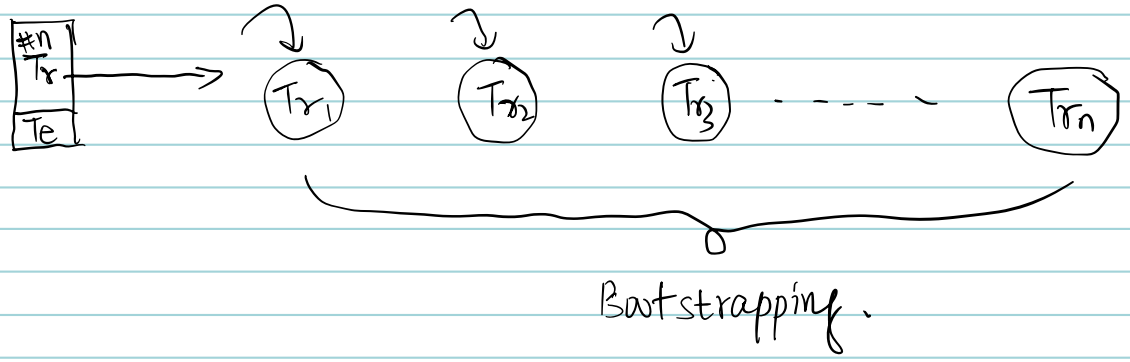
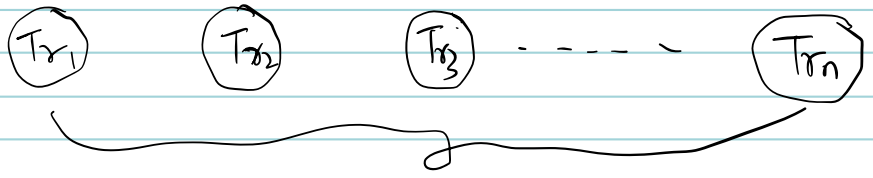
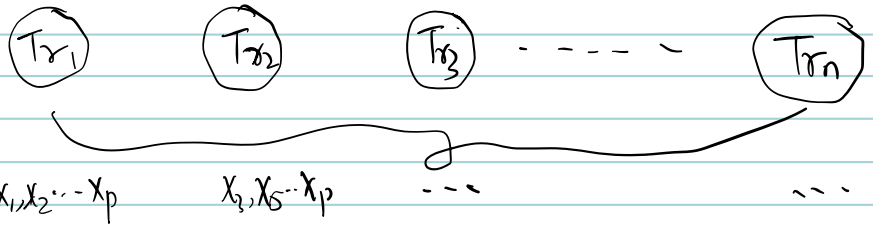
Begging = Bootstrapping + Aggregation.



in average 2/3 of obs in Tr goes to one of Tri

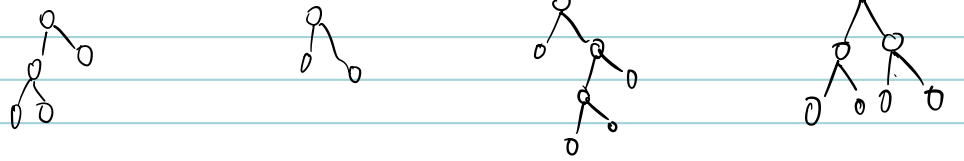


if there is a strong attr in original tree,
it might be used everywhere in Tri.
(then trees might look the same)



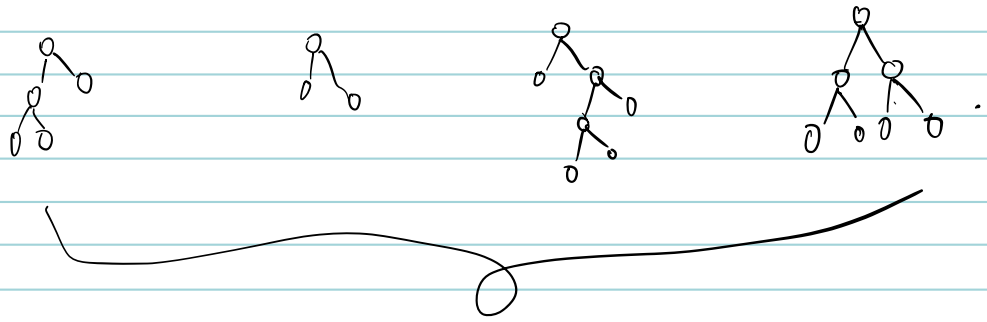
So, we randomly select $\sqrt{p}$ of attrs

⇓ build tree.



