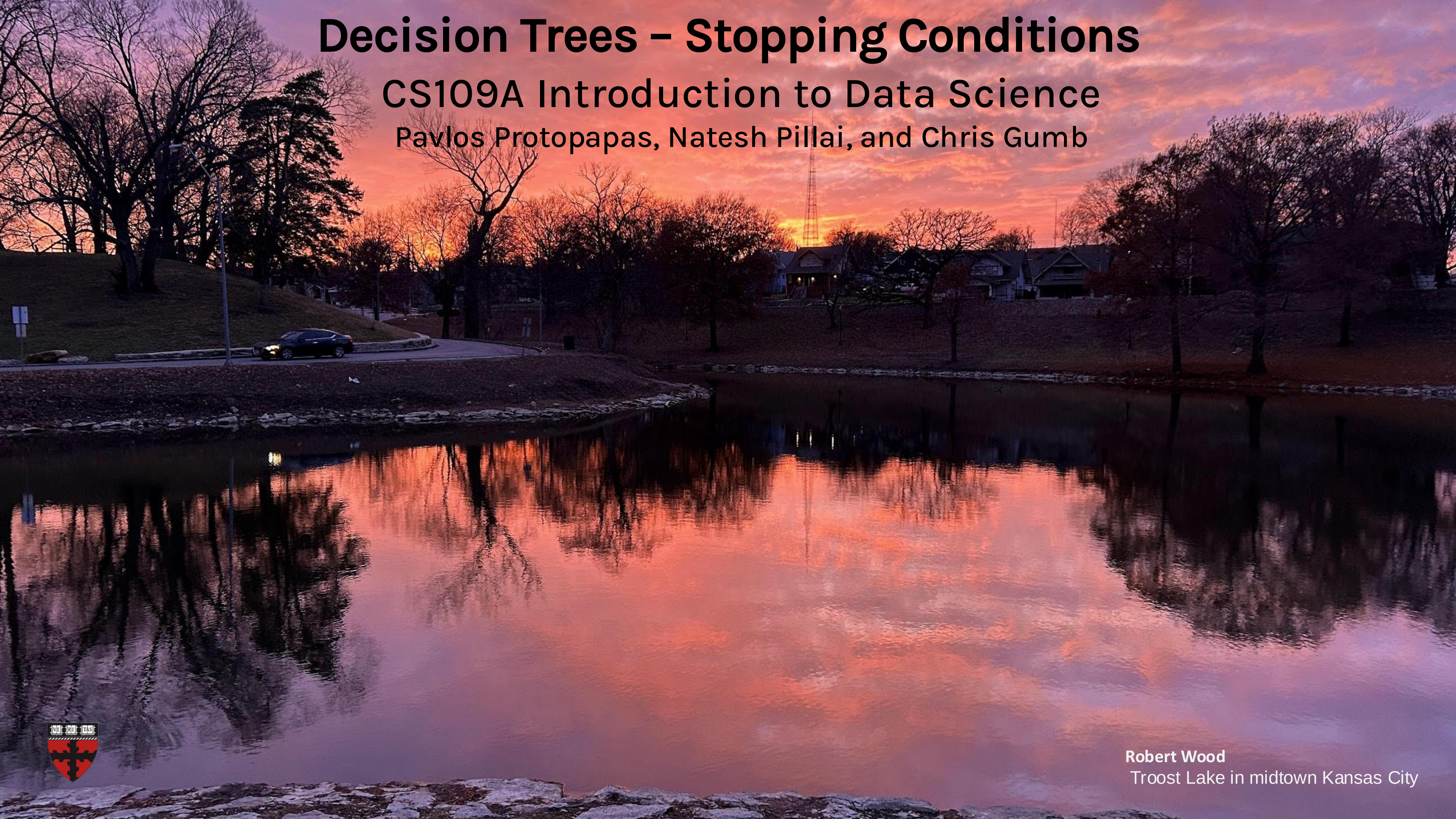
Decision Trees – Stopping Conditions

CS109A Introduction to Data Science

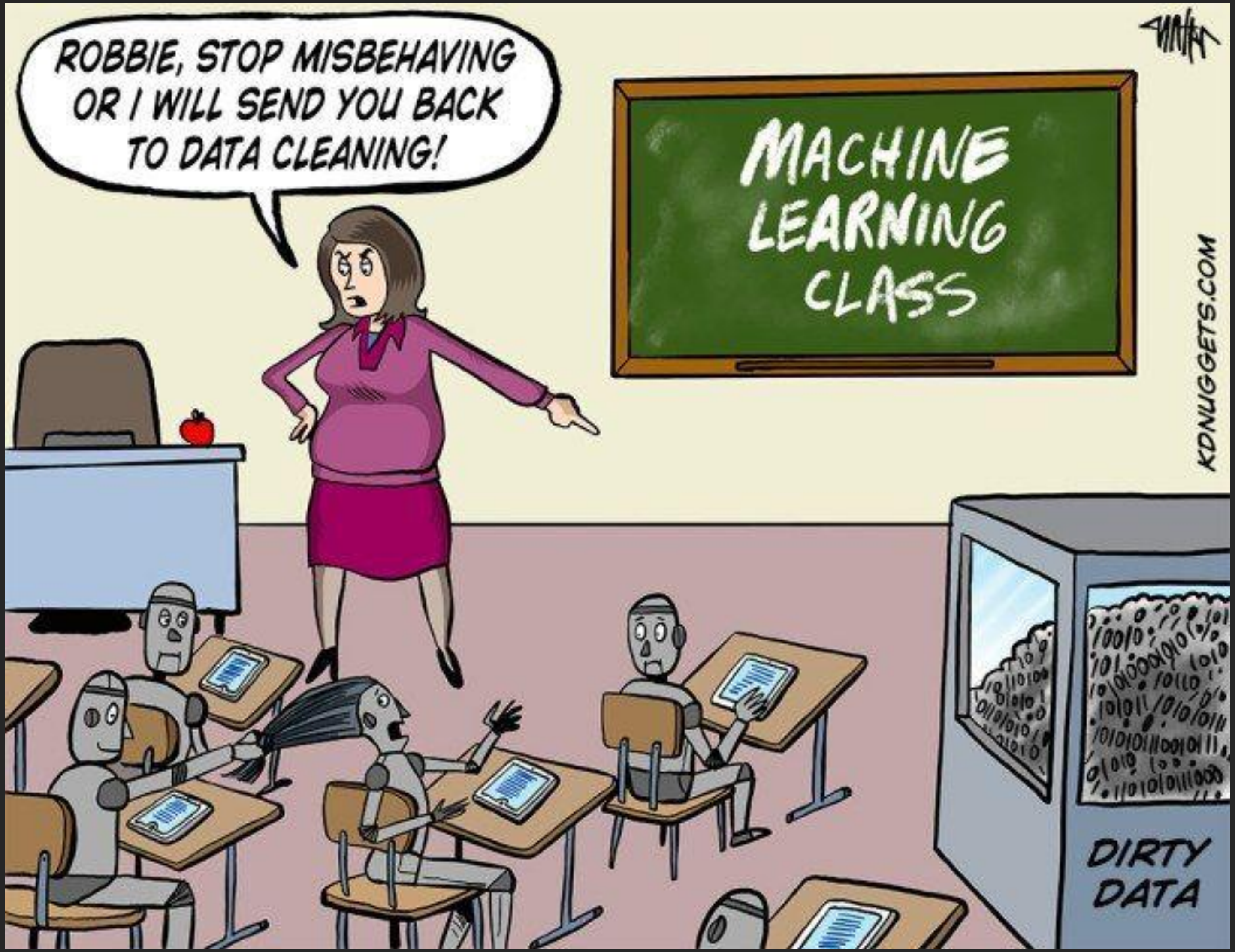
Pavlos Protopapas, Natesh Pillai, and Chris Gumb



