Decision Trees

Part A – Classification using trees

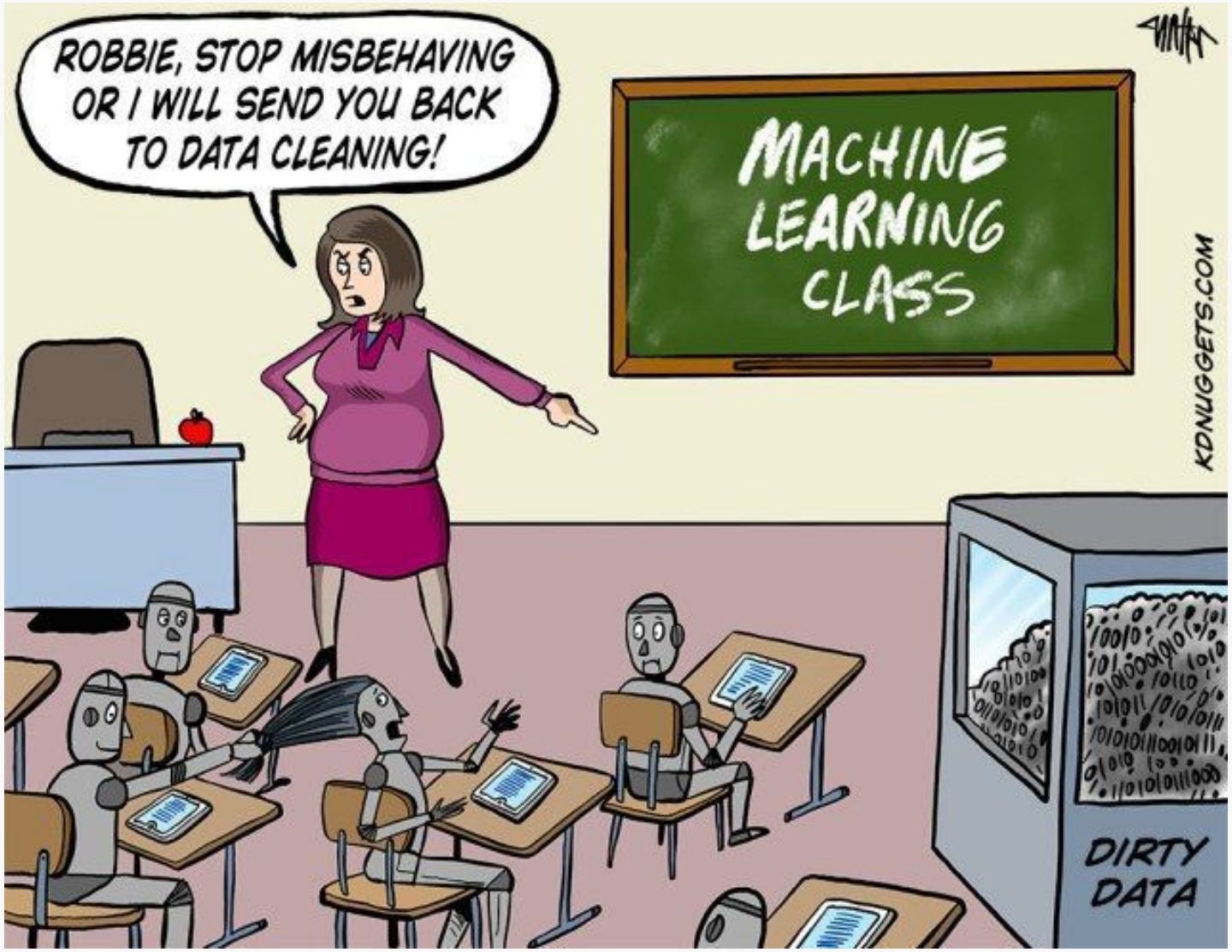
Pavlos Protopapas

Outline

- Motivation
- Decision Trees - Classification
- Splitting Criteria
- Stopping Conditions
- Decision Trees – Regression
- Numerical vs Categorical Attributes
- Pruning

Outline

- **Motivation**
- Decision Trees - Classification
- Splitting Criteria
- Stopping Conditions
- Decision Trees – Regression
- Numerical vs Categorical Attributes
- Pruning

