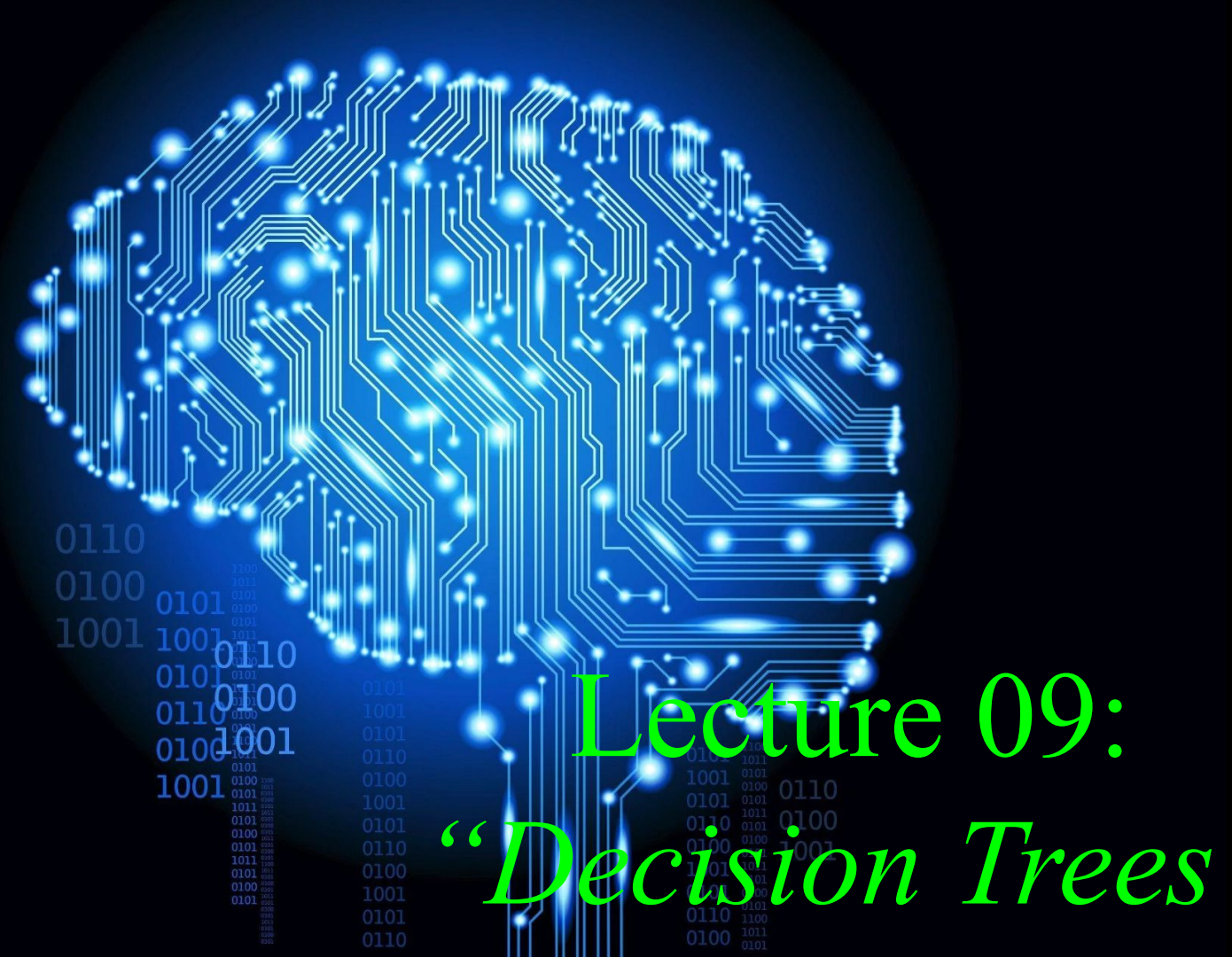
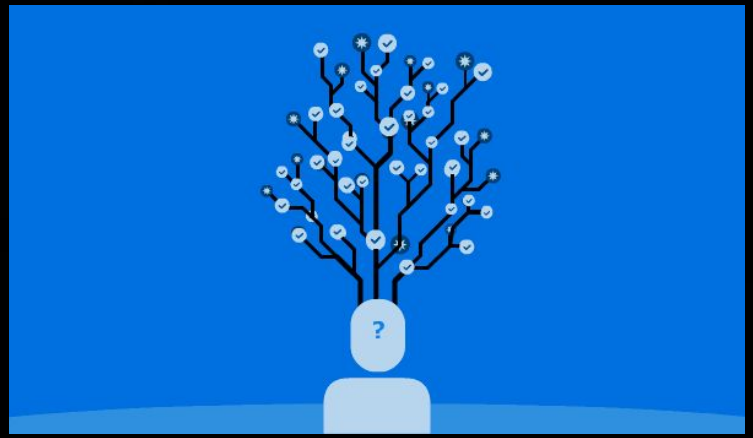
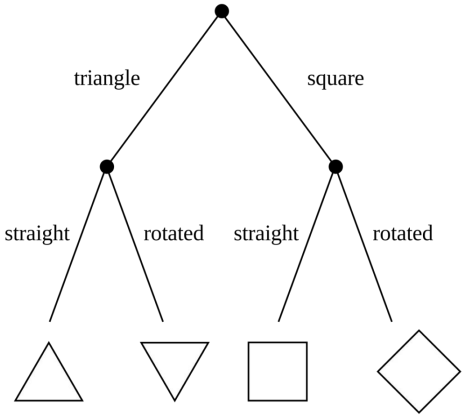


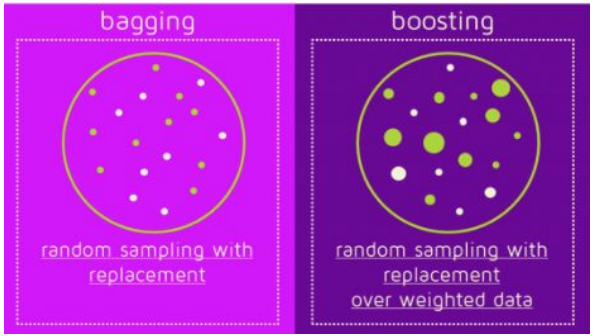
Lecture 09:
"Decision Trees"



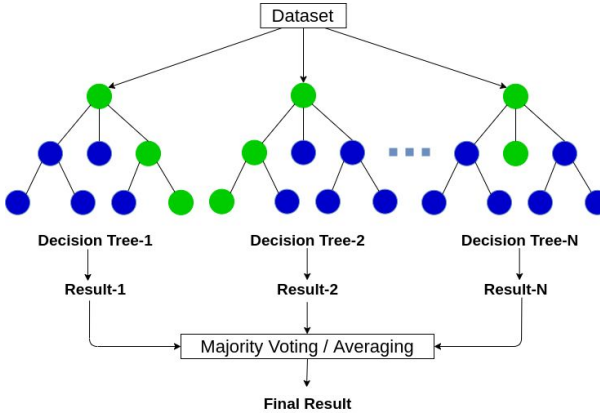
This week...