Robert Wood
Troost Lake in midtown Kansas City

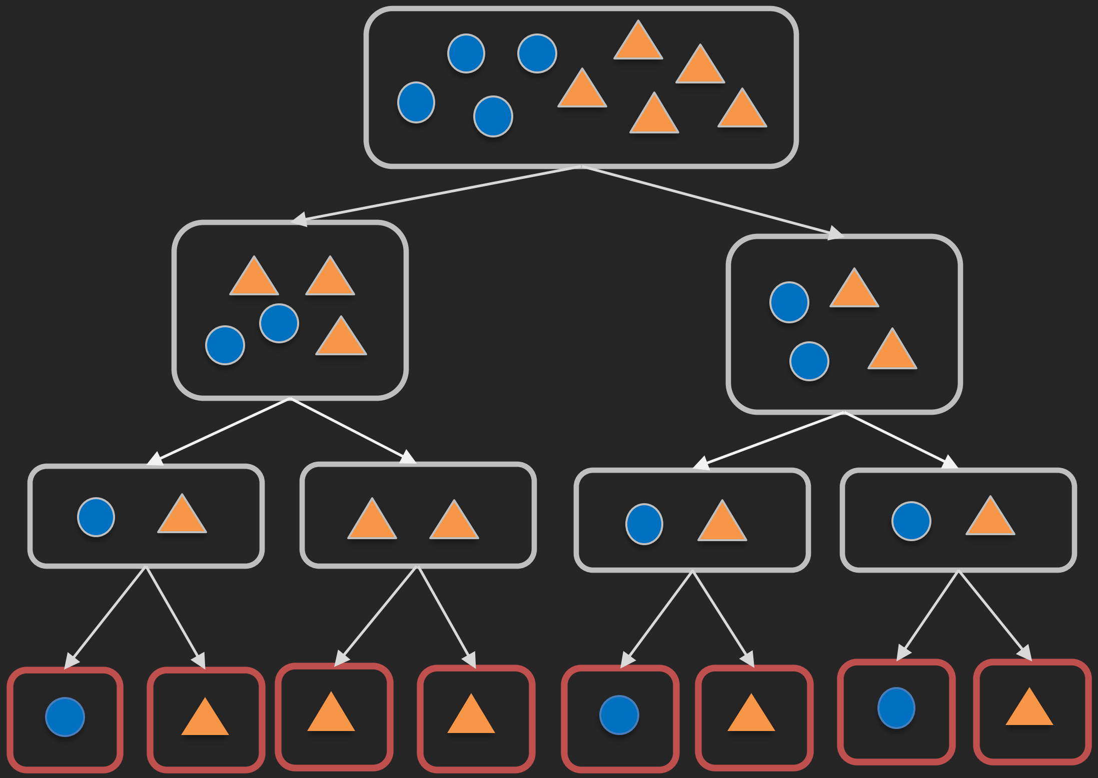
Outline

- Motivation
- Decision Trees – Classification
    - Intuition
    - Predictions
    - Splitting Criteria
    - Stopping Conditions


ROBBIE, STOP MISBEHAVING OR I WILL SEND YOU BACK TO DATA CLEANING!
MACHINE LEARNING CLASS
KDNUGGETS.COM
DIRTY DATA

Stopping Conditions

Question: If we don't terminate the decision tree algorithm manually, what will the leaf nodes of the decision tree look like?



The tree will continue to grow until each region contains **exactly one training point** and the model attains 100% **training** accuracy.

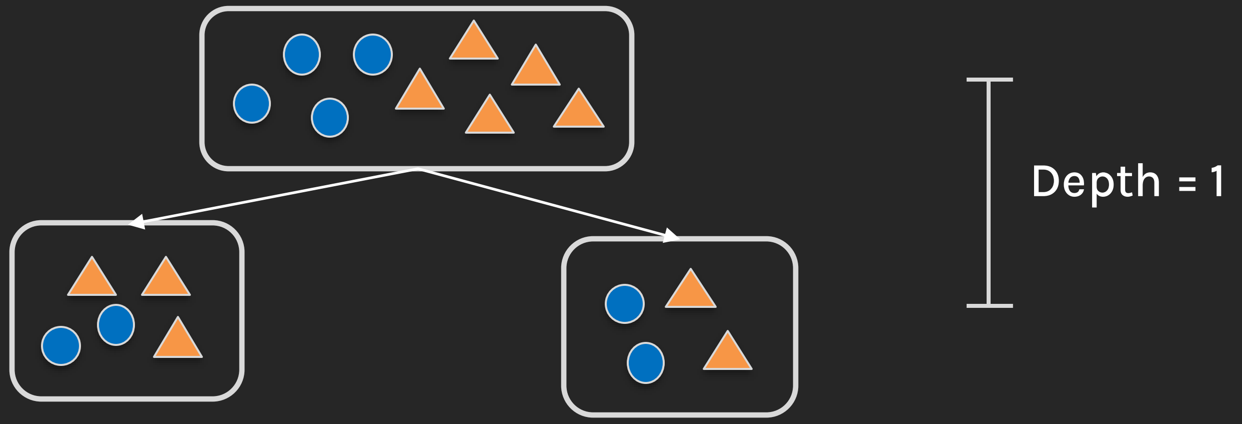
Stopping Conditions

Question: How can we prevent this from happening?

Stopping Conditions

The most common stopping condition is to limit the maximum depth *(max_depth)* of the tree.

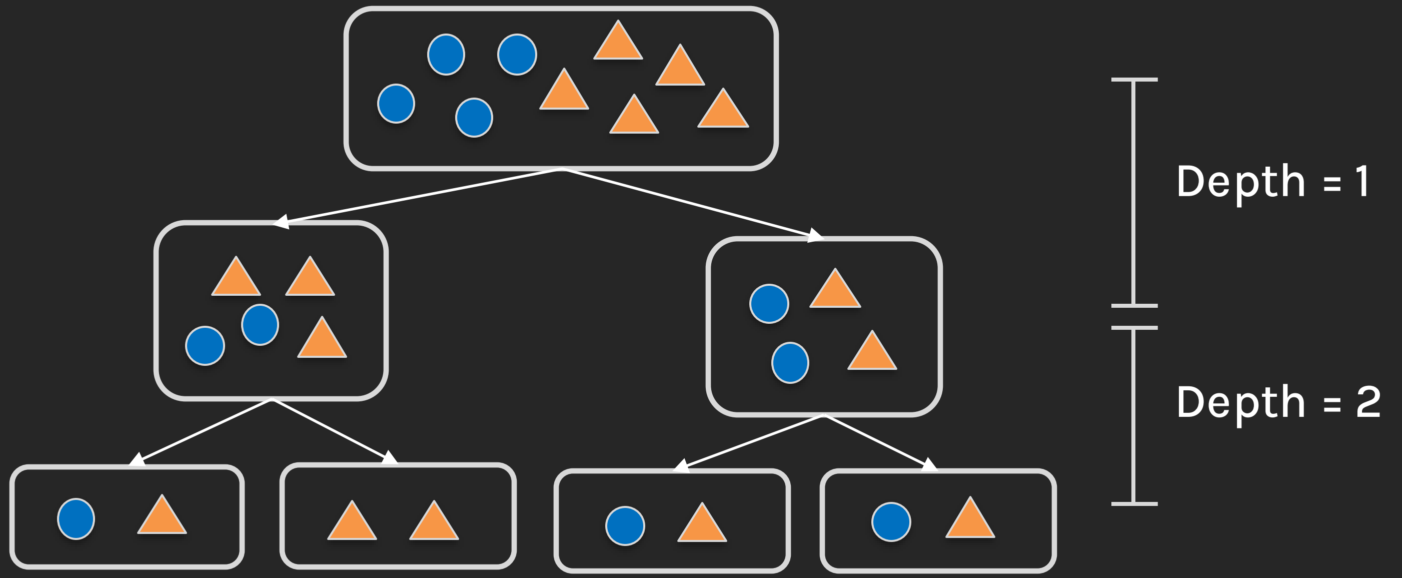
max_depth = 1



Stopping Conditions

The most common stopping condition is to limit the maximum depth (max_depth) of the tree.