⇓ Predict the Te set

roughly $\frac{1}{3}$ of obs in Tr does not go to any of T_{ri}.



any tree did not used observation X_x
then used to make a prediction for X_x in test

Out of bagging error:

each tree used to predict has an error.

$$OBE = \frac{1}{\#\text{tree did prediction}\ (m)} \cdot \sum_{i=1}^{m} Error_i$$

Random forest can run Parallell, cause each tree does not depend on other.

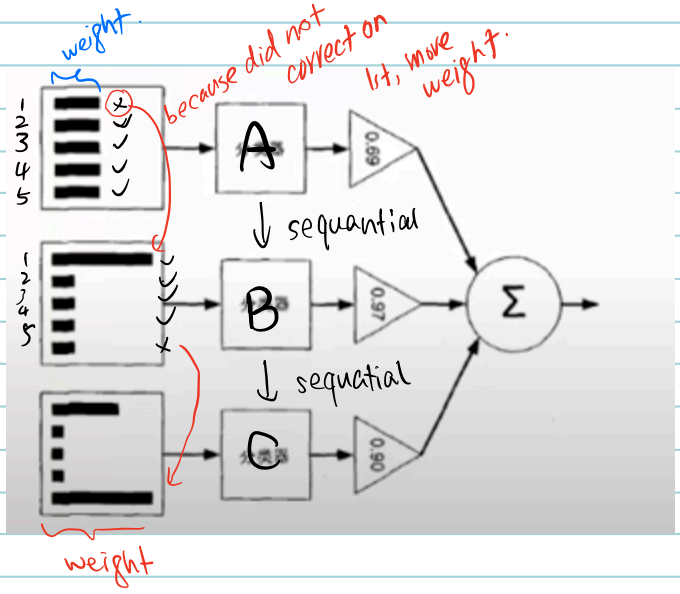
Boosting trees are different, they run sequential.

for Boosting Tree, we mainly do Adaptive Boosting

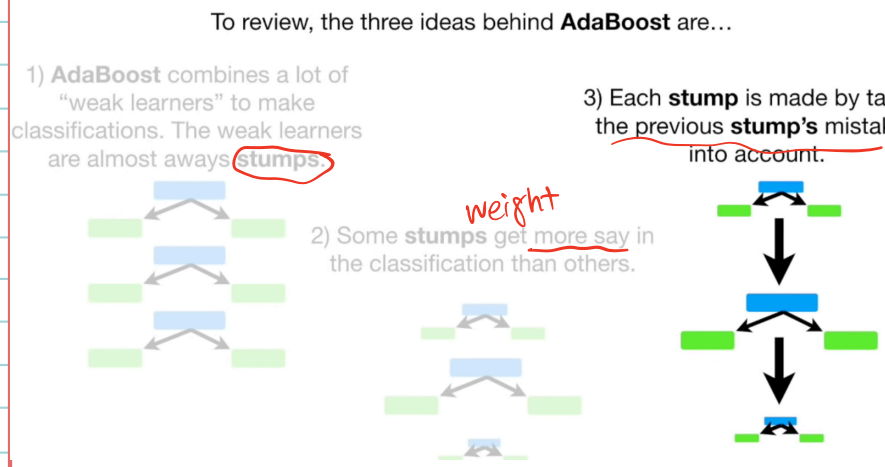
adaboost

Boosting: ① sequential
② each new tree tries to correct the error of the first one.