Decision Trees

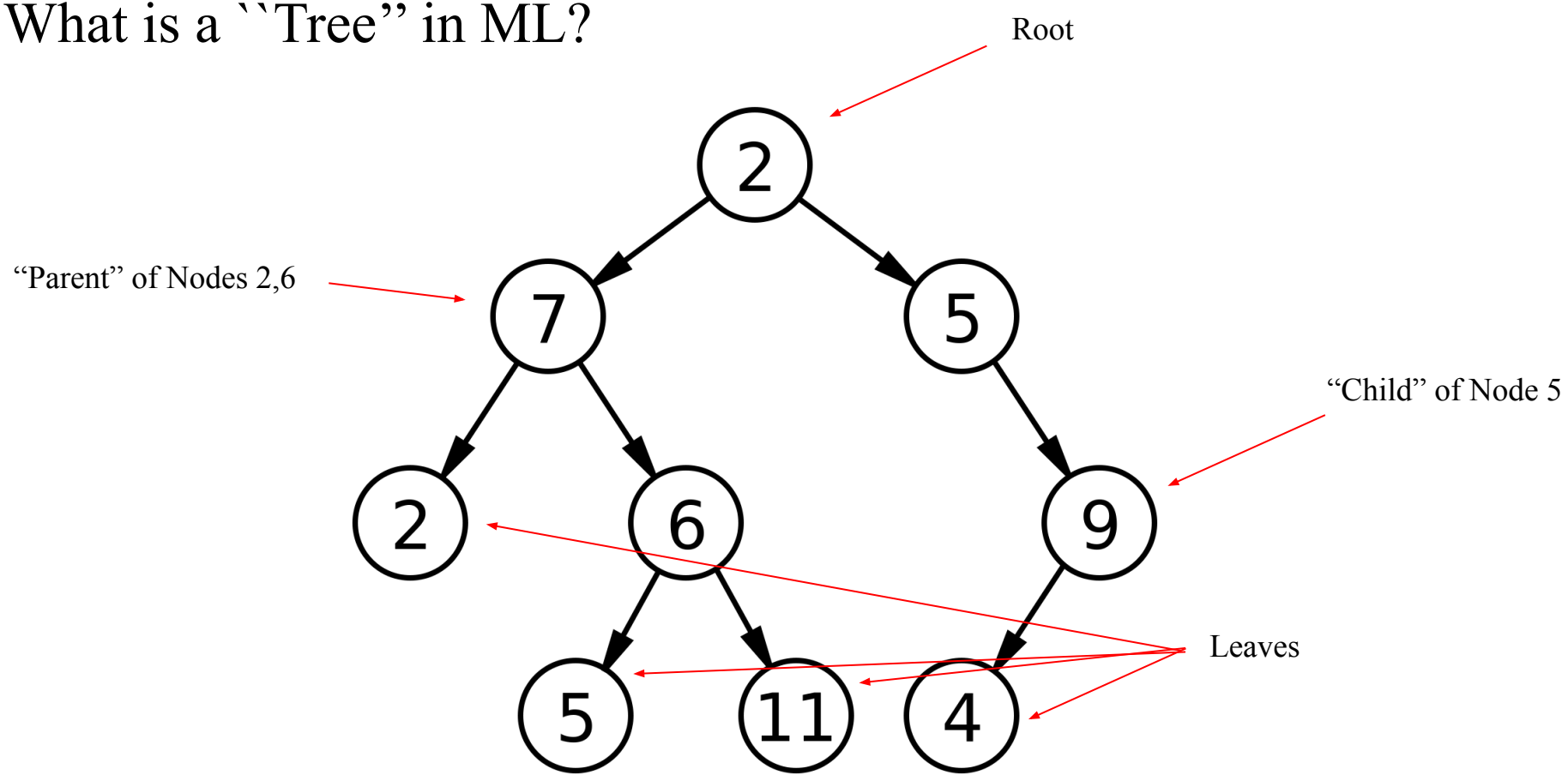


Bagging/Boosting

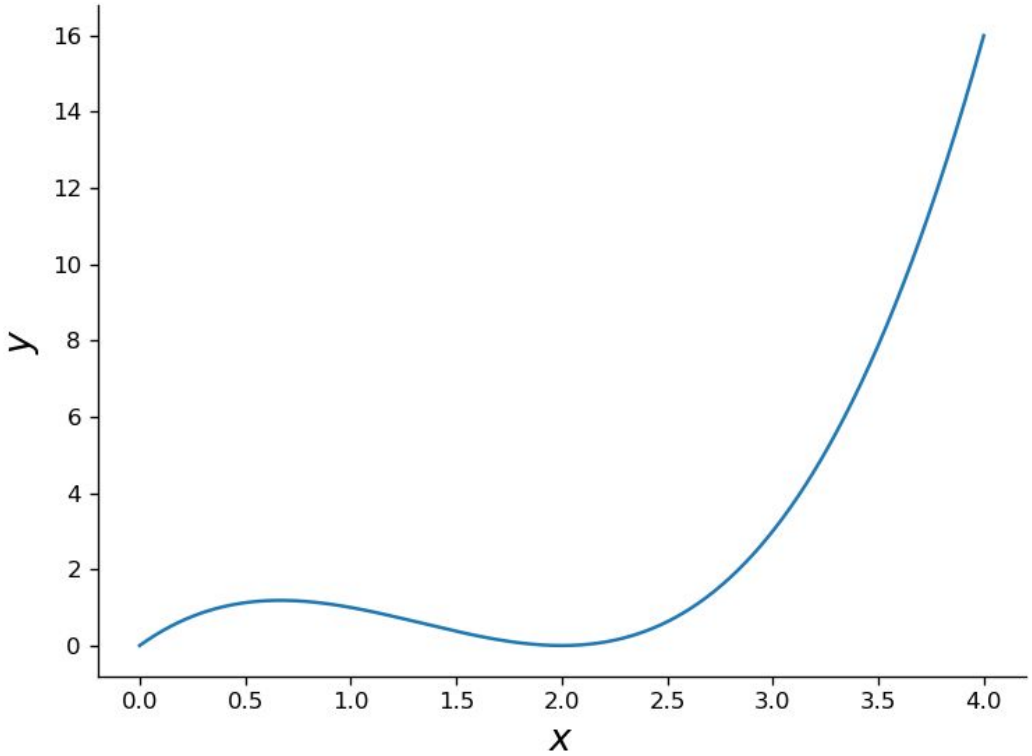


Random Forest

What is a ``Tree'' in ML?

