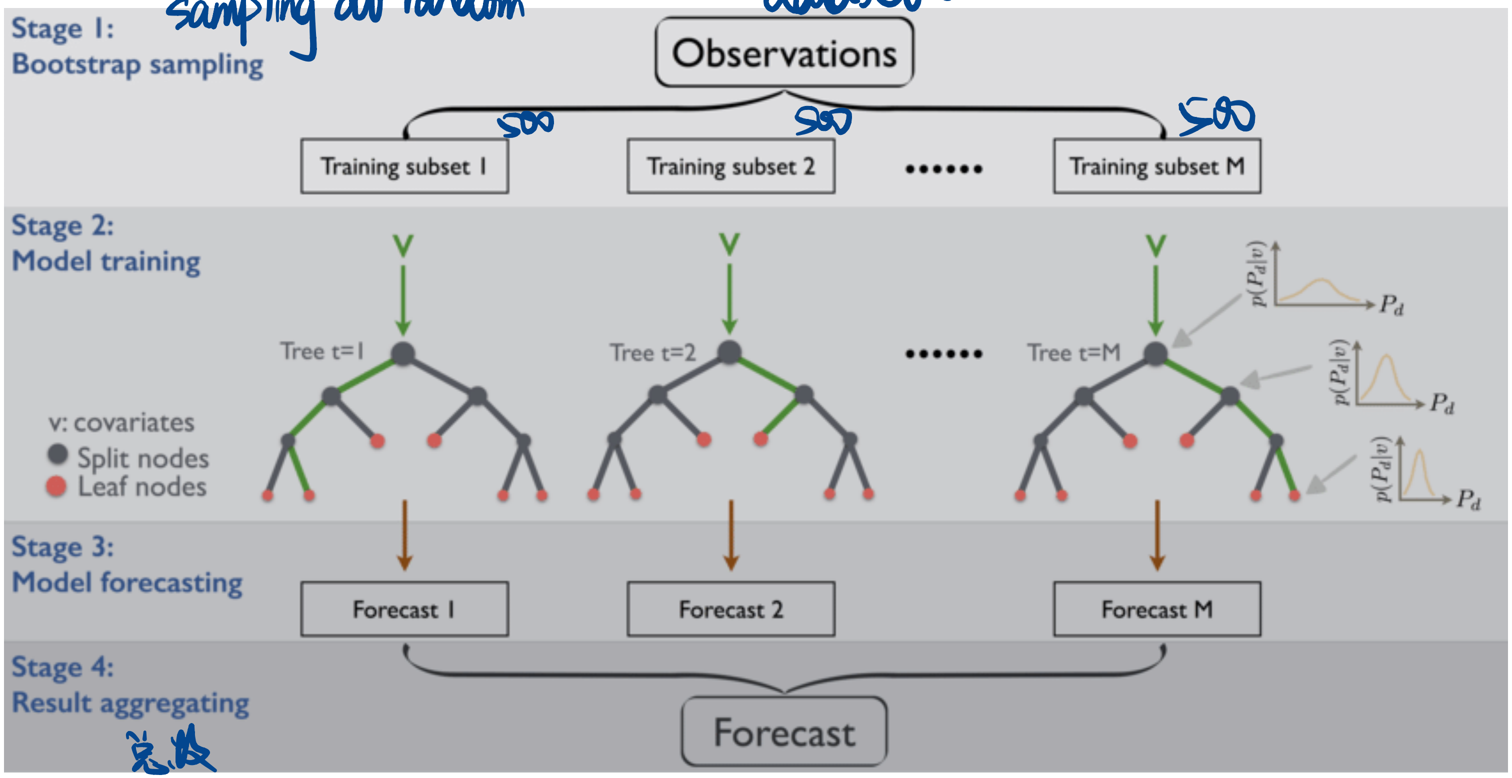
Ensemble Learning Techniques

voting
stacking
bagging
boosting

如何随机生成　　如何整合

- Bagging　bootstrape aggregating

sampling at random　　dataset (1000)