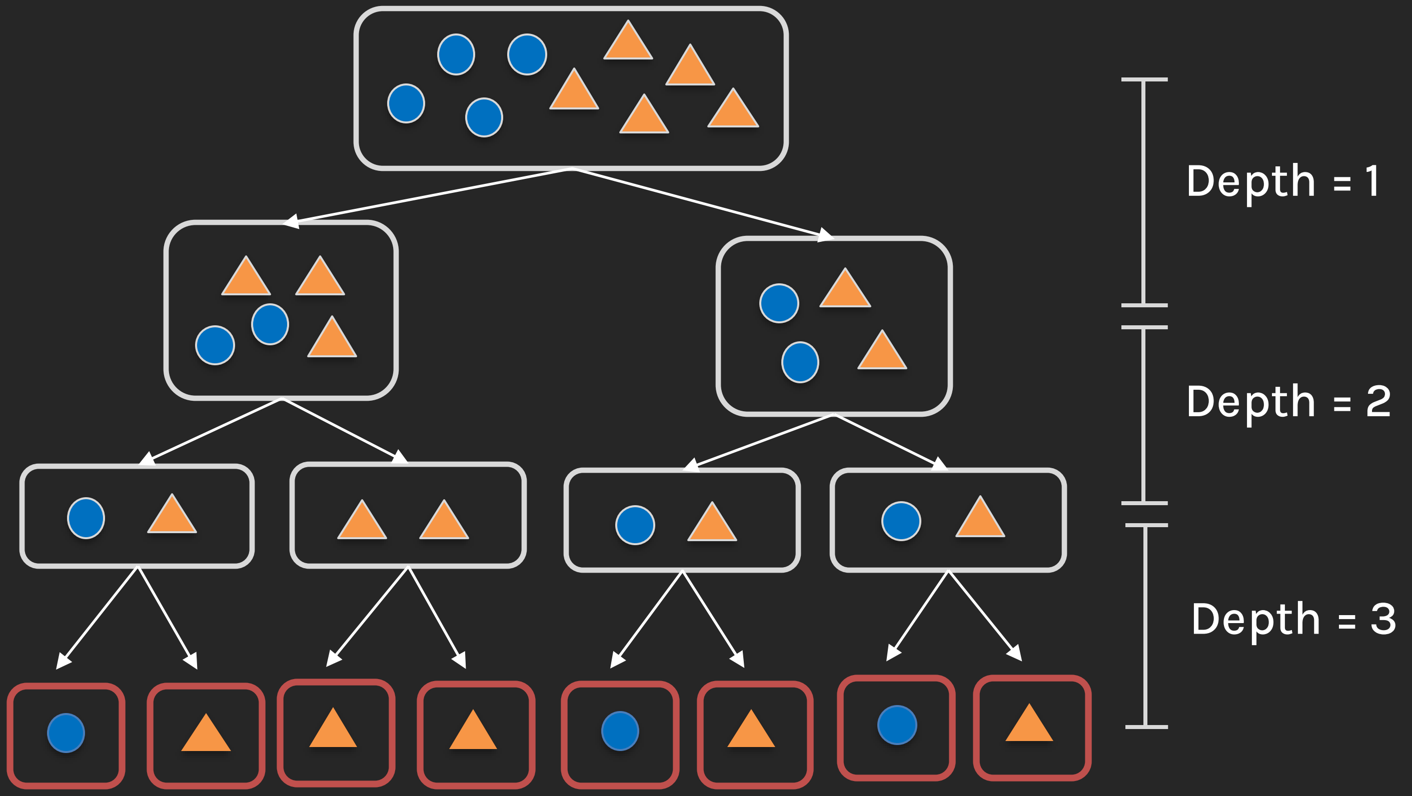
$max\_depth = 2$



Stopping Conditions

The most common stopping condition is to limit the maximum depth *(max_depth)* of the tree.

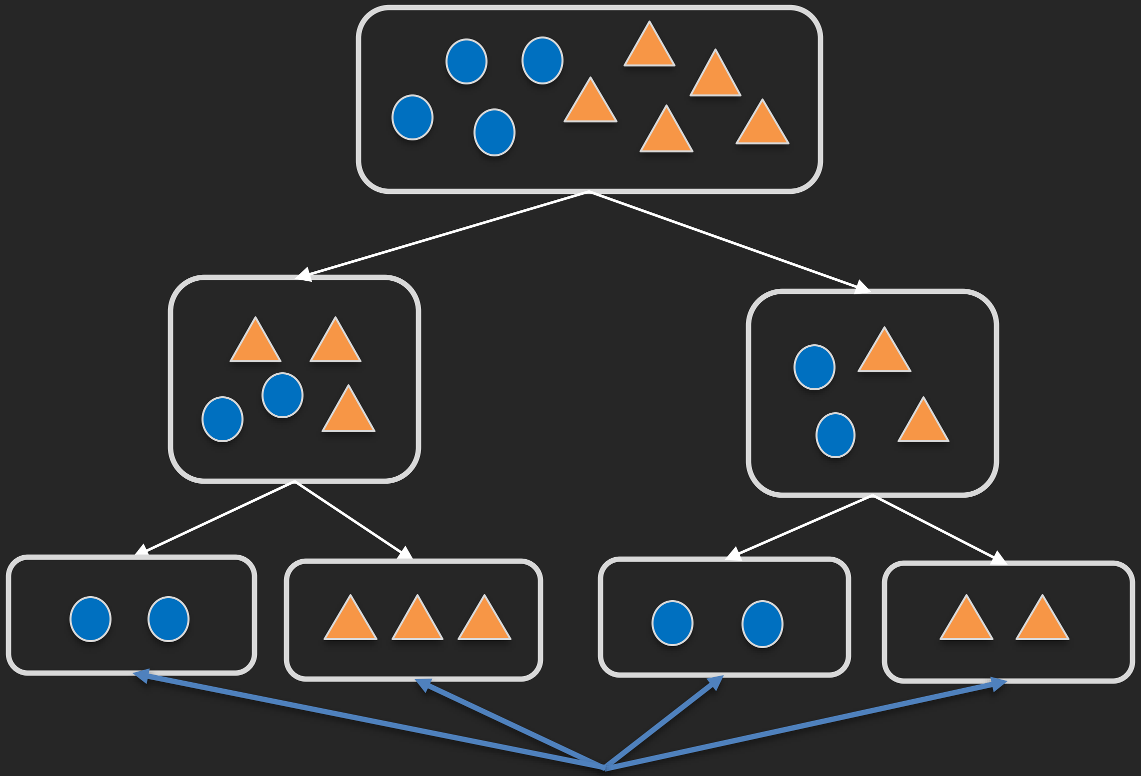
max_depth = 3



Stopping Conditions

Other common simple stopping conditions are:

- Don't split a region if all instances in the region belong to the same class.



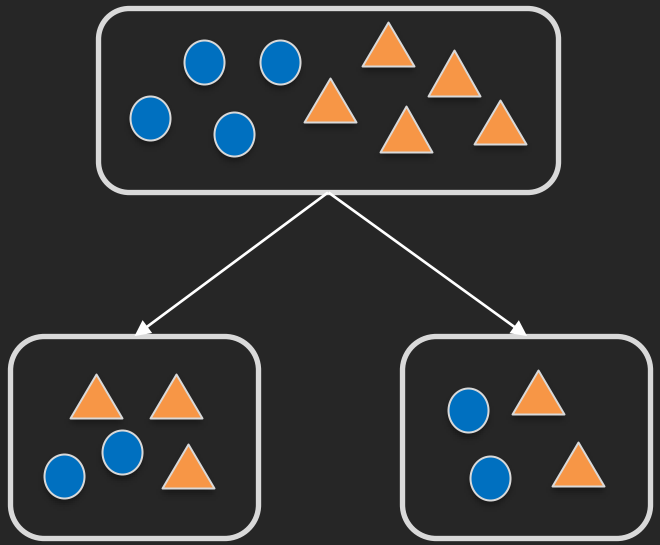
Pure leaf Nodes

Stopping Conditions

Other common simple stopping conditions are:

- Don't split a region if the number of instances in any of the sub-regions will fall below pre-defined threshold *(min_samples_leaf)*.

min_samples_leaf = 4



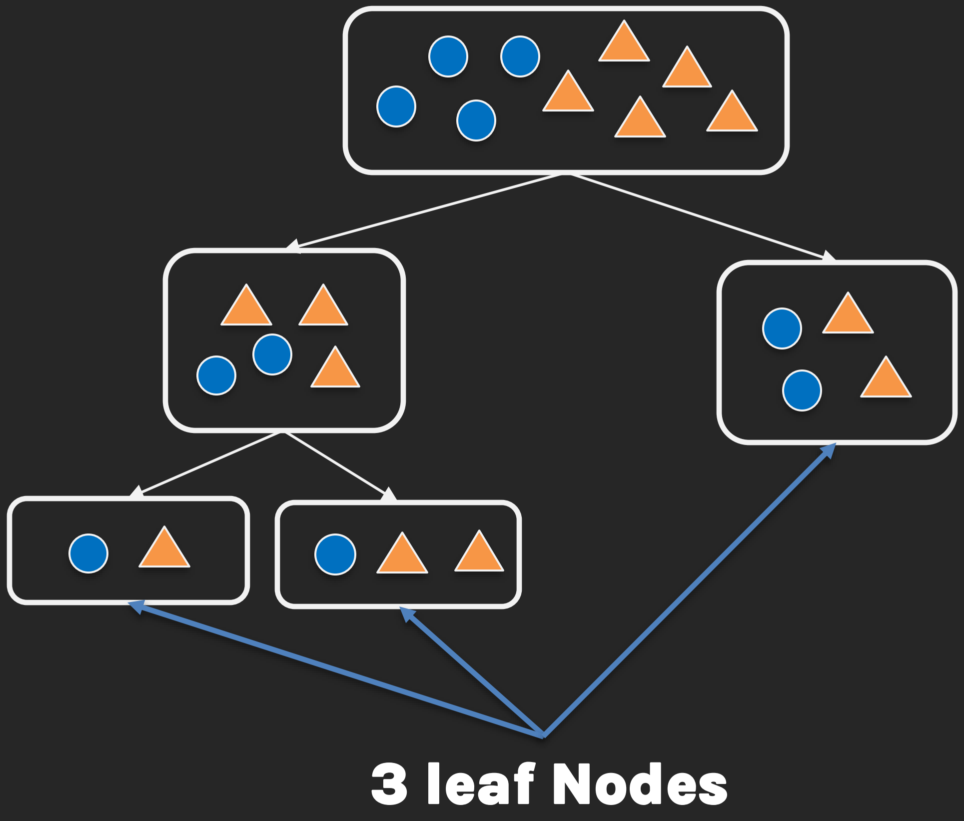
Stopping Conditions

Other common simple stopping conditions are:

- Don’t split a region if the total number of leaves in the tree will exceed a pre-defined threshold *(max_leaf_nodes)*.

max_leaf_nodes = 3

Do you see any issue with this?