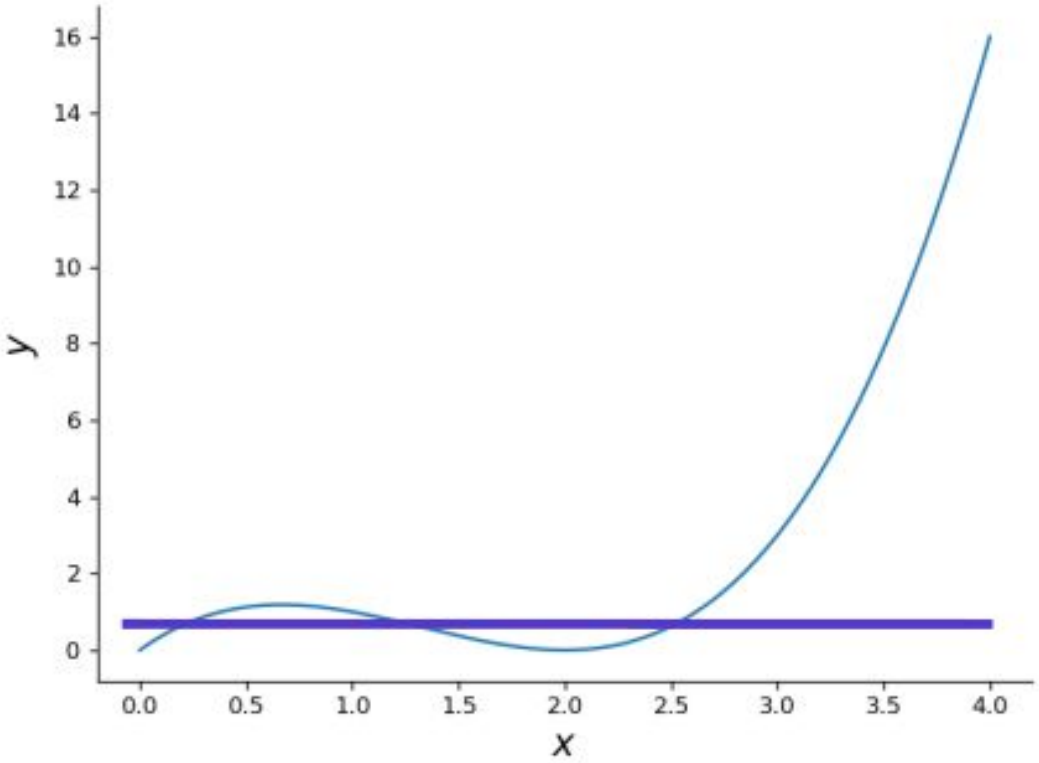


Sample Decision Tree Approach



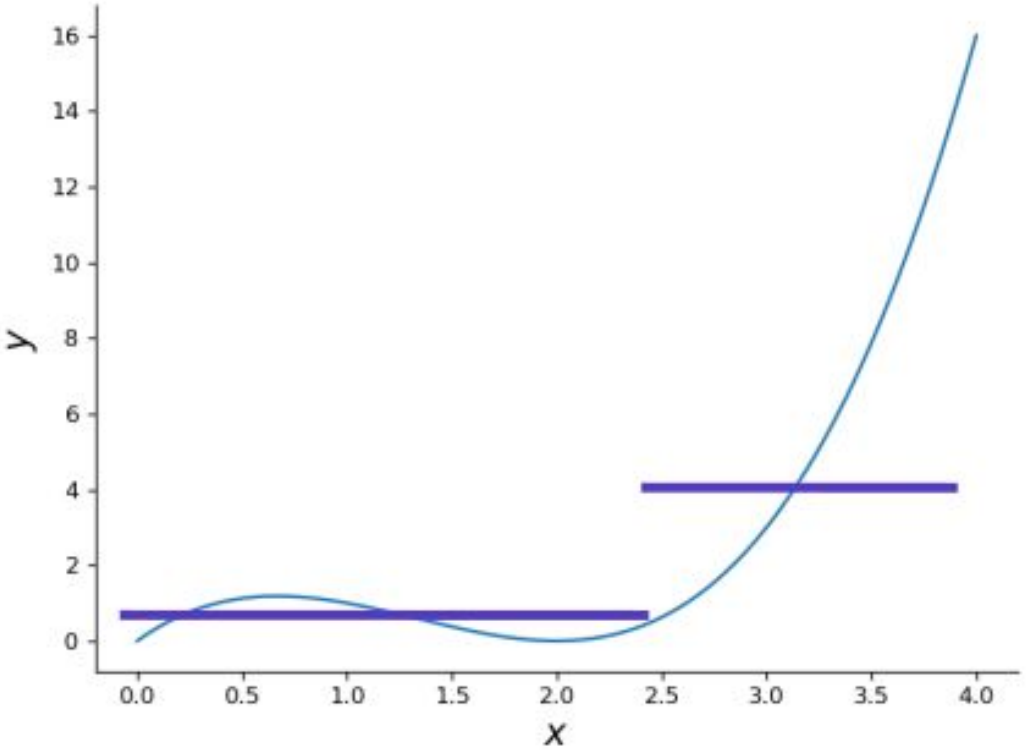
- IDEA: Approximate curve with some tuneable number of constant functions

Sample Decision Tree Approach



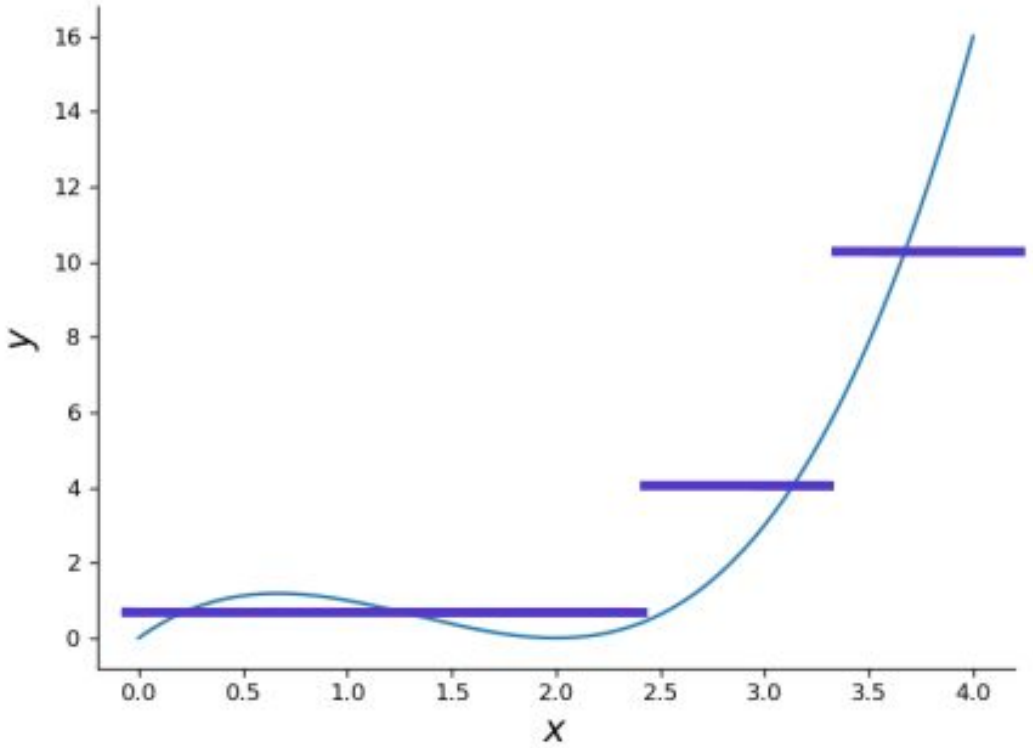
- IDEA: Approximate curve with some tuneable number of constant functions

Sample Decision Tree Approach



- IDEA: Approximate curve with some tuneable number of constant functions

Sample Decision Tree Approach



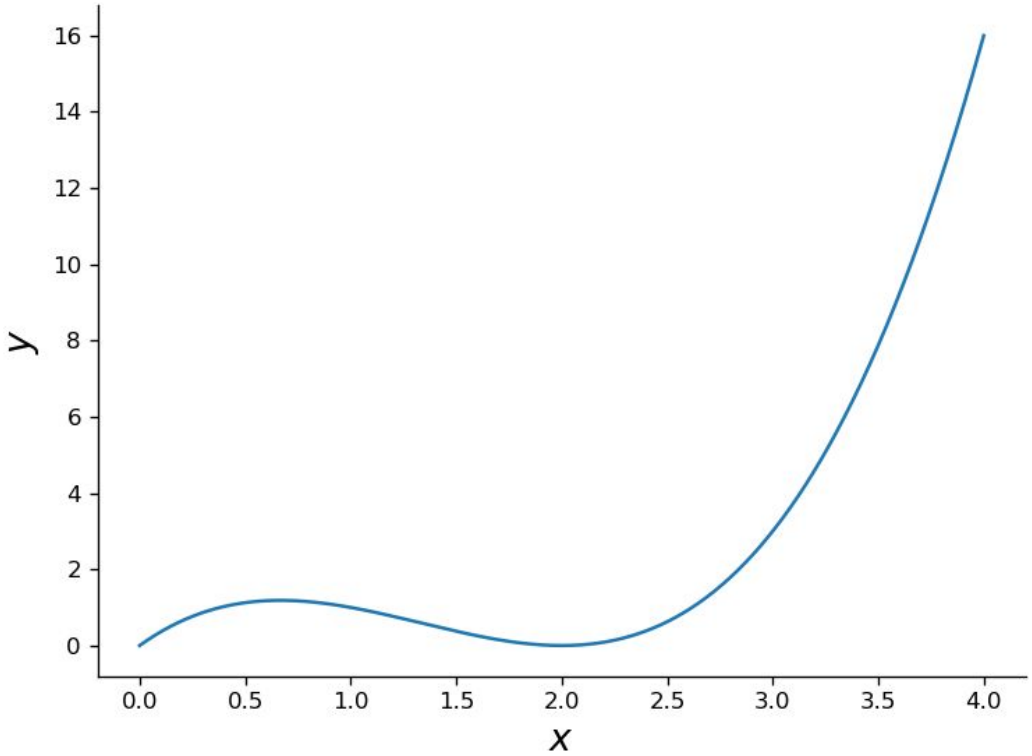
- IDEA: Approximate curve with some tuneable number of constant functions

How can we formalize this?