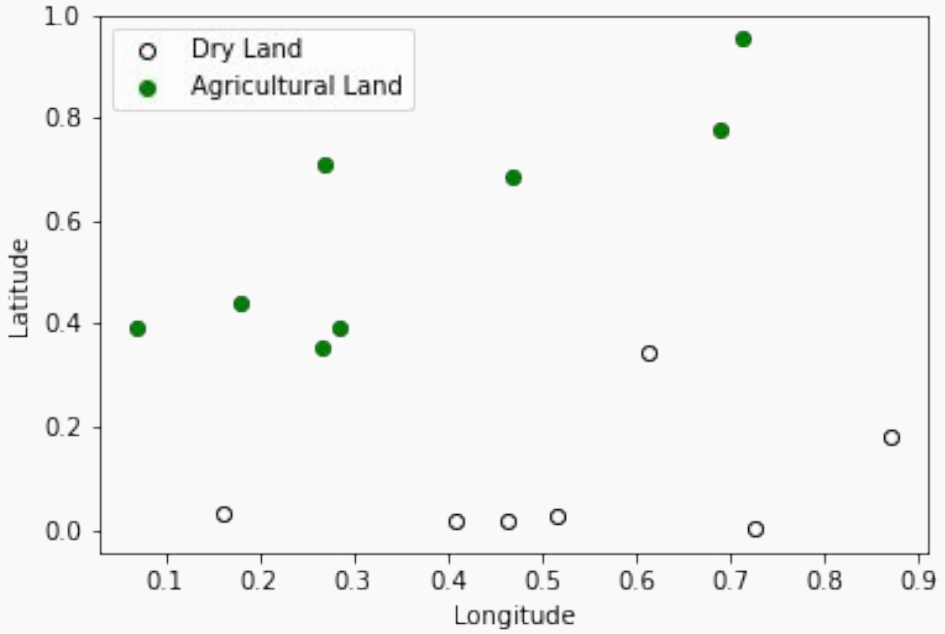

ROBBIE, STOP MISBEHAVING OR I WILL SEND YOU BACK TO DATA CLEANING!
MACHINE LEARNING CLASS
KDNUGGETS.COM
DIRTY DATA

Motivation

Recall that:

Logistic regression for building classification boundaries works best when:

- The classes are well-separated in the feature space
- The classification boundary has a nice geometry



Motivation

Recall that:

The decision boundary is defined where the probability of being in class 1 and class 0 are equal, i.e.

$$P(Y = 1) = 1 - P(Y = 1)$$

$$\Rightarrow P(Y = 1) = 0.5$$

which is equivalent to when the log-odds=0:

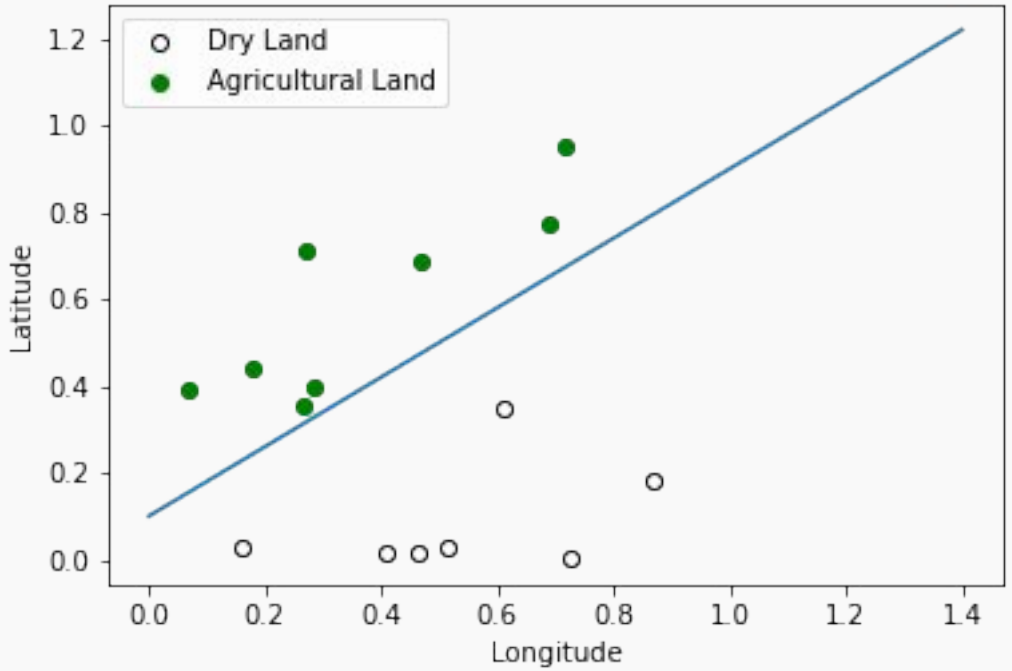
$$X\beta = 0$$

This equation defines a line or a hyperplane.