Stopping Conditions

Normally, Sklearn grows trees in what is called **'level-order'**-fashion until a stopping condition such as *max_depth* is met.

However, if a value for *max_leaf_nodes* is specified, Sklearn will instead grow the tree in a **'best-first'** fashion.

But what do level-order and best-first growth ***mean***?

Stopping Conditions

Normally, Sklearn grows trees in what is called **'level-order'**-fashion until a stopping condition such as *max_depth* is met.

However, if a value for *max_leaf_nodes* is specified, Sklearn will instead grow the tree in a **'best-first'** fashion.

But what do level-order and best-first growth ***mean***?

Level-order is also sometimes called '*breadth*-first'-search but we will use the term 'level-order' so it won't be confused with the similar sounding '*best-first*' search.

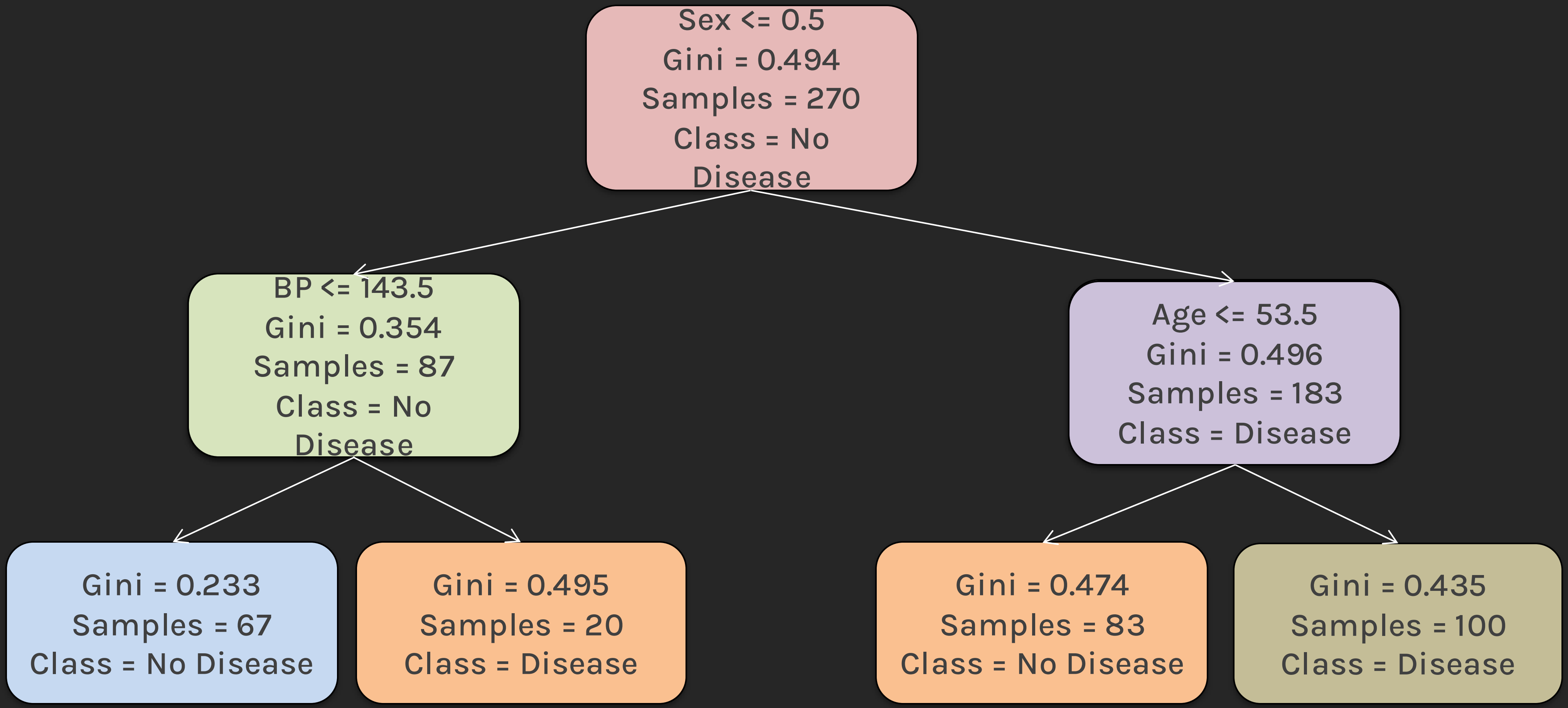
Example 1: Level-Order

Consider the following decision tree with max_depth=2 that predicts if a person has heart disease based on age, sex, BP and cholesterol:

Gini = 0.494
Samples = 270
Class = No Disease

Example 1: Level-Order

Consider the following decision tree with max_depth=2 that predicts if a person has heart disease based on age, sex, BP and cholesterol:



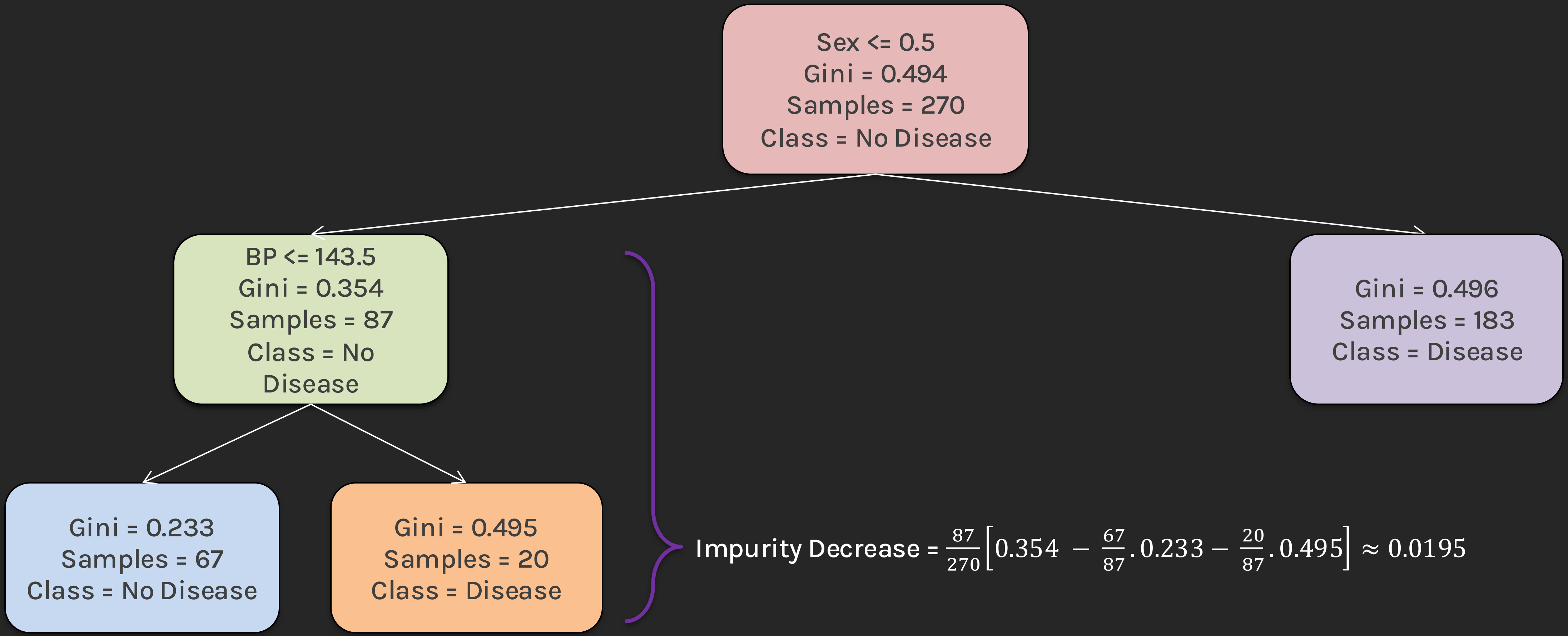
Example 1: Best-first growth

Sklearn determines the best split based on **impurity decrease**. The resulting tree will be the same when fully grown, just the order in which it is built is different.