Ada boost is doing mostly on weight.



https://www.youtube.com/watch?v=LsK-xG1cLYA&t=378s

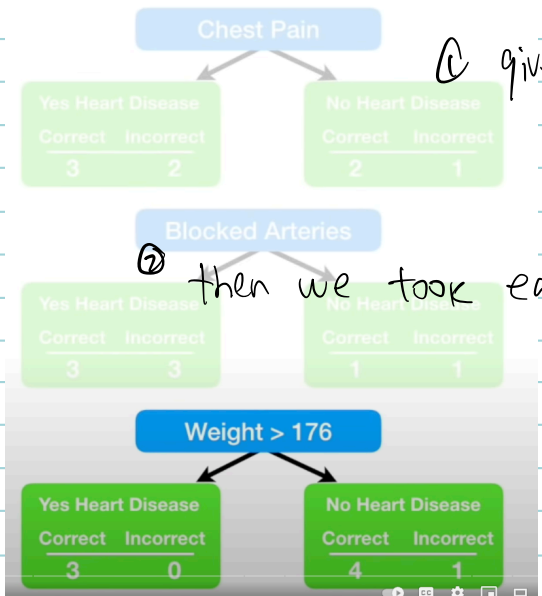


dataset

Chest Pain (X_1)	Blocked Arteries (X_2)	Patient Weight (X_3)	Heart Disease (Y)
Yes	Yes	205	Yes
No	Yes	180	Yes
Yes	No	210	Yes
Yes	Yes	167	Yes
No	Yes	156	No
No	Yes	125	No
Yes	No	168	No
Yes	Yes	172	No

Sample Weight
1/8
1/8
1/8
1/8
1/8
1/8
1/8
1/8

① give sample weight to each obs (sample) $w = \frac{1}{N}$



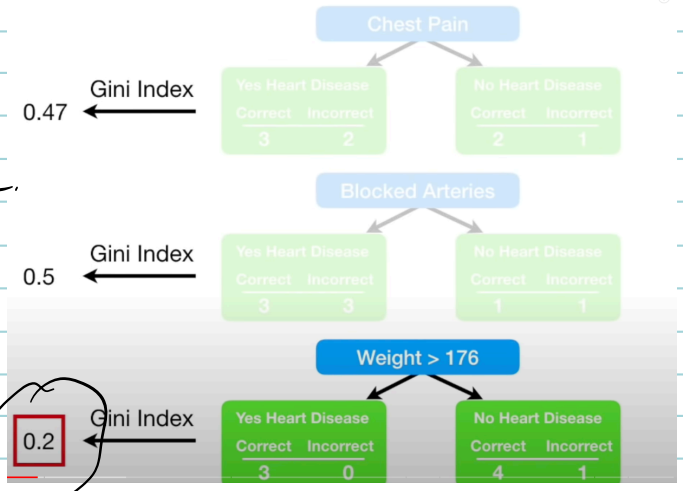
② then we took each of X_1, X_2, X_3 to classify the samples,

③ then we calculate gini index for each of them

Gini Index

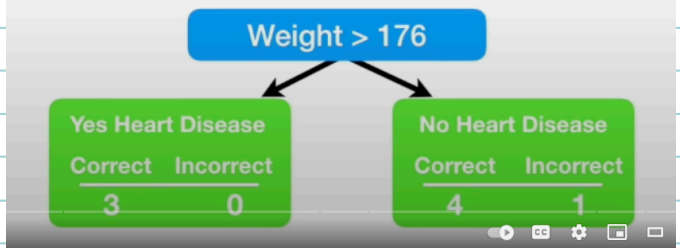
$$I(A) = 1 - \sum_{k=1}^{m} p_k^2$$

p_k: Proportion of obs belong to class k.

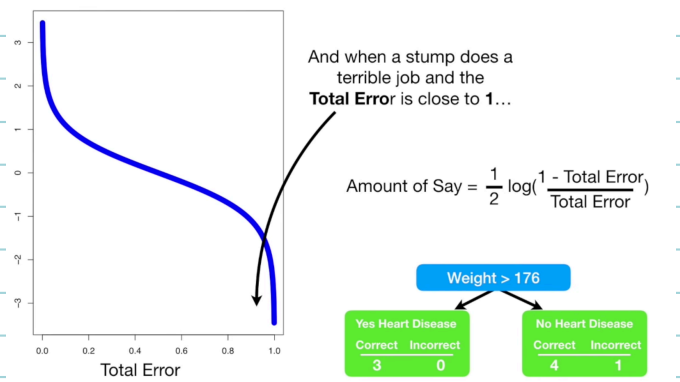


smaller gini index, better impurity,

④ we select this one as it has the smallest



⑤ then we need to redefine the new weights



to increase

$$\text{New Sample Weight} = \text{sample weight} \times e^{\text{amount of say}}$$

decrease

$$\text{New Sample Weight} = \text{sample weight} \times e^{-\text{amount of say}}$$