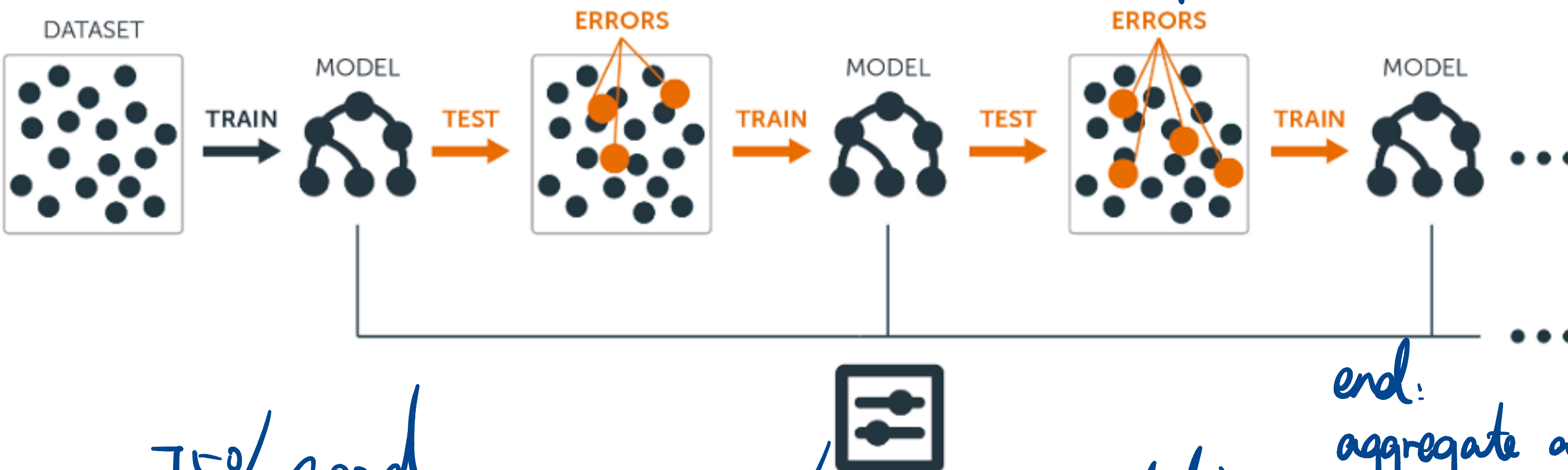
all learners

Random Forest　belong to bagging algorithm

Ensemble Learning Techniques

- Boosting "sequential method"

train more on the previous errors



end: aggregate all the model

75% good

transform

75% errors (repeated)

25% error

AdaBoost, Gradient Boosting, XGBoost popular examples

25% good

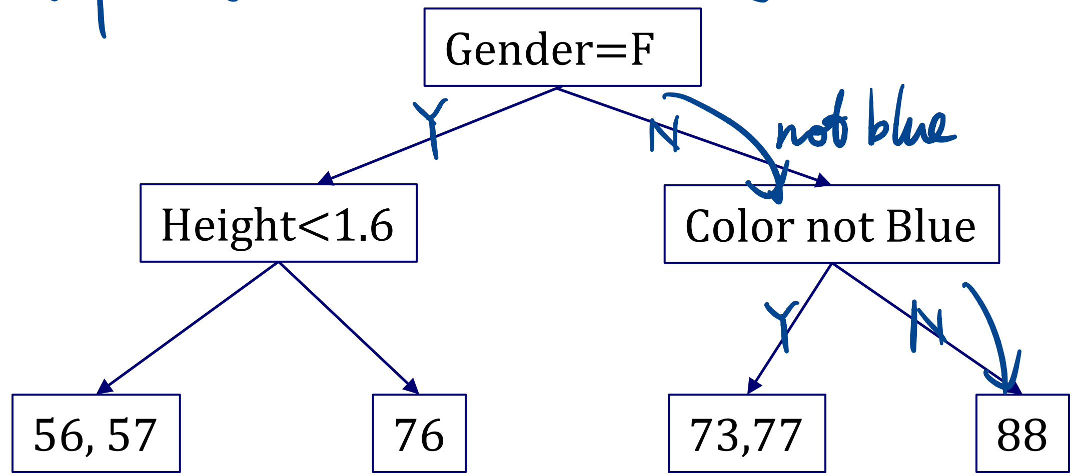
Boosting matter: Decision Tree

x: features Y

use x to predict Y

Height (m)	Favorite Color	Gender	Weight (kg)
1.6	Blue	Male	88
1.6	Green	Female	76
1.5	Blue	Female	56
1.8	Red	Male	73
1.5	Green	Male	77
1.4	Blue	Female	57

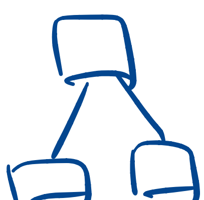
specify level



Question 1: How do we determine the next node (starting from root)?