Gini = 0.494
Samples = 270
Class = No Disease

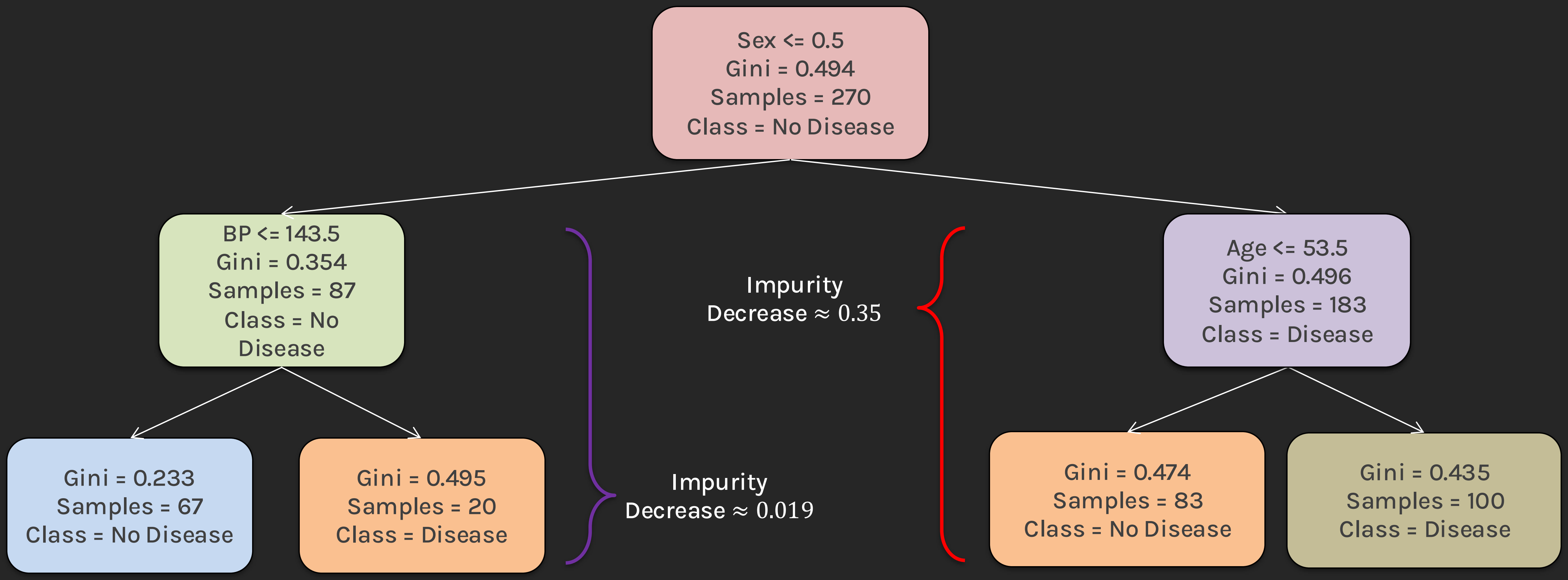
Example 1: Best-first growth

Sklearn determines the best split based on **impurity decrease.** The resulting tree will be the same when fully grown, just the order in which it is built is different.

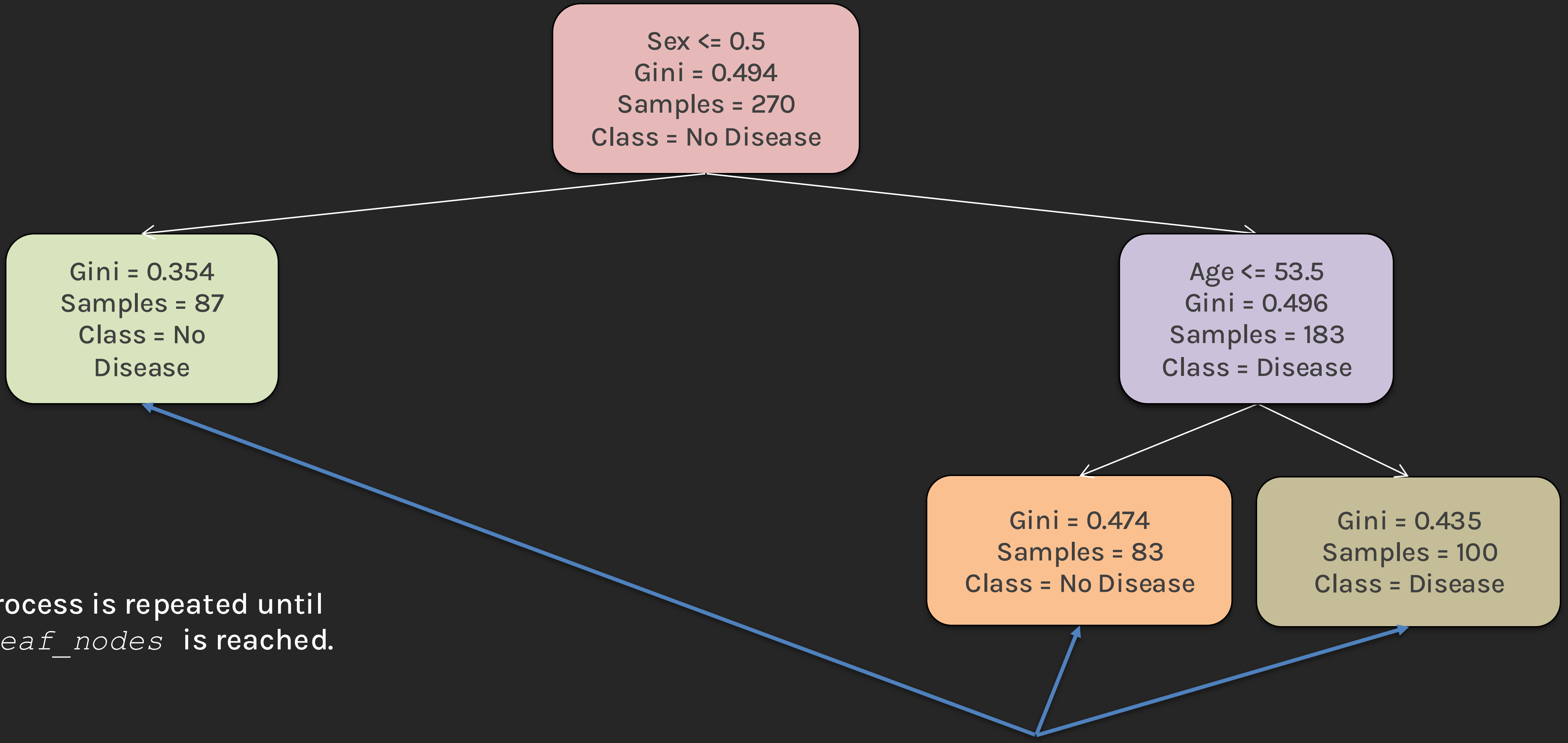


Impurity Decrease = $\frac{87}{270}\left[0.354 - \frac{67}{87}.0.233 - \frac{20}{87}.0.495\right] \approx 0.0195$