Sample Decision Tree Approach

- Recursively split feature space and fit a constant function to each
- Greedy choice of splits to optimize for loss function or accuracy
- The algorithm terminates once
  - A) Enough splits made, OR
  - B) constant functions have some min # of observations

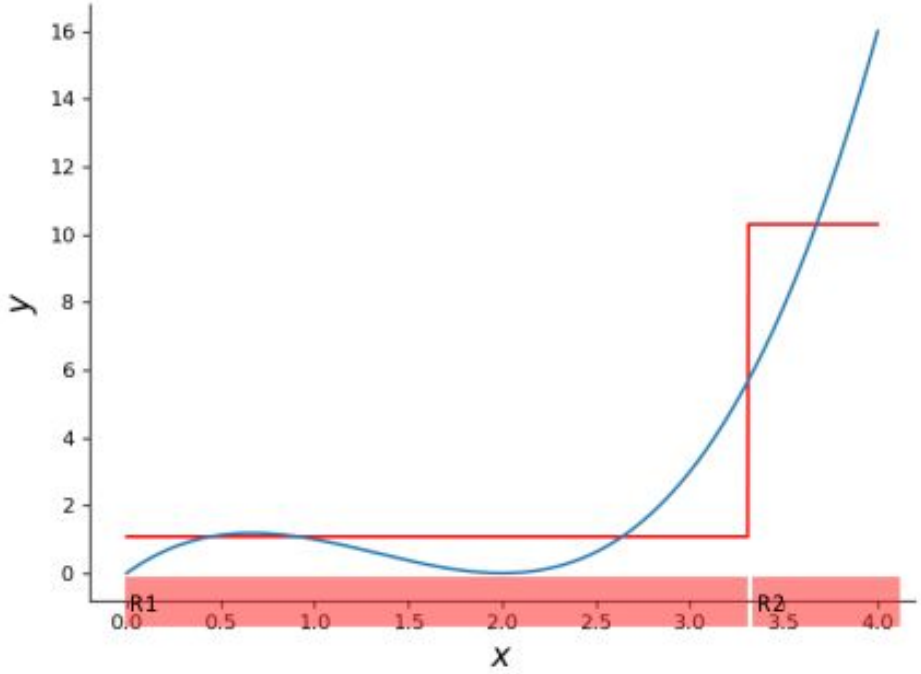
Sample Decision Tree Approach



- Back to our example… let's make cuts and fit constant functions to minimize loss function

- We haven't specified our loss function here… but it could be **MSE**, **cross-entropy loss**, etc.

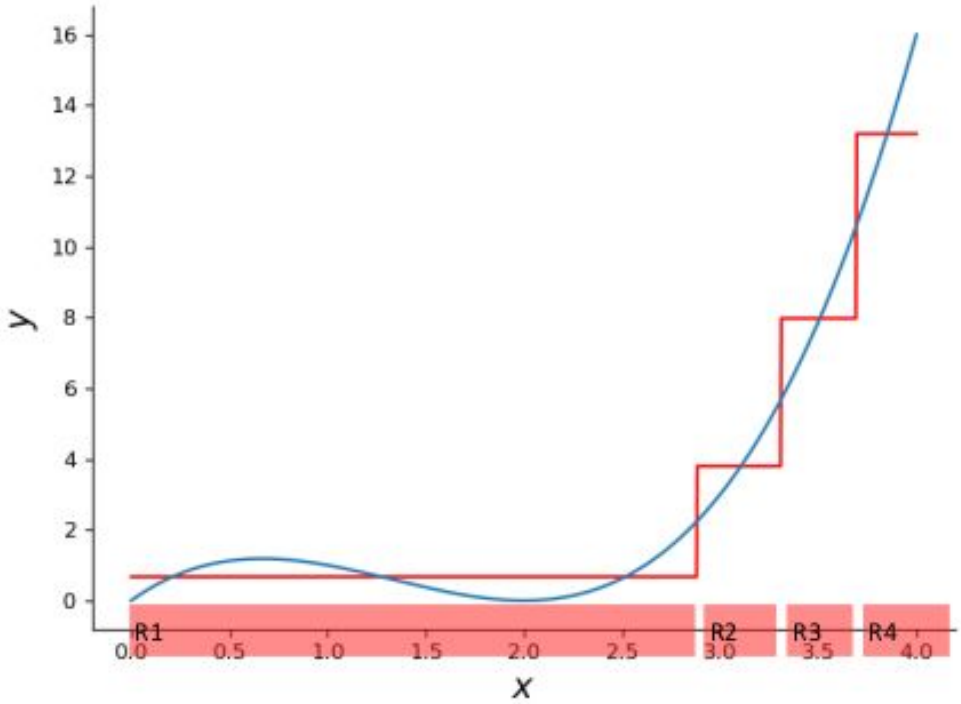
Sample Decision Tree Approach



- Back to our example… let's make cuts and fit constant functions to minimize loss function

- We haven't specified our loss function here… but it could be **MSE**, **cross-entropy loss**, etc.