从小到大

Question 2: Should we split at the current node?



level = 1 "stump"

How to determine and split a node?

Measure of impurity (for regression) is deviance

for classification is gini index

deviance: sum of $(y_i-\bar{y})^2$

Height (m)	Favorite Color	Gender	Weight (kg)
1.6	Blue	Male	88
1.6	Green	Female	76
1.5	Blue	Female	56
1.8	Red	Male	73
1.5	Green	Male	77
1.4	Blue	Female	57

88, 76, 56, 73, 77, 57

Deviance = 774.83

Gender=F

76, 56, 57 | 88, 73, 77

Deviance = 254 + 120.67 = 374.67

lowest deviance

pick this one as node

Height<1.6

56, 77, 57 | 88, 76, 73

Deviance = 280.67 + 126 = 406.67

Not Blue

76, 73, 77 | 88, 56, 57

Deviance = 8.67 + 662 = 670.67

Gradient Boosting

未选择 decision tree

F0 = Initial Model = Taking the mean

Height (m)	Favorite Color	Gender	Weight (kg)
1.6	Blue	Male	88
1.6	Green	Female	76
1.5	Blue	Female	56
1.8	Red	Male	73
1.5	Green	Male	77
1.4	Blue	Female	57

Height (m)	Favorite Color	Gender	Weight (kg)	F0	PR0
1.6	Blue	Male	88	71.2	16.8
1.6	Green	Female	76	71.2	4.8
1.5	Blue	Female	56	71.2	-15.2
1.8	Red	Male	73	71.2	1.8
1.5	Green	Male	77	71.2	5.8
1.4	Blue	Female	57	71.2	-14.2

mean

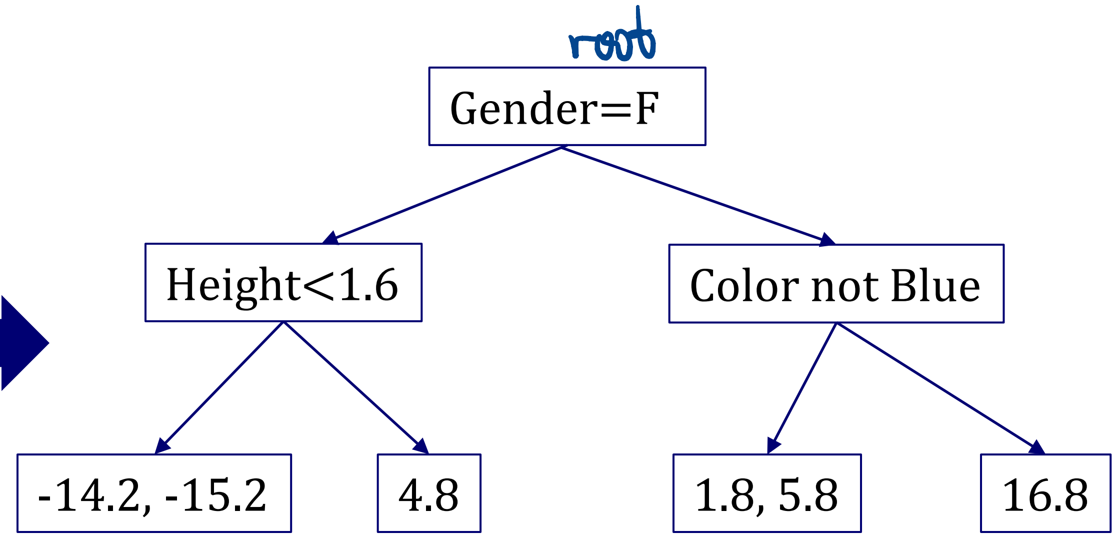
Pseudo Residual (PR) = True Value – Predicted Value

Gradient Boosting

Fit PR0 into a decision tree (up to four leaves)

specify the level of decision tree

Height (m)	Favorite Color	Gender	PR0
1.6	Blue	Male	16.8
1.6	Green	Female	4.8
1.5	Blue	Female	-15.2
1.8	Red	Male	1.8
1.5	Green	Male	5.8
1.4	Blue	Female	-14.2