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight	New Weight	Norm. Weight
Yes	Yes	205	Yes	1/8	0.05	0.07
No	Yes	180	Yes	1/8	0.05	0.07
Yes	No	210	Yes	1/8	0.05	0.07
Yes	Yes	167	Yes	1/8	0.33	0.49
No	Yes	156	No	1/8	0.05	0.07
No	Yes	125	No	1/8	0.05	0.07
Yes	No	168	No	1/8	0.05	0.07
Yes	Yes	172	No	1/8	0.05	0.07

normalize weight so that they add up to 1.

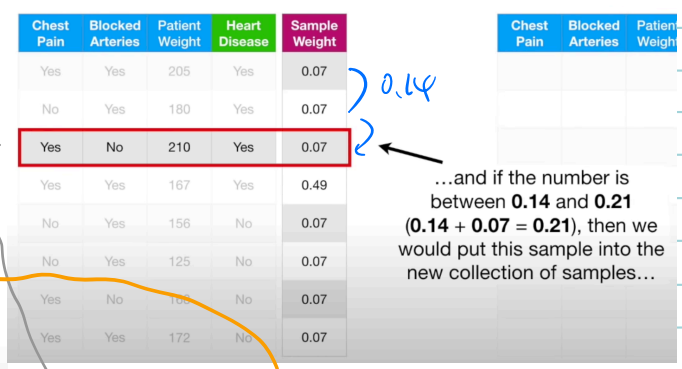
move weight to error

5. sample weights like a distribution

6. then we generate random number between 0 and 1

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease
No	Yes	156	No
Yes	Yes	167	Yes
No	Yes	125	No
Yes	Yes	167	Yes
Yes	Yes	167	Yes
Yes	Yes	172	No
Yes	Yes	205	Yes
Yes	Yes	167	Yes

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Yes	Yes	205	Yes	0.07
No	Yes	180	Yes	0.07
Yes	No	210	Yes	0.07
Yes	Yes	167	Yes	0.49
No	Yes	156	No	0.07
No	Yes	125	No	0.07
Yes	No	168	No	0.07
Yes	Yes	172	No	0.07



according to number lies in which range, we add corresponding obs to the new collection

obs = n

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Yes	Yes	205	Yes	0.07
No	Yes	180	Yes	0.07
Yes	No	210	Yes	0.07
Yes	Yes	167	Yes	0.49
No	Yes	156	No	0.07
No	Yes	125	No	0.07
Yes	No	168	Yes	0.07
Yes	Yes	172	No	0.07

Ultimately, this sample was added to the new collection of samples **4** times, reflecting its larger **Sample Weight**.