And it can be *generalized* with higher order polynomial terms.

Motivation

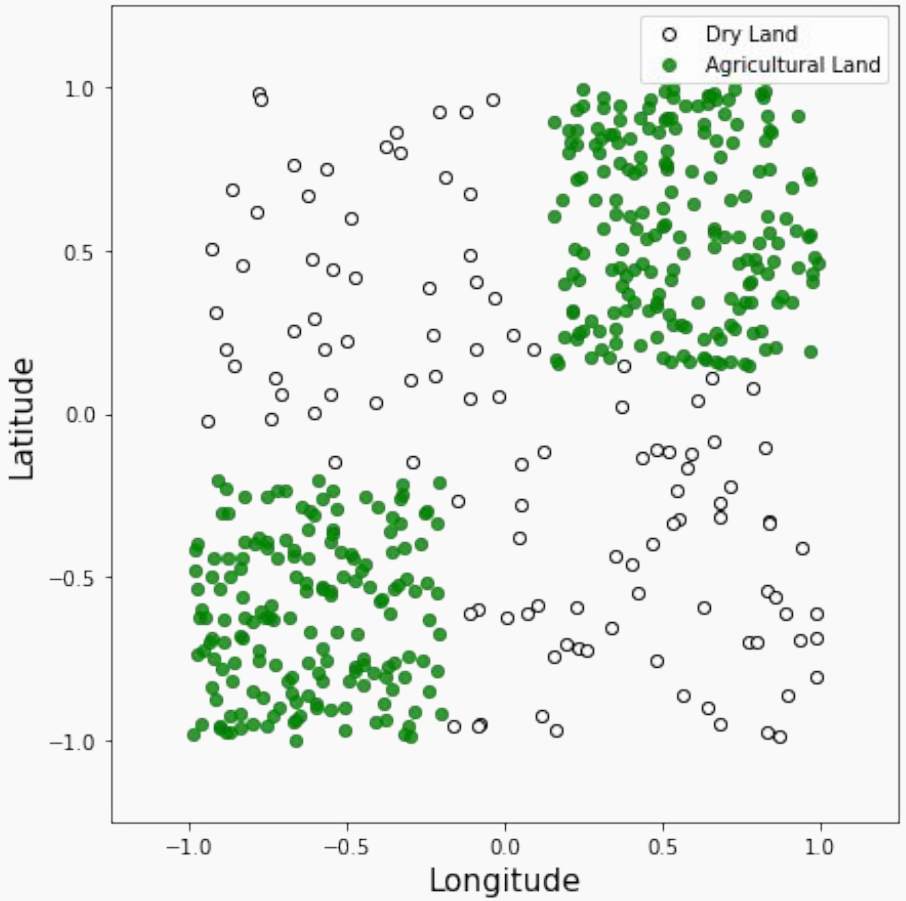
Question: Can you guess the equation that defines the decision boundary below?