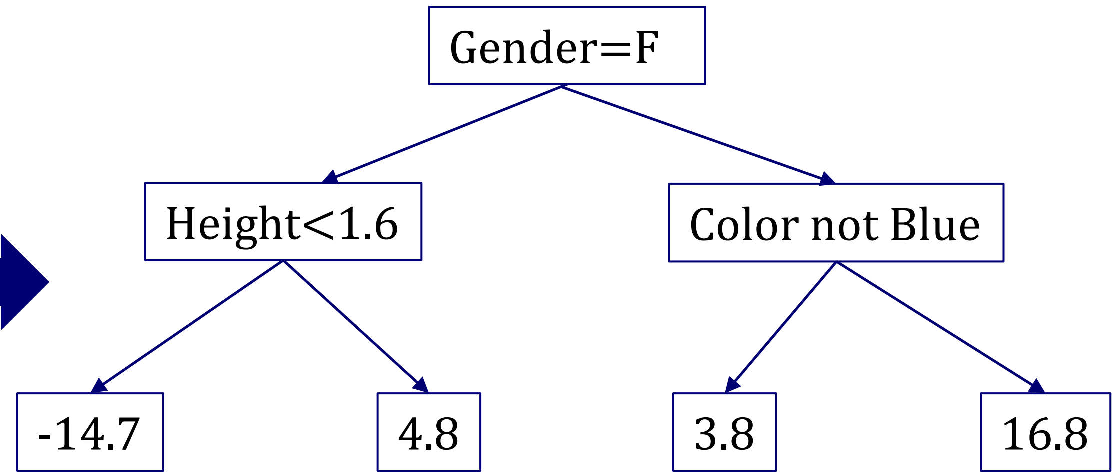


Pseudo Residual (PR) = True Value – Predicted Value

Gradient Boosting

Fit PR0 into a decision tree (up to four leaves)

Height (m)	Favorite Color	Gender	PR0
1.6	Blue	Male	16.8
1.6	Green	Female	4.8
1.5	Blue	Female	-15.2
1.8	Red	Male	1.8
1.5	Green	Male	5.8
1.4	Blue	Female	-14.2



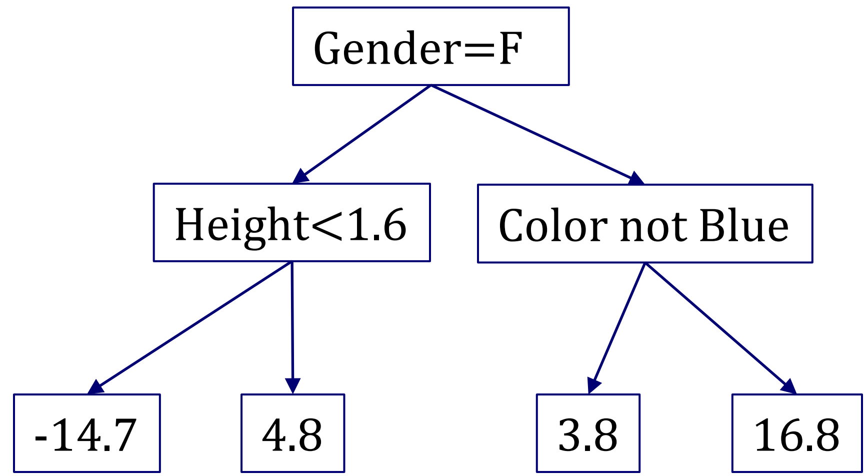
Averaging the residuals on each leaf…

Gradient Boosting

Learning rate = 0.1

$$F1(x) = F0(x) + \gamma_1 \times \text{Output of DT}(x)$$

Height (m)	Favorite Color	Gender	Weight (kg)	F0
1.6	Blue	Male	88	71.2
1.6	Green	Female	76	71.2
1.5	Blue	Female	56	71.2
1.8	Red	Male	73	71.2
1.5	Green	Male	77	71.2
1.4	Blue	Female	57	71.2



F1((1.6, Blue, Male)) = $71.2 + 0.1 \times 16.8 = 72.9$
F1((1.6, Green, Female)) = $71.2 + 0.1 \times 4.8 = 71.7$
F1((1.5, Blue, Female)) = $71.2 + 0.1 \times -14.7 = 69.7$
F1((1.8, Red, Male)) = $71.2 + 0.1 \times 3.8 = 71.6$
F1((1.5, Green, Male)) = $71.2 + 0.1 \times 3.8 = 71.6$
F1((1.4, Blue, Female)) = $71.2 + 0.1 \times -14.7 = 69.7$