Example 1: Best-first growth



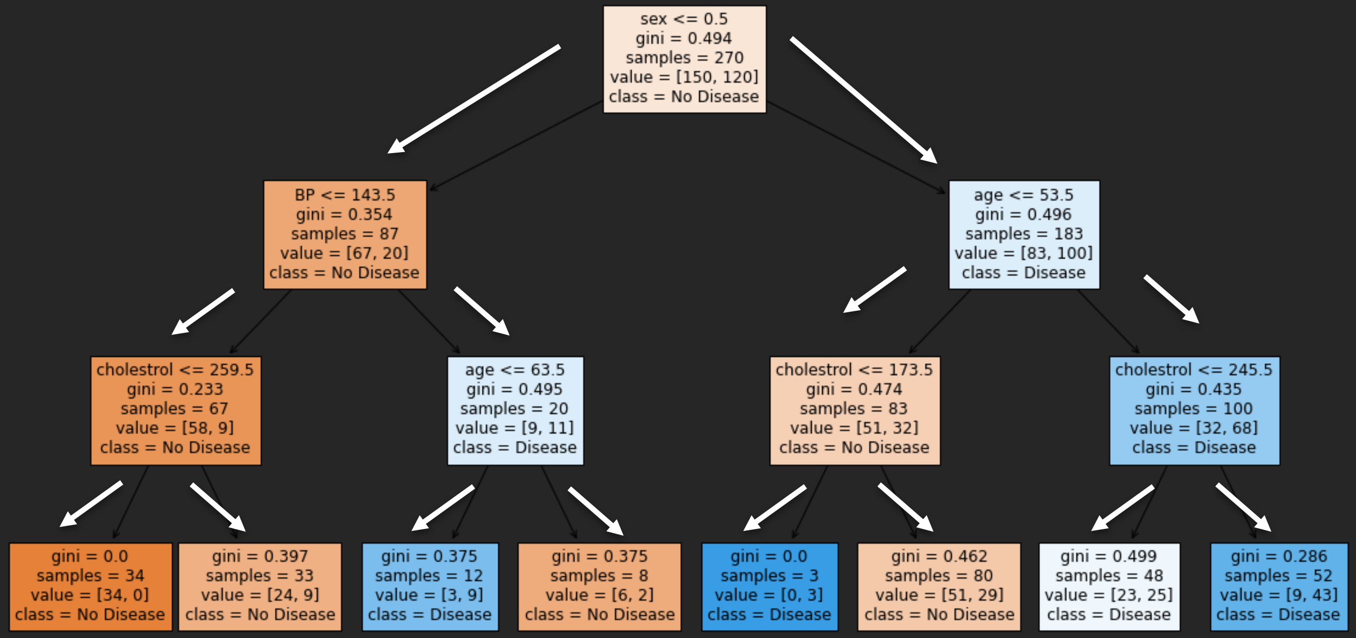
Example 1: Best-first growth



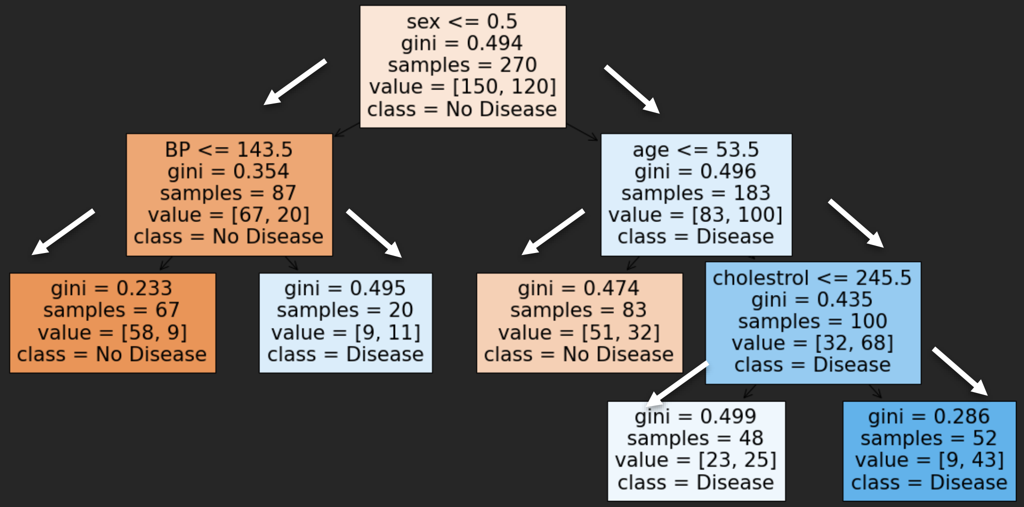
This process is repeated until *max_leaf_nodes* is reached.

Now we compare the impurity decrease when splitting these 3 nodes!

Example 2: Level-order vs Best-first growth



$max\_depth = 3$



$max\_leaf\_nodes = 5$

Stopping Conditions

A more restrictive stopping condition is:

Compute the gain in purity of splitting a region R into R_1 and R_2:

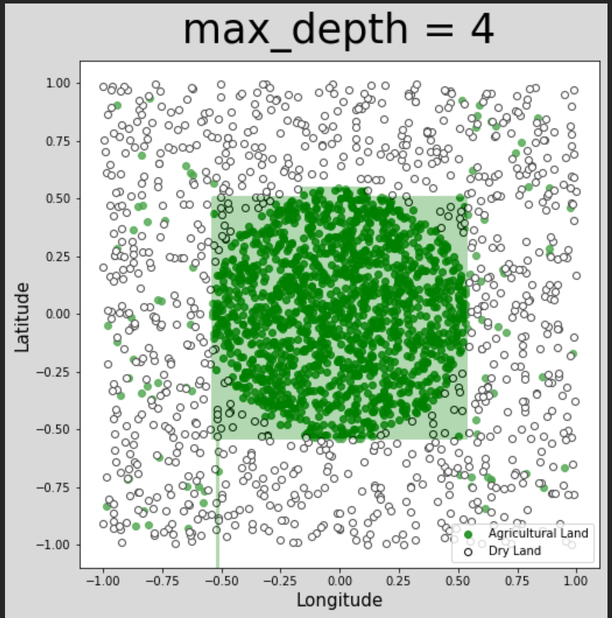
$$Gain(R) = \Delta(R) = \underset{\uparrow}{m}(R) - \frac{N_1}{N} m(R_1) - \frac{N_2}{N} m(R_2)$$

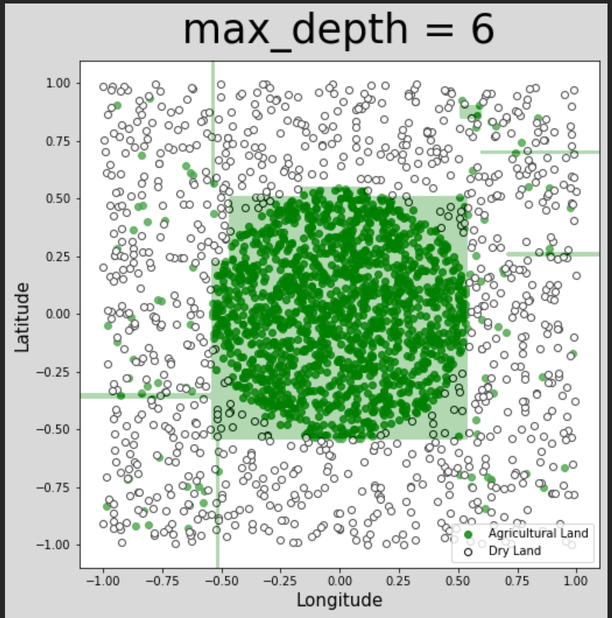
Classification Error/Gini Index/Entropy

Don't split if the gain is less than some pre-defined threshold (`min_impurity_decrease`).

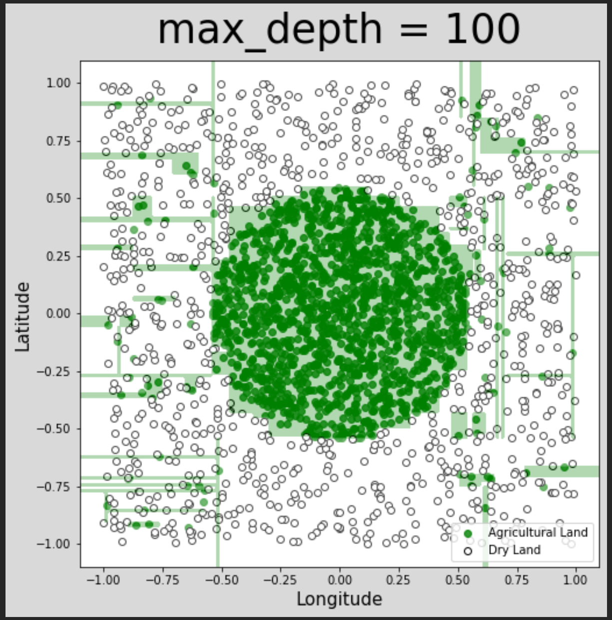
How do we decide what is the appropriate stopping condition or stopping method?