# of obs in new collection = n

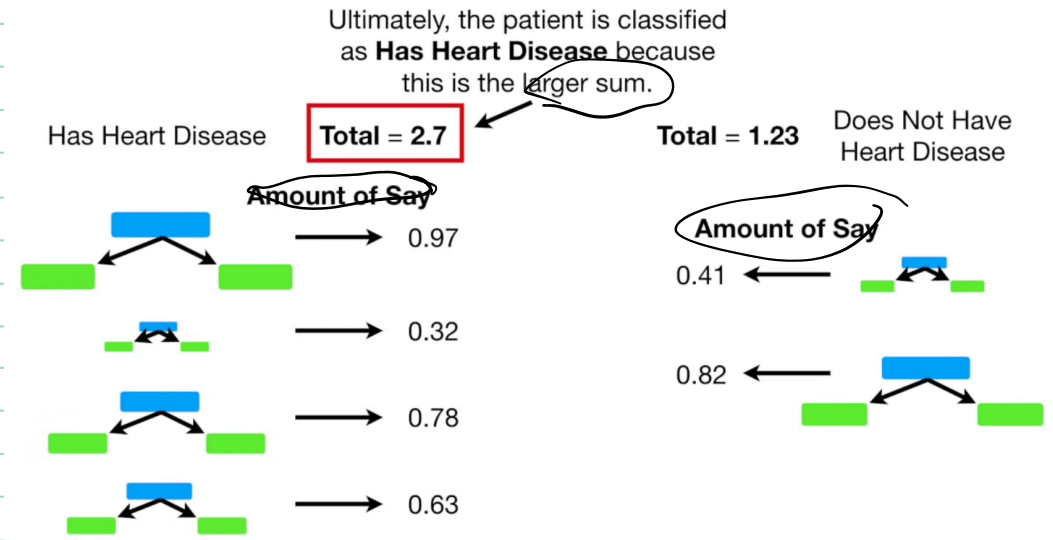
Chest Pain	Blocked Arteries	Patient Weight	Heart Disease
No	Yes	156	No
Yes	Yes	167	Yes
No	Yes	125	No
Yes	Yes	167	Yes
Yes	Yes	167	Yes
Yes	Yes	172	No
Yes	Yes	205	Yes
Yes	Yes	167	Yes

⑦ Now we give new equal weight to them

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
No	Yes	156	No	1/8
Yes	Yes	167	Yes	1/8
No	Yes	125	No	1/8
Yes	Yes	167	Yes	1/8
Yes	Yes	167	Yes	1/8
Yes	Yes	172	No	1/8
Yes	Yes	205	Yes	1/8
Yes	Yes	167	Yes	1/8

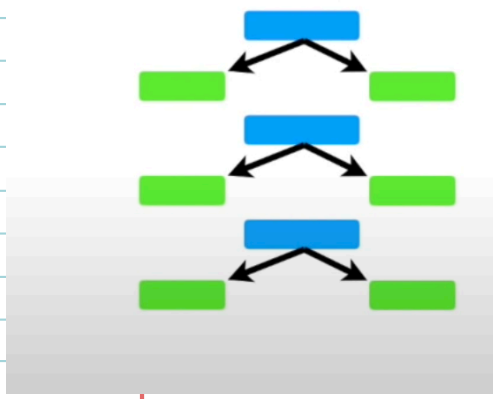
⑧ then we repeat the overall process above

⑨ Prediction

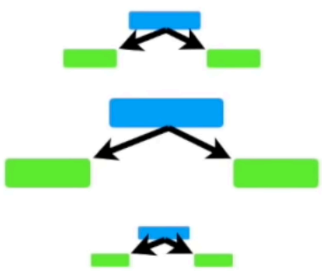


AdaBoost Summary:

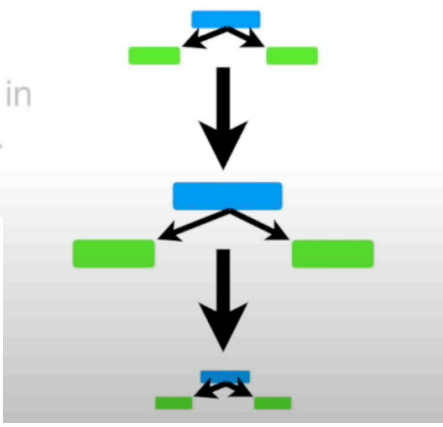
1) **AdaBoost** combines a lot of "weak learners" to make classifications. The weak learners are almost aways **stumps**.



2) Some **stumps** get more say in the classification than others.



3) Each **stump** is made by taking the previous **stump's** mistakes into account.



Adaboost → classification.
adaboost

GBM: gradient Boosting Machine.

gbm 2. regression
regression math behind
classification
classification math
} 4 videos on StatQuest recommended by hocam. on Youtube.

gbm: gradient boosting machine
↓
for regression

ADA Boost 2 DT
Random forest

In RF, each tree has the same amount of say.
On AdaBoost (ADA) ADB trees have different say.