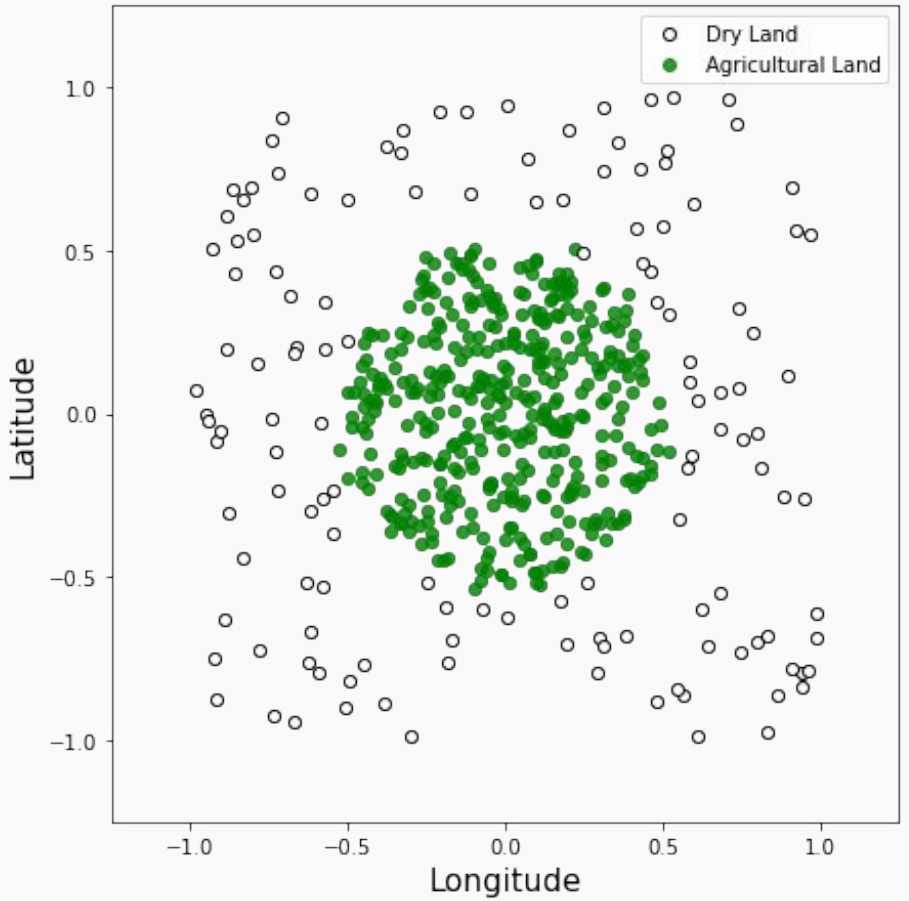


$$Latitude = 0.8\, Longitude + 0.1$$
$$or$$
$$-0.8 Longitude + Latitude - \ 0.1 = 0$$

Motivation

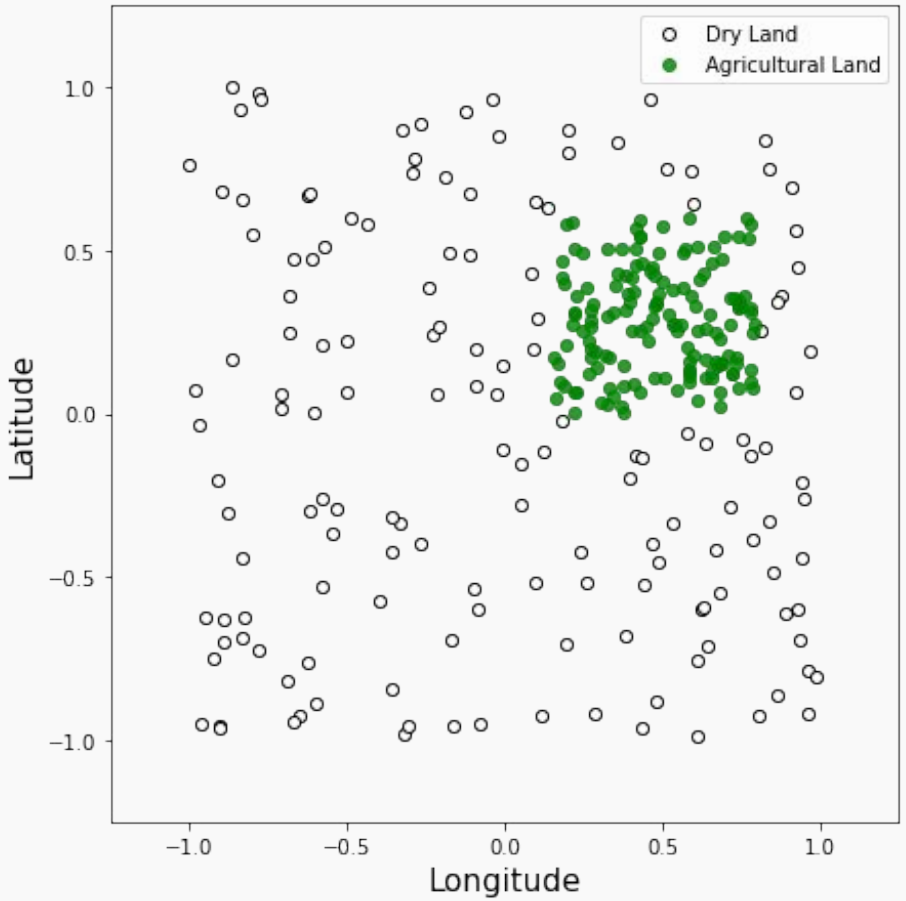
Question: How about these?