Gradient Boosting

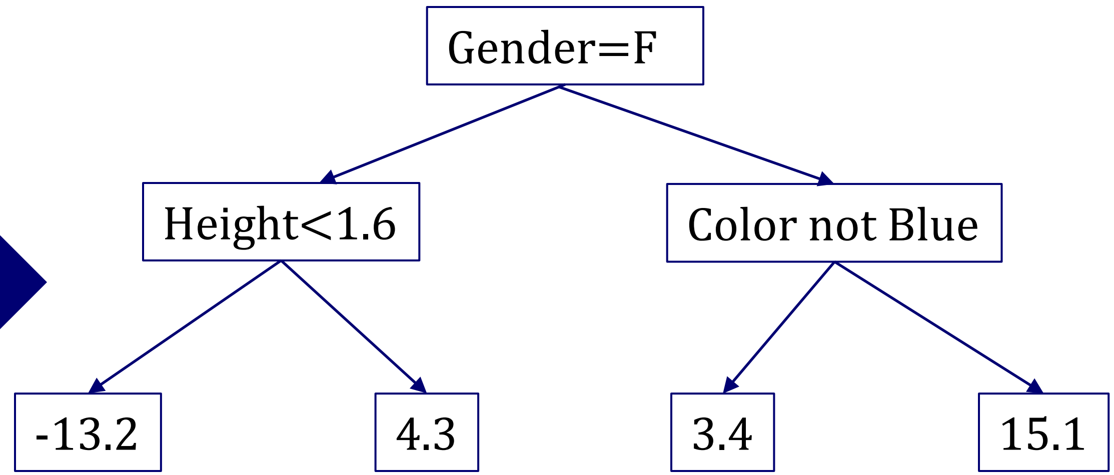
So after building the first DT, we obtain…

Height (m)	Favorite Color	Gender	Weight (kg)	F0	PR0	F1	PR1
1.6	Blue	Male	88	71.2	16.8	72.9	15.1
1.6	Green	Female	76	71.2	4.8	71.7	4.3
1.5	Blue	Female	56	71.2	-15.2	69.7	-13.7
1.8	Red	Male	73	71.2	1.8	71.6	1.4
1.5	Green	Male	77	71.2	5.8	71.6	5.4
1.4	Blue	Female	57	71.2	-14.2	69.7	-12.7

Gradient Boosting

Fit PR1 into a decision tree (up to four leaves)

Height (m)	Favorite Color	Gender	PR1
1.6	Blue	Male	15.1
1.6	Green	Female	4.3
1.5	Blue	Female	-13.7
1.8	Red	Male	1.4
1.5	Green	Male	5.4
1.4	Blue	Female	-12.7



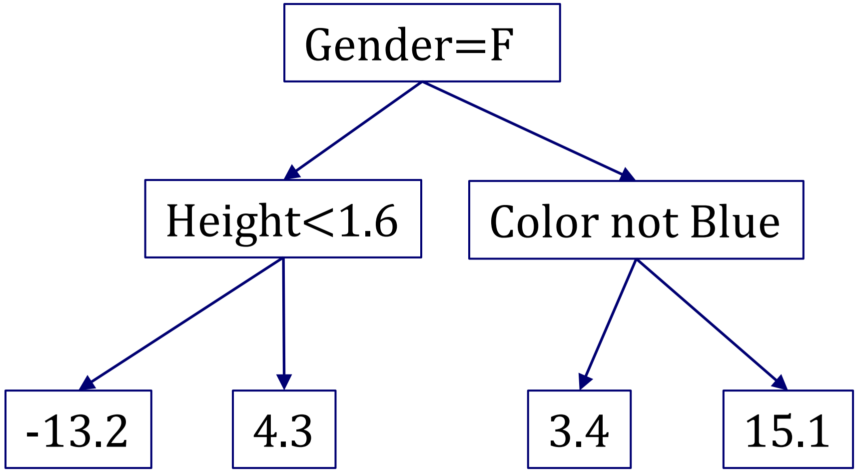
Gradient Boosting

Learning rate = 0.1

之后再说怎么决定 learning rate

$$F2(x) = F1(x) + \gamma_2 \times \text{Output of DT}(x)$$

Height (m)	Favorite Color	Gender	Weight (kg)	F1
1.6	Blue	Male	88	72.9
1.6	Green	Female	76	71.7
1.5	Blue	Female	56	69.7
1.8	Red	Male	73	71.6
1.5	Green	Male	77	71.6
1.4	Blue	Female	57	69.7