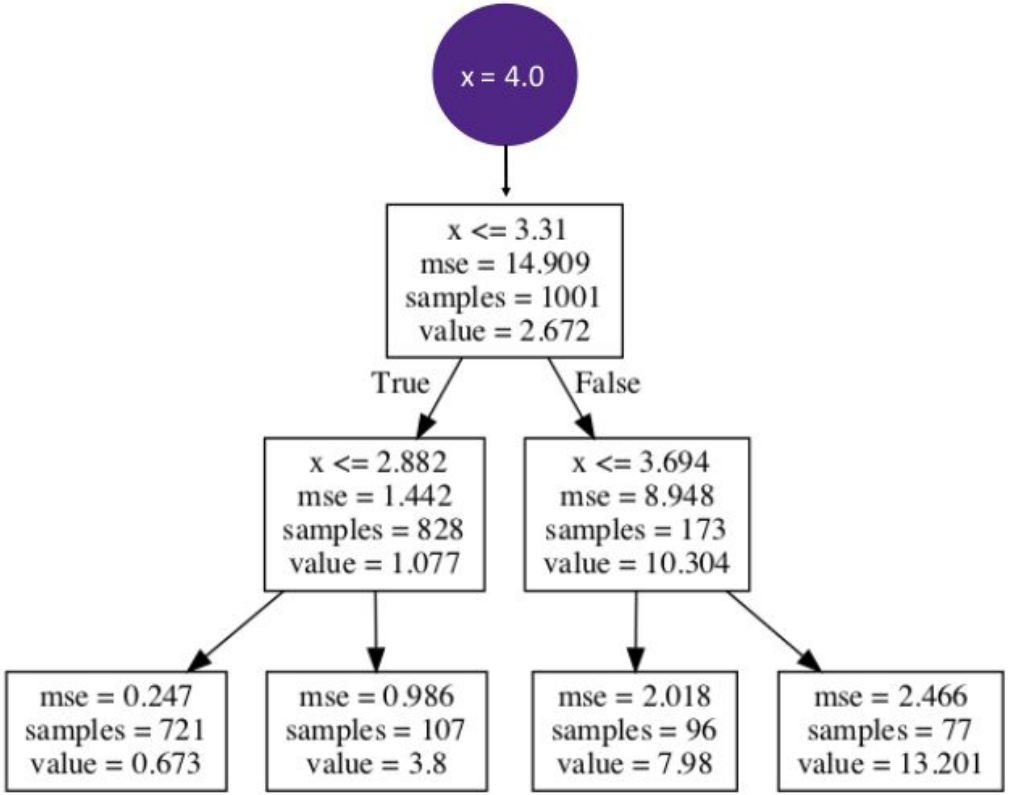
Sample Decision Tree Approach



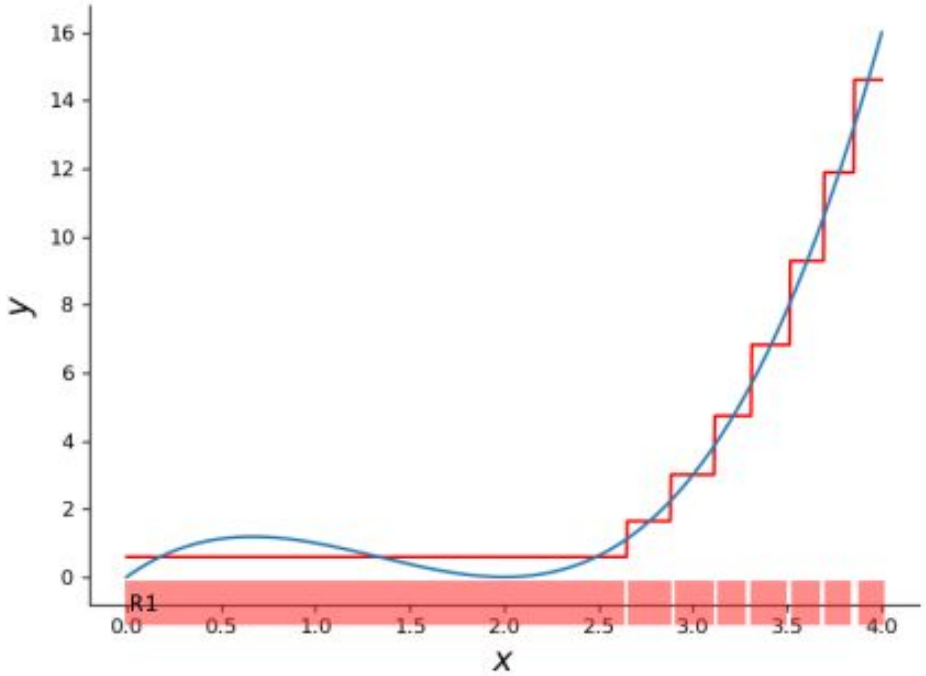
- Back to our example… let's make cuts and fit constant functions to minimize loss function

- We haven't specified our loss function here… but it could be **MSE**, **cross-entropy loss**, etc.

Decision Tree Algorithm



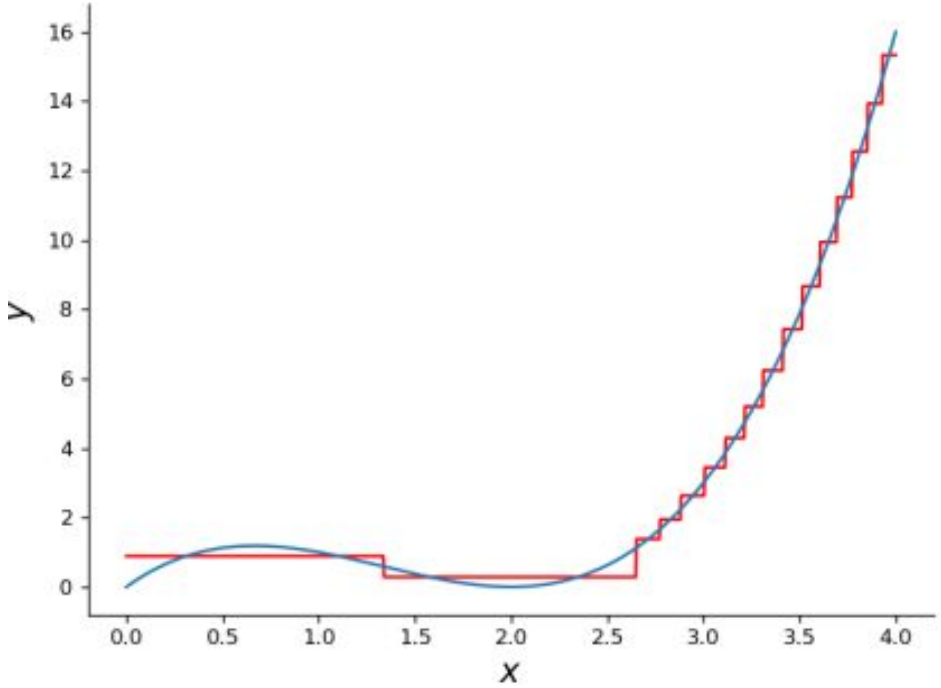
Sample Decision Tree Approach



- Back to our example… let's make cuts and fit constant functions to minimize loss function

- We haven't specified our loss function here… but it could be **MSE**, **cross-entropy loss**, etc.

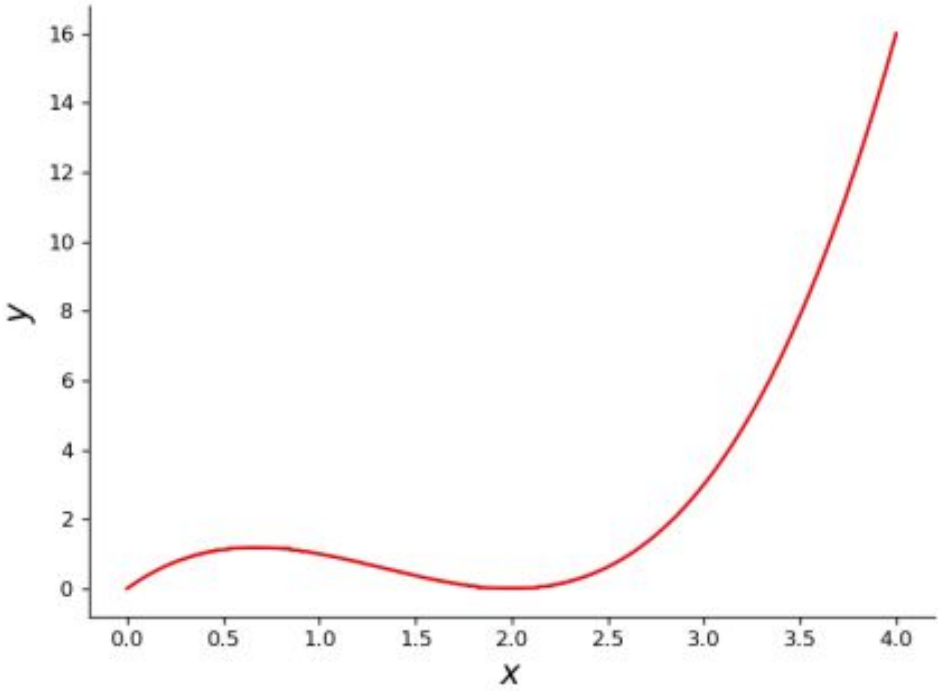
Sample Decision Tree Approach



- Back to our example… let's make cuts and fit constant functions to minimize loss function

- We haven't specified our loss function here… but it could be **MSE**, **cross-entropy loss**, etc.

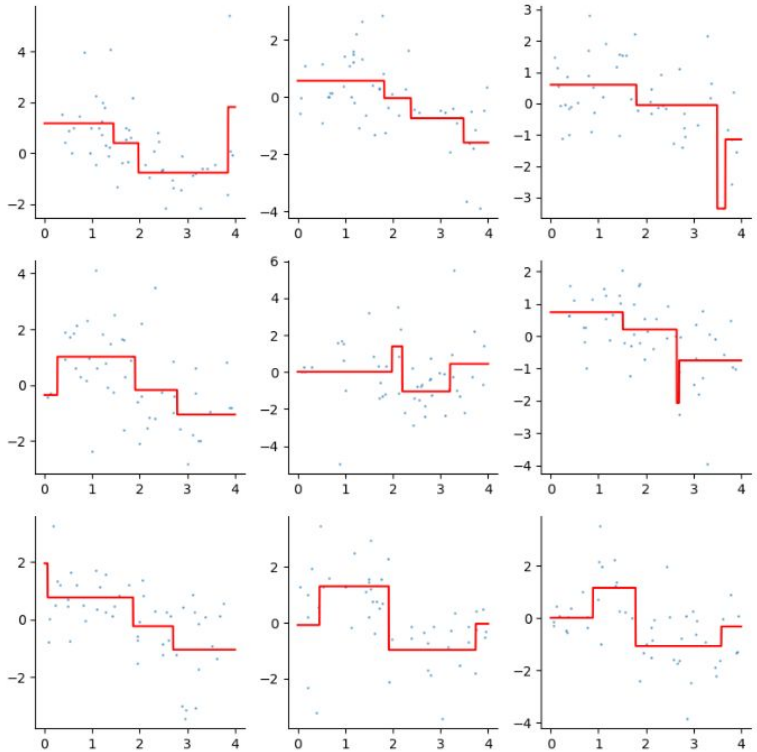
Sample Decision Tree Approach



- Back to our example… let's make cuts and fit constant functions to minimize loss function

- We haven't specified our loss function here… but it could be **MSE**, **cross-entropy loss**, etc.