Variance vs Bias

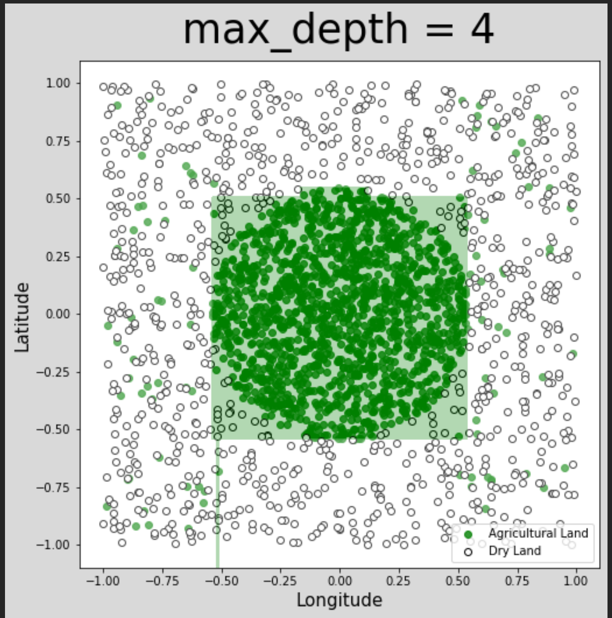


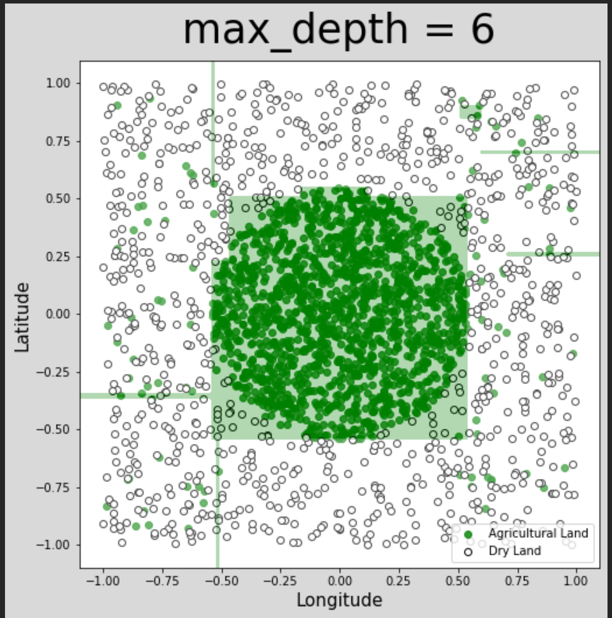


• • • •

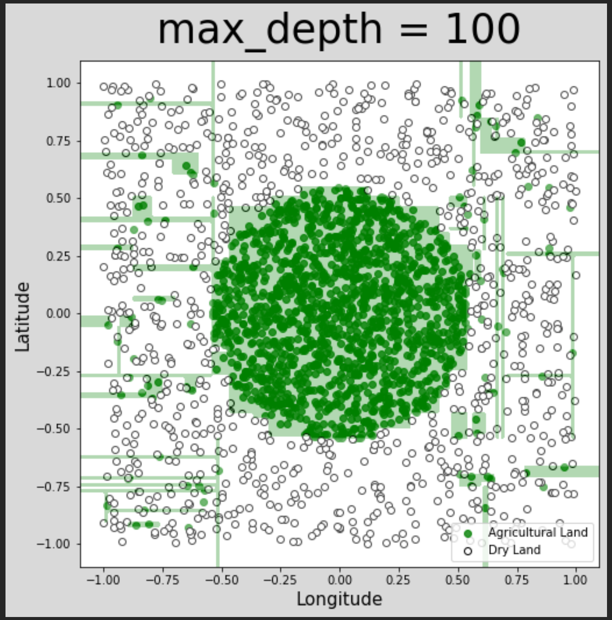


Variance vs Bias



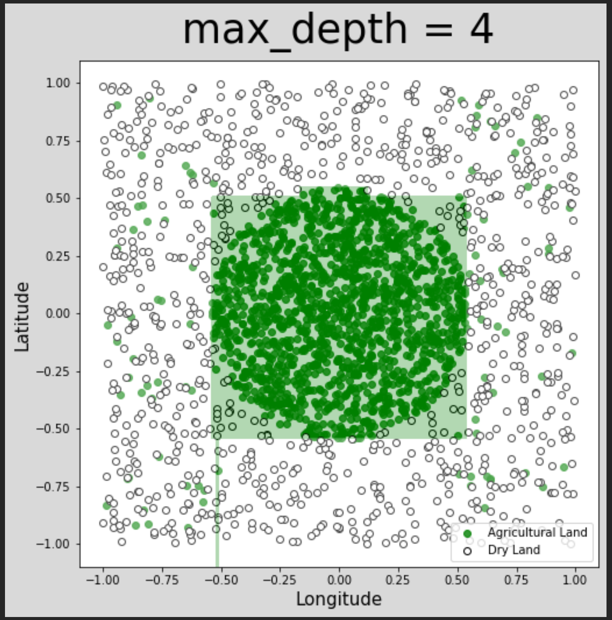


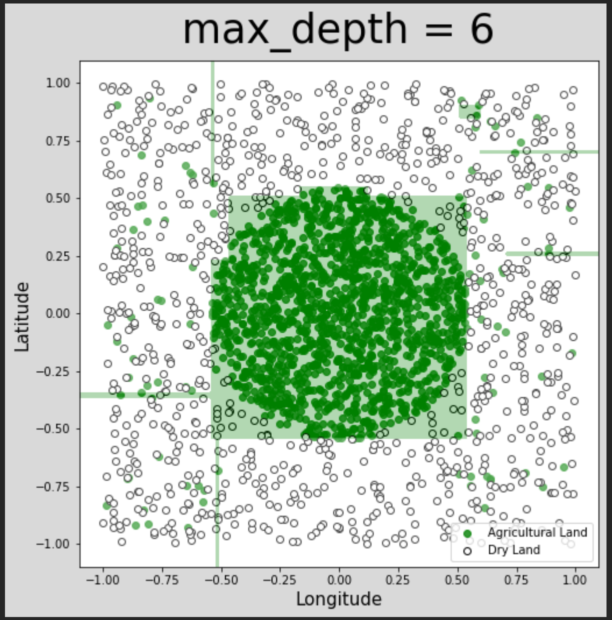
. . . .



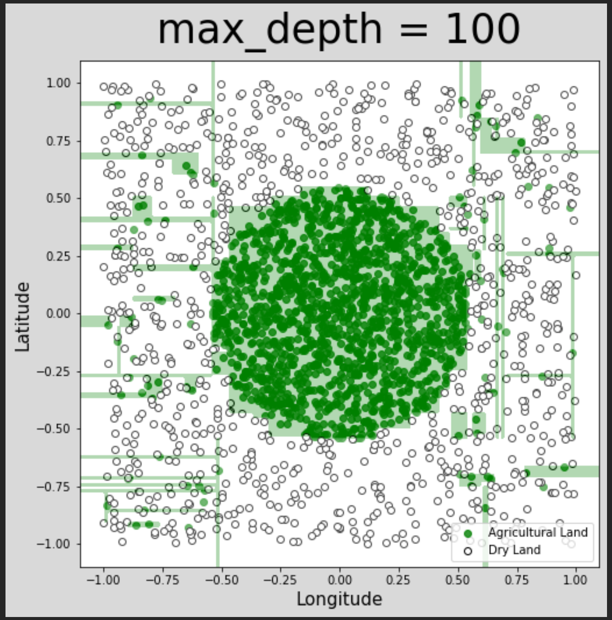
Bias decreases (can overfit)

Variance vs Bias





. . . .

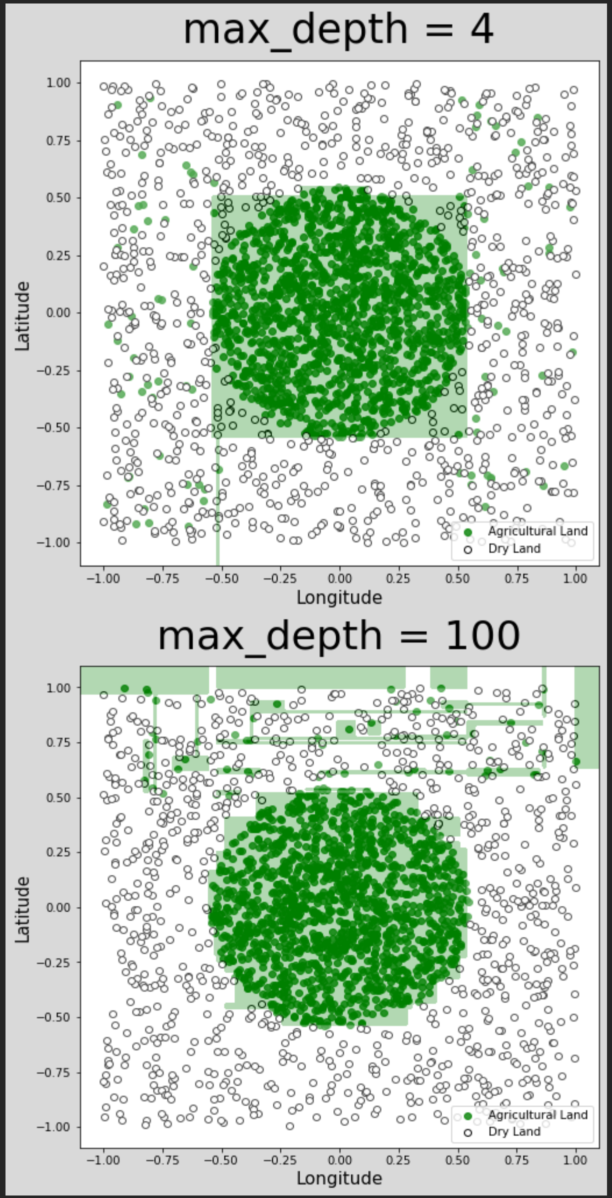


Bias decreases (can overfit)

Variance decreases (can underfit)

Complex trees are also harder to interpret and more computationally expensive to train.