$$F2((1.6, \text{Blue}, \text{Male})) = 72.9 + 0.1 \times 15.1 = 74.4$$
$$F2((1.6, \text{Green}, \text{Female})) = 71.7 + 0.1 \times 4.3 = 72.1$$
$$F2((1.5, \text{Blue}, \text{Female})) = 69.7 + 0.1 \times \text{-}13.2 = 68.4$$
$$F2((1.8, \text{Red}, \text{Male})) = 71.6 + 0.1 \times 3.4 = 71.9$$
$$F2((1.5, \text{Green}, \text{Male})) = 71.6 + 0.1 \times 3.4 = 71.9$$
$$F2((1.4, \text{Blue}, \text{Female})) = 69.7 + 0.1 \times \text{-}13.2 = 68.4$$

Gradient Boosting

So after building the second DT, we obtain…

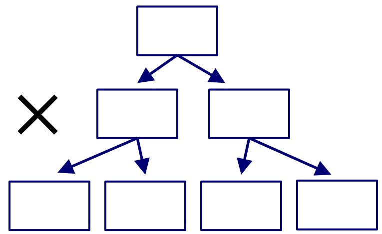
Height (m)	Favorite Color	Gender	Weight (kg)	F0	PR0	F1	PR1	F2	PR2
1.6	Blue	Male	88	71.2	16.8	72.9	15.1	74.4	13.6
1.6	Green	Female	76	71.2	4.8	71.7	4.3	72.1	3.9
1.5	Blue	Female	56	71.2	-15.2	69.7	-13.7	68.4	-12.4
1.8	Red	Male	73	71.2	1.8	71.6	1.4	71.9	1.1
1.5	Green	Male	77	71.2	5.8	71.6	5.4	71.9	5.1
1.4	Blue	Female	57	71.2	-14.2	69.7	-12.7	68.4	-11.4

Notice the PR's are shrinking: Small steps towards the right direction!

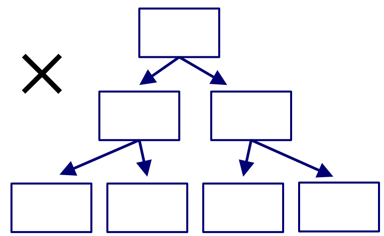
Gradient Boosting

reset
(1) m
(2) stop until sudo-residual doesn't change anymore

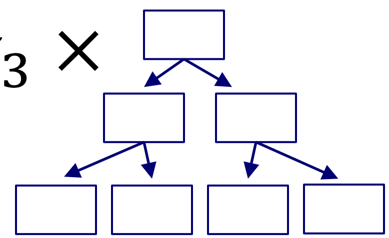
$\mathrm{Fm} = \mathrm{F0} + \gamma_1 \times$



$+\ \gamma_2 \times$



$+\ \gamma_3 \times$



$+\ \ldots$

Fit the new PR into DT

Stop until the pre-specified #DTs or the PR stops improving!

Python Time!

- from sklearn import ensemble

pseudo residual ~ gradient

Some Mathematics...

Input: training set $\{(x_i, y_i)\}_{i=1}^{n}$, a differentiable loss function $L(y, F(x))$, number of iterations M.

Algorithm:

1. Initialize model with a constant value:

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma).$$ γ : predicted of F_0

2. For m = 1 to M:
   1. Compute so-called *pseudo-residuals*:

   $$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \quad \text{for } i = 1, \ldots, n.$$

   2. Fit a base learner (or weak learner, e.g. tree) $h_m(x)$ to pseudo-residuals, i.e. train it using the training set $\{(x_i, r_{im})\}_{i=1}^{n}$.
   3. Compute multiplier γ_m by solving the following one-dimensional optimization problem:

   $$\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{n} L\left(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)\right).$$

   learning rate

   4. Update the model:

   $$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x).$$

3. Output $F_M(x)$.

construction of DT: use $h_m(x)$ to approximate the gradient function

Gradient Boosting

特殊地

- Works exceptionally well in practice
- Won a series of Kaggle competitions
- More robust and explainable