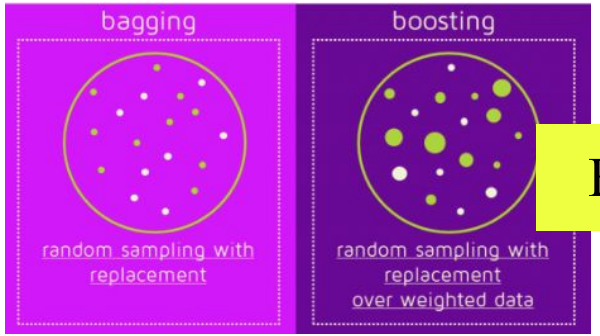
Decision Trees have **High Variance**



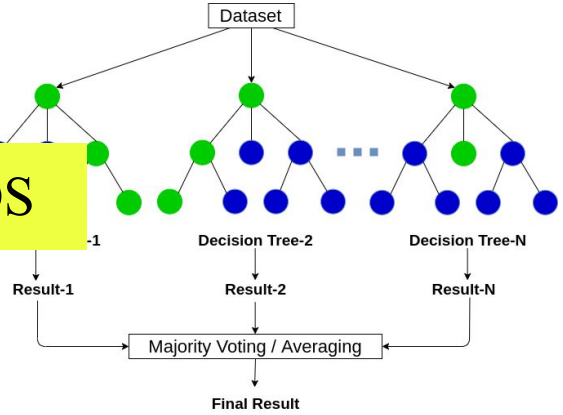
- These predictions are all from the same dataset

- Tree depth is only 2!

How to reduce this variance?



ENSEMBLE METHODS



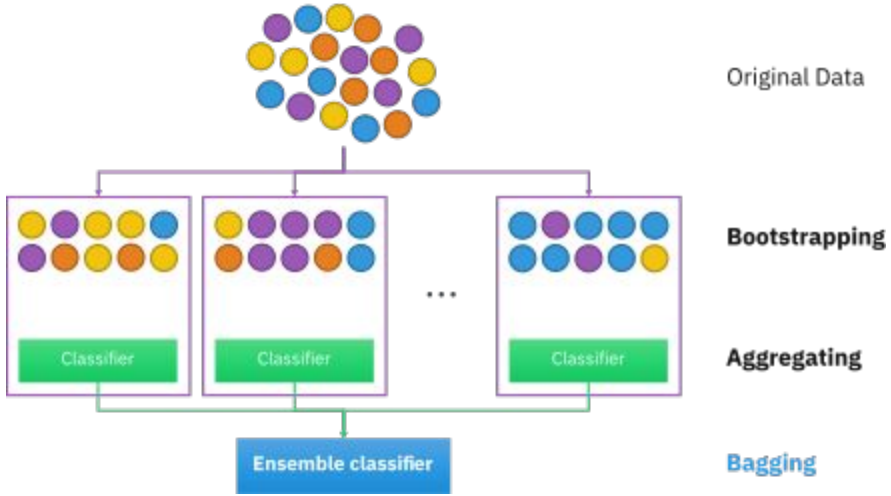
Bagging/Boosting

Random Forest

How to reduce this variance?

Bagging (**B**ootstrap **Agg**regation)

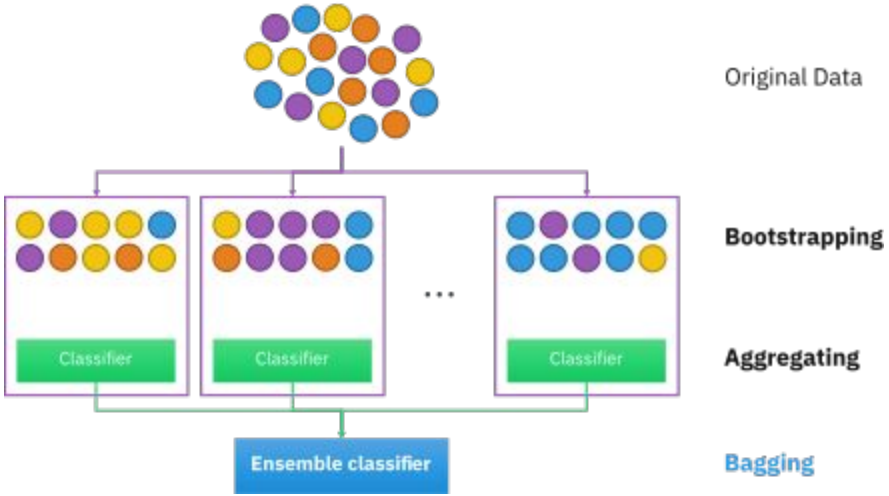
- Bootstrap some number N of datasets
- For n=1..N, fit a tree
- Combine the trees and average the N predictions



How to reduce this variance?

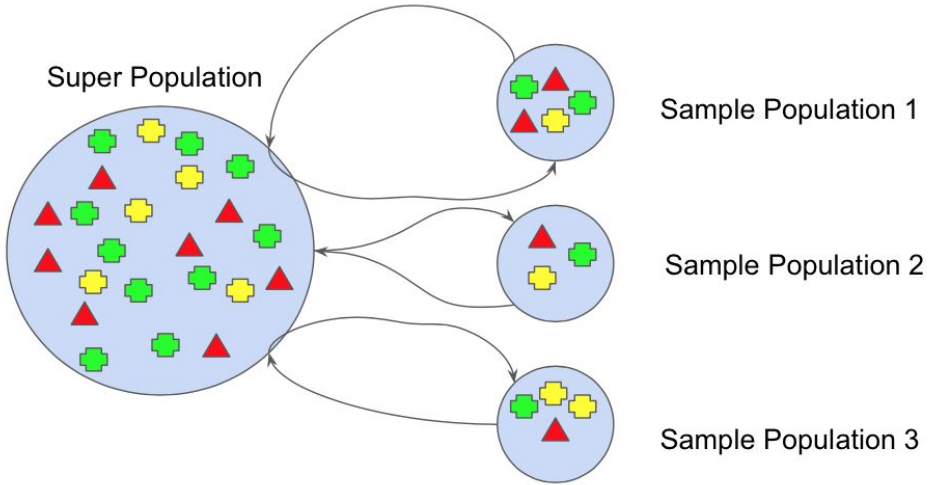
Bagging (**B**ootstrap **Agg**regation)

- Why does this reduce variance?
  - If variance is σ^2, the variance of the average is σ^2/N. As N increases, the variance of the average decreases.



Bagging for Classification

- The trees in the bag will vote for each class: [Class 1 votes, Class 2 votes, etc.]. These are *not probabilities*, but *proportions*.
- One issue in classification is that bagging a poor classifier may make it worse (ESLII pp. 286).