Variance vs Bias

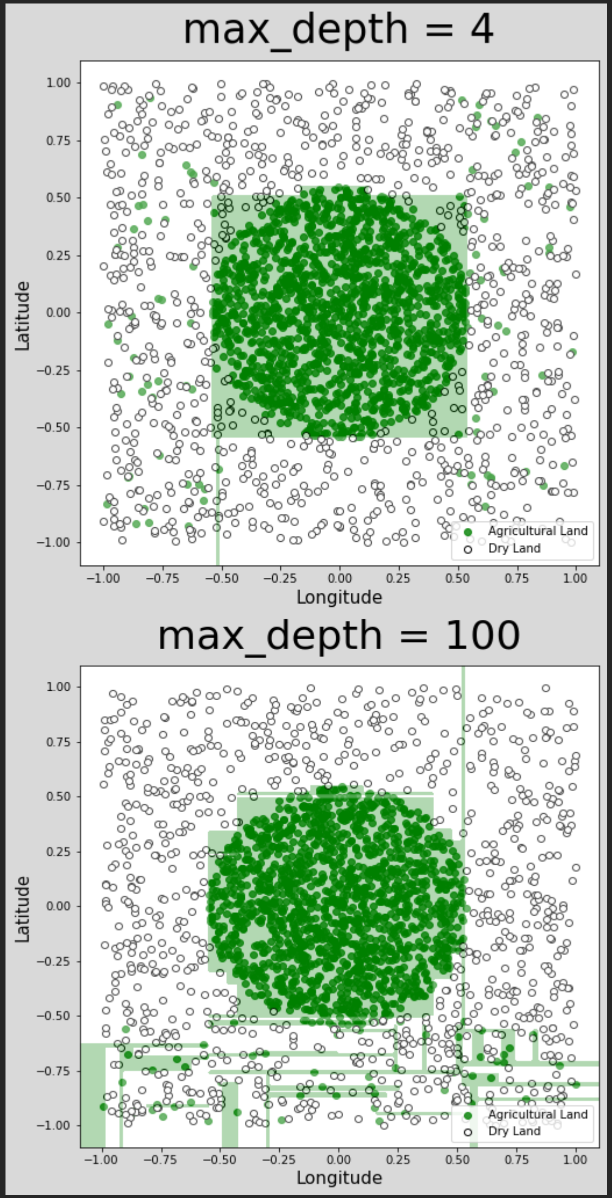


- **High Bias:** Trees of low depth are not a good fit for the training data - it's unable to capture the nonlinear boundary separating the two classes.

- **Low Variance:** Trees of low depth are robust to slight perturbations in the training data - the square carved out by the model is stable if you move the boundary points a bit.

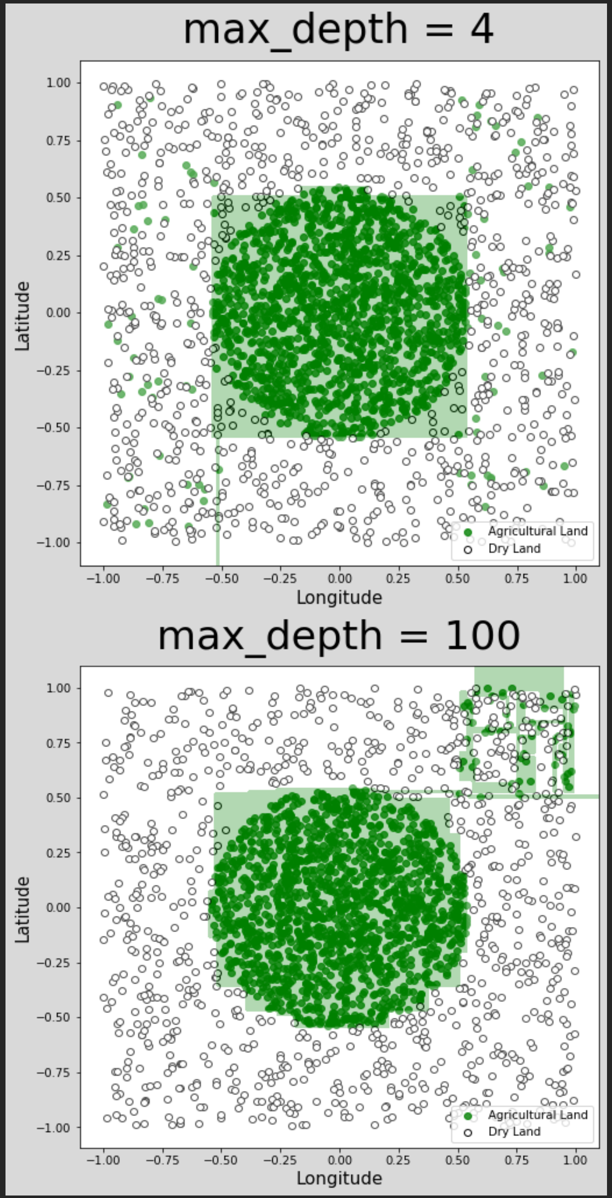
- **Low Bias:** With a high depth, we can obtain a model that correctly classifies all points on the boundary (by zig-zagging around each point).

Variance vs Bias



- **High Bias:** Trees of low depth are not a good fit for the training data - it's unable to capture the nonlinear boundary separating the two classes.
- **Low Variance:** Trees of low depth are robust to slight perturbations in the training data - the square carved out by the model is stable if you move the boundary points a bit.
- **Low Bias:** With a high depth, we can obtain a model that correctly classifies all points on the boundary (by zig-zagging around each point).

Variance vs Bias



- **High Bias:** Trees of low depth are not a good fit for the training data - it's unable to capture the nonlinear boundary separating the two classes.
- **Low Variance:** Trees of low depth are robust to slight perturbations in the training data - the square carved out by the model is stable if you move the boundary points a bit.
- **Low Bias:** With a high depth, we can obtain a model that correctly classifies all points on the boundary (by zig-zagging around each point).
- **High Variance:** Trees of high depth are sensitive to perturbations in the training data, especially to changes in the boundary points.

Stopping Conditions

`max_depth` `min_samples_leaf` `max_leaf_nodes` `min_impurity_decrease`

How can we determine the appropriate hyperparameters?

cross-validation

Game time

Consider the ROC plot below. A, B, C, and D represent 4 different binary classification models.

Arrange the model names such that they correspond to the sequence of statements below:

1. This model is the perfect/optimal classifier.
2. The model is the worst classifier.
3. The classifier is a chance based classifier, each class has an equal probability.
4. The model is a good classifier.

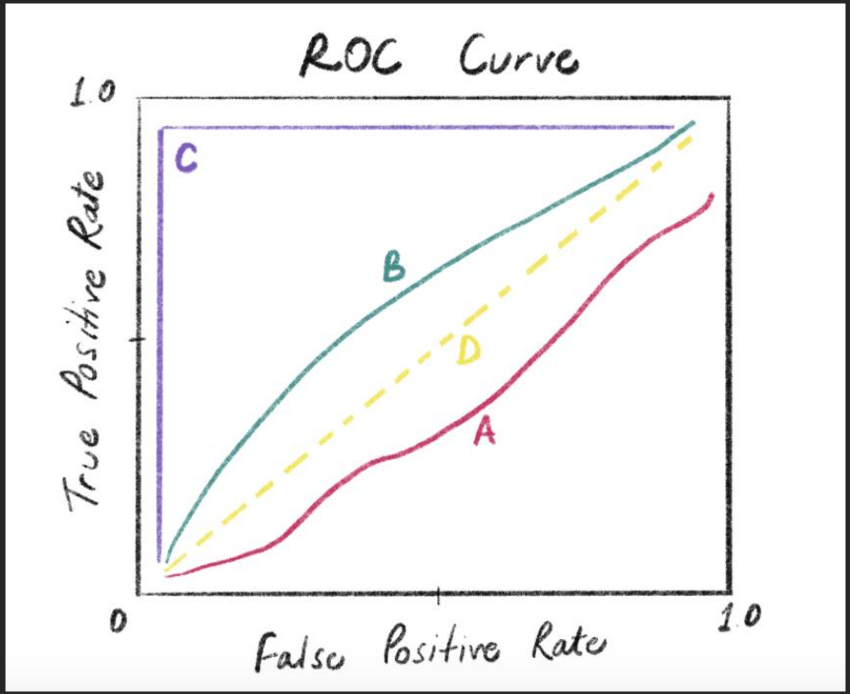
Options

A. 1-A, 2-B, 3-C, 4-D

B. 1-C, 2-A, 3-D, 4-B

C. 1-C, 2-D, 3-B, 4-A

D. 1-B, 2-A, 3-C, 4-D



Thank you