Random Forest

- A problem with bagging is that the bagged trees are *correlated*. This limits the benefits of averaging. Variance of uncorrelated trees is σ^2/N; Variance of correlated trees is $\rho\sigma^2 + (1-\rho)\sigma^2/N$
- What if we bagged, but instead of showing the estimator all features, we showed it a random subset which changed each time the tree had to make a split? Random Forest

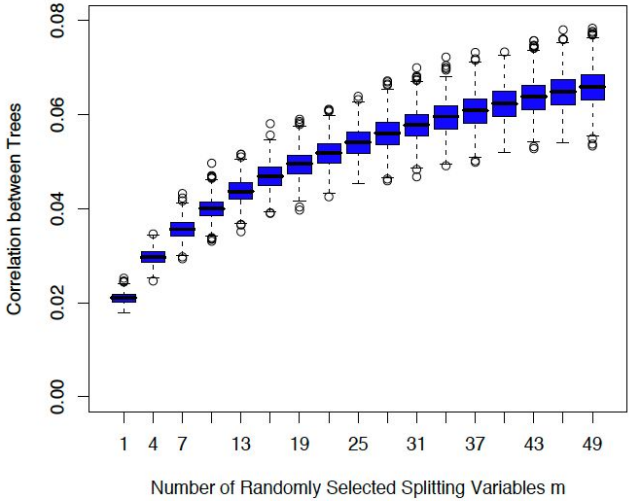
Random Forest

Steps:

1. Bootstrap N datasets
2. Grow N trees → at each option to split, tree decides where to split by random sample of features
3. Average the N predictions

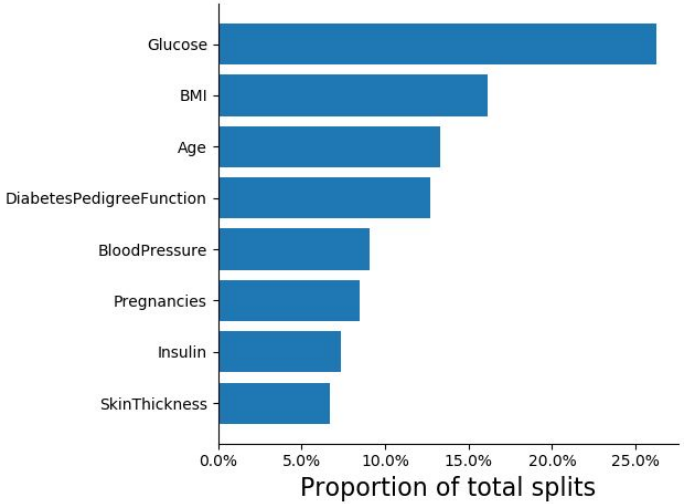
RF Reduces Correlation

- Pairs of tree predictions less correlated if they don’t use the same splitting features



RF vs. DT vs. Bagging

- RF (Random Forest) has low bias and lower variance than the other two
- It is the preferred method among the three because of this
- We can also examine the **feature importance** using RF.

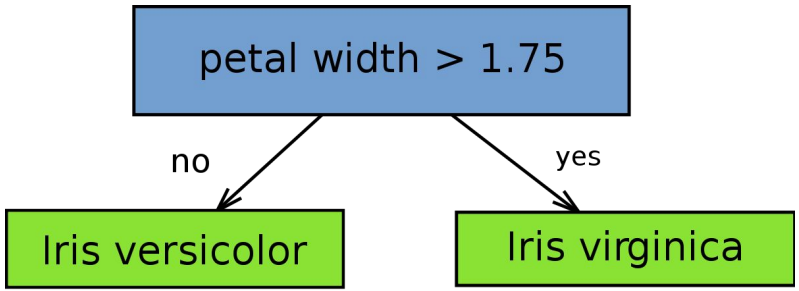


Out of Bag (OOB) Error

- Not all observations are in each bootstrapped dataset
- Can use trees that did not see samples (x_i,y_i) to make predictions → validation
  - This is called “Out of Bag” (OOB) error

Boosting

- One issue with previous techniques → if tree depth too large it can lead to overfitting; if tree depth not enough then it can lead to underfitting
- Boosting = apply a stump (tree with one split) → Example: AdaBoost.M1



AdaBoost

Similar to bagging, except:

- Trees are grown **in order** → the previous tree is used to grow the current one

1) All samples have the same weight adding to 1 (i.e. 5 samples with weight 0.2 each).
2) For the first stump, determine importance of features and set the most important feature at each split → the weight of this stump in the final prediction is based on its **total prediction error**. This comes from adding the incorrect sample weights.
3) The sample weight for any previous incorrect sample(s) increases; all others decrease.
4) Re-sample the re-weighted data (with replacement). This new data is used for the next stump.

AdaBoost Example

- Fit weak learner (stump)
- Reweigh the observations
- Fit weak learner to reweighted data
- Reweight the observations again
- …
- Prediction is weighted predictions of learners

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Yes	Yes	205	Yes	1/8
No	Yes	180	Yes	1/8
Yes	No	210	Yes	1/8
Yes	Yes	167	Yes	1/8
No	Yes	156	No	1/8
No	Yes	125	No	1/8
Yes	No	168	No	1/8
Yes	Yes	172	No	1/8

https://www.youtube.com/watch?v=LsK-xG1cLYA

AdaBoost Example

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

AdaBoost Example

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease
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	Sample Weight	Patient Weight	Heart Disease
1/8	0.07	156	No
1/8	0.07	167	Yes
1/8	0.07	125	No
1/8	0.49	167	Yes
1/8	0.07	167	Yes
1/8	0.07	172	No
1/8	0.07	205	Yes
1/8	0.07	167	Yes

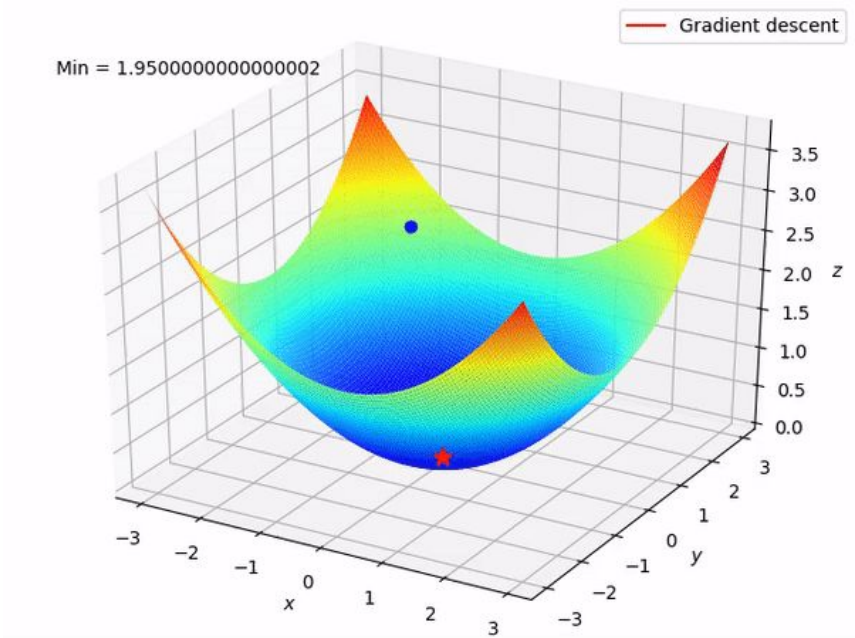
Sample Weight
1/8
1/8
1/8
1/8
1/8
1/8
1/8
1/8

Gradient Boosting

- Fit a tree to the negative gradient values of the loss using least squares + a weak learner
- Similar to gradient descent → what is gradient descent? It was briefly mentioned in Lecture 01, but let's clarify...

Gradient Descent

- From the current position, move in the direction of steepest descent (i.e. along the gradient)

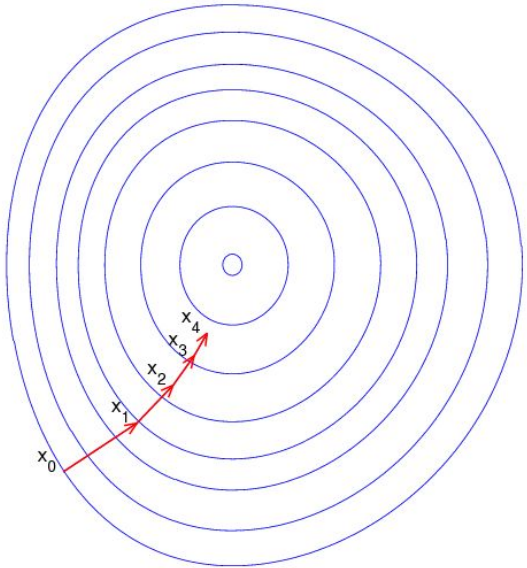


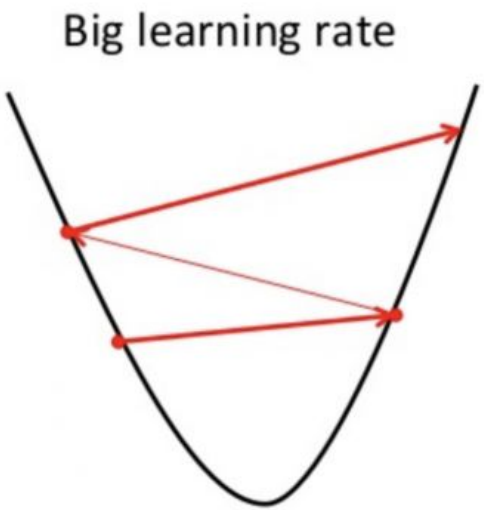
Gradient Descent

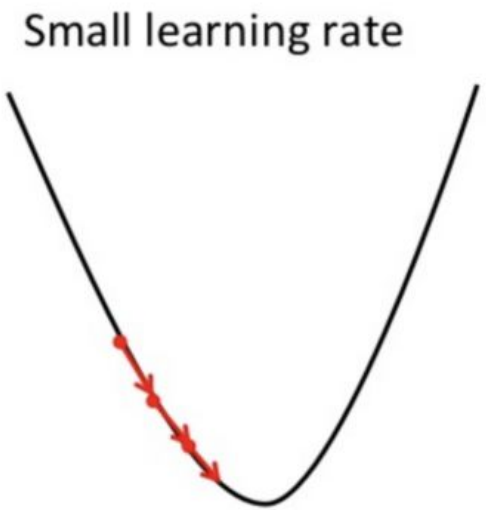
Learning rate

(This is a vector)

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \alpha\left(\nabla f\left(\mathbf{x}_n\right)\right)$$







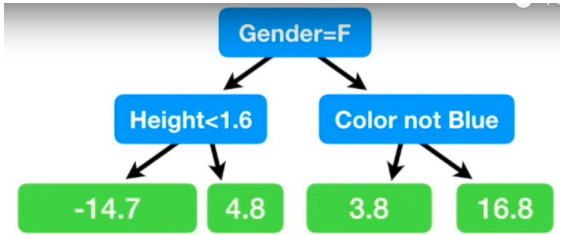
https://builtin.com/data-science/gradient-descent

Gradient Boosting (Simple Regression)

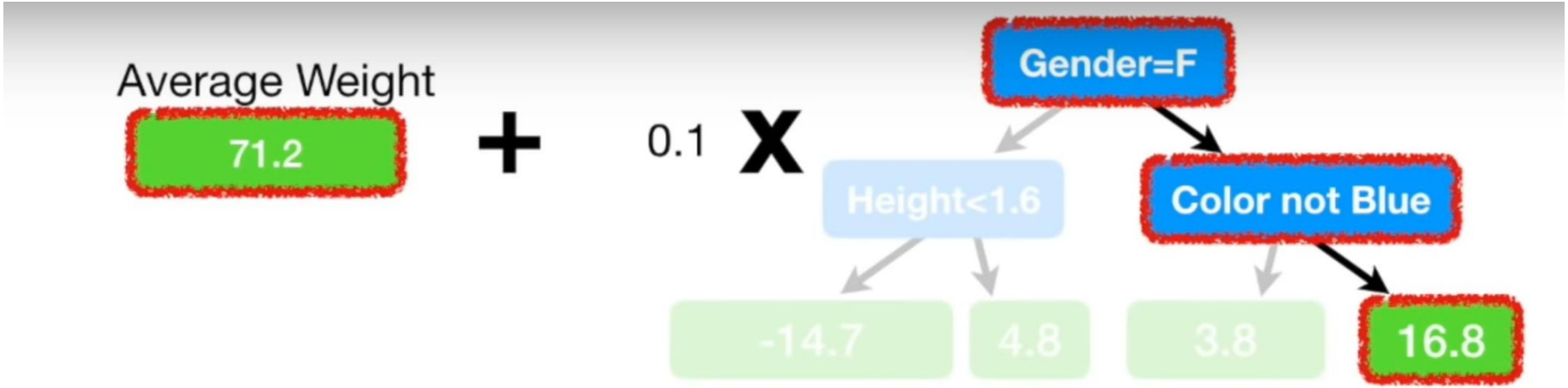
1) Instead of a stump (i.e. in AdaBoost), make a single leaf (initial guess). You can use the average value for regression.
2) Build a tree (size-restricted) from residuals of previous tree (the leaf in this case). Get a prediction from the leaf, multiply this by learning rate and add to the leaf value.
3) Use the previous tree to calculate the residuals; these should be smaller. Build a new tree (size-restricted) from those residuals. Get new predictions, re-calculate residuals.
4) …
5) When done, get a prediction for a particular test input value by adding the initial guess (the single leaf) to all weighted (multiplied by learning rate) tree predictions

Gradient Boosting Example

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



Gradient Boosting Example

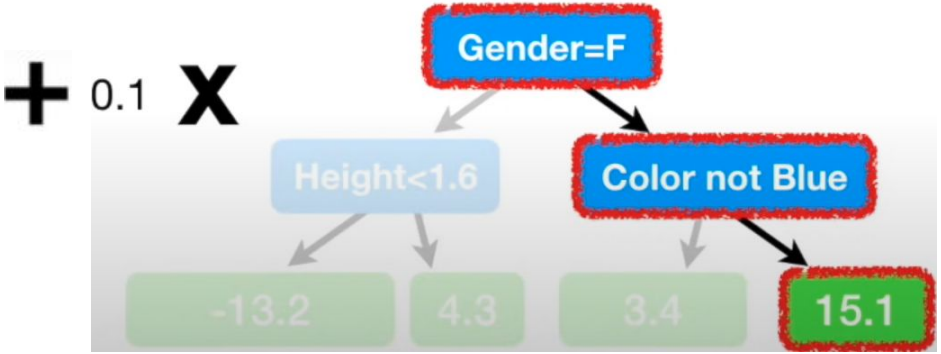


Predicted weight = 71.2 + (0.1*16.8) = 72.9

Gradient Boosting Example

Height (m)	Favorite Color	Gender	Weight (kg)	Residual	Residual
1.6	Blue	Male	88	16.8	15.1
1.6	Green	Female	76	4.8	4.3
1.5	Blue	Female	56	-15.2	-13.7
1.8	Red	Male	73	1.8	1.4
1.5	Green	Male	77	5.8	5.4
1.4	Blue	Female	57	-14.2	-12.7

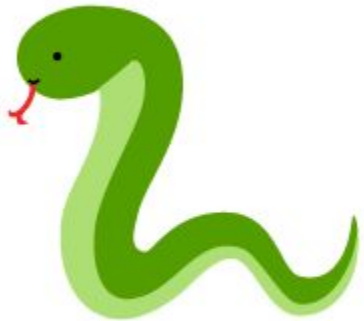
Gradient Boosting Example



Predicted weight = 71.2 + (0.1*16.8) + (0.1*15.1)

= 74.4

Let’s try it in Python...



Summary

- Decision Trees have high variance, but we have solutions for this…
- Bagging → Bootstrap datasets, fit trees and average the predictions
  - Correlation is an issue
- Random Forest → Bootstrap datasets, fit random sample of features and average predictions
  - Correlation not an issue
  - Can use for feature importance
- Boosting → Apply weak learner (stump), get weighted average of learner predictions
  - AdaBoost
  - Gradient Descent Revisited
  - Gradient Boosting