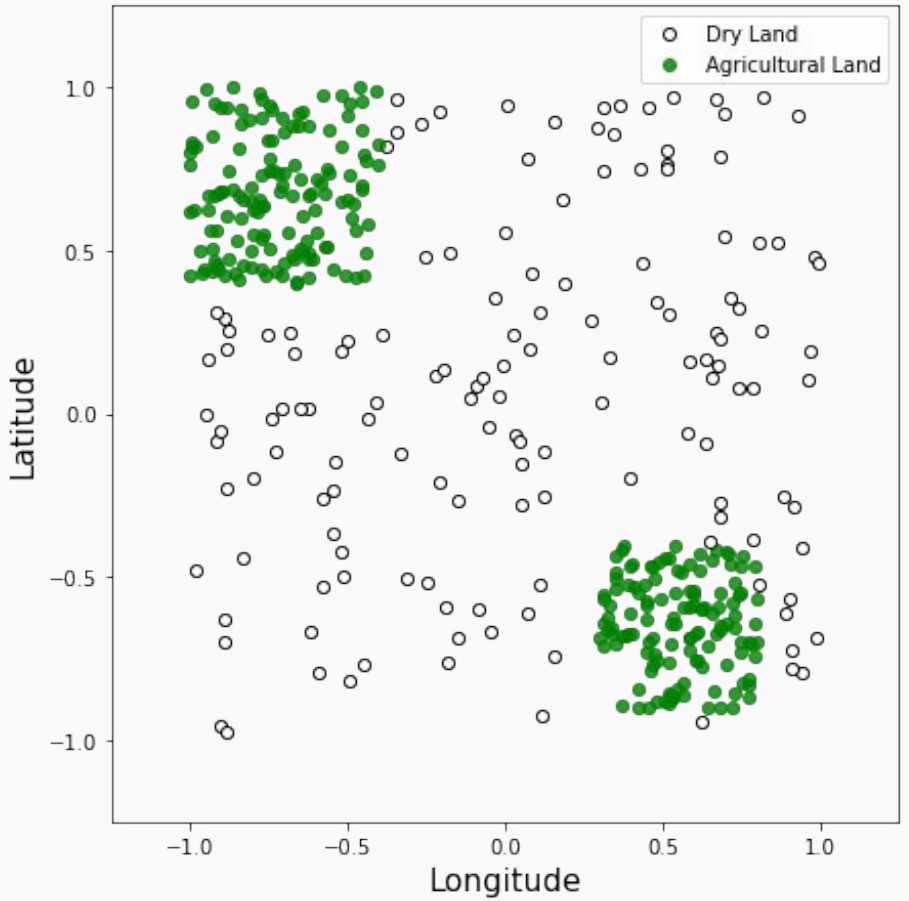




Motivation

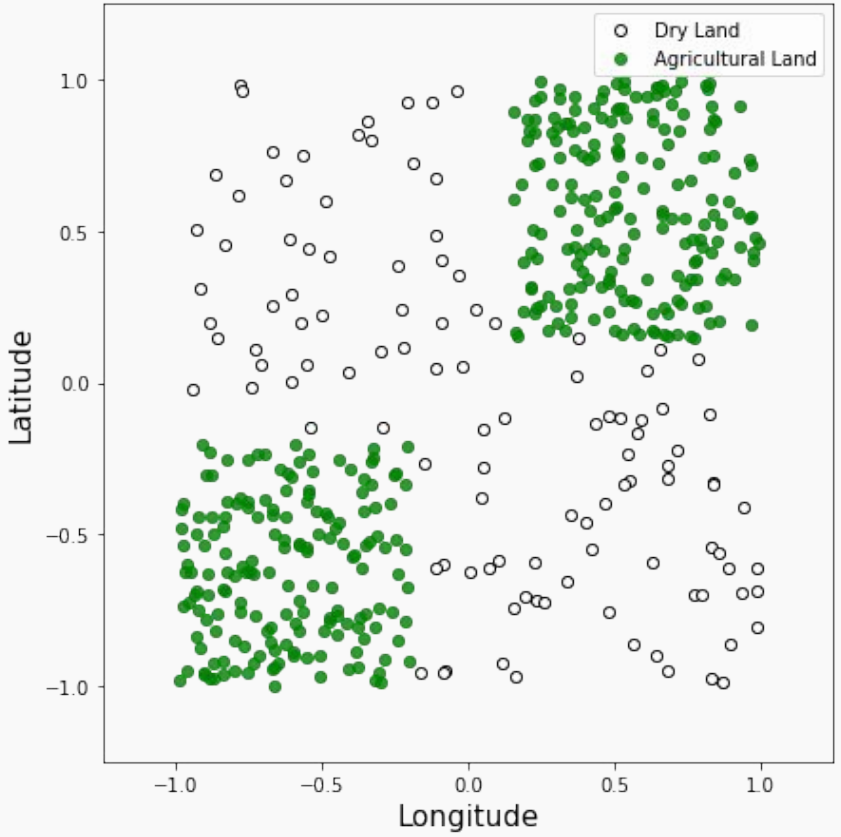
Question: Or these?





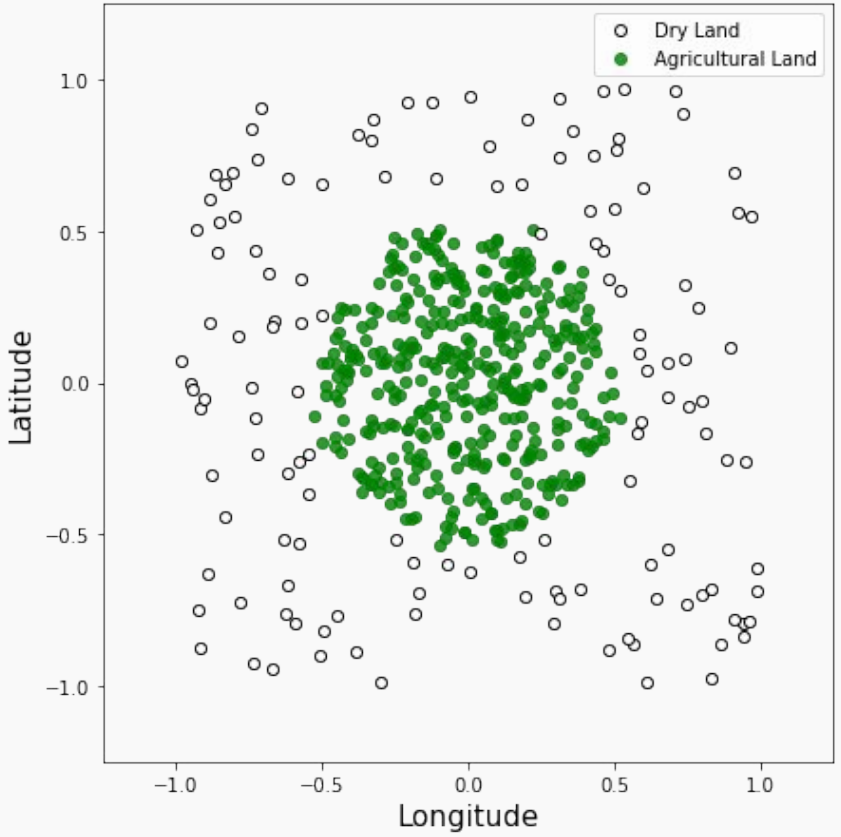
Motivation

Notice that in all the datasets the classes are still well-separated in the feature space, but ***the decision boundaries cannot easily be described by a single equation.***



Motivation

While logistic regression models with linear boundaries are intuitive to interpret by examining the impact of each predictor on the log-odds of a positive classification, it is **less straightforward to interpret nonlinear decision boundaries** in context.



$$x_1^2 + x_2^2 - 0.25 = 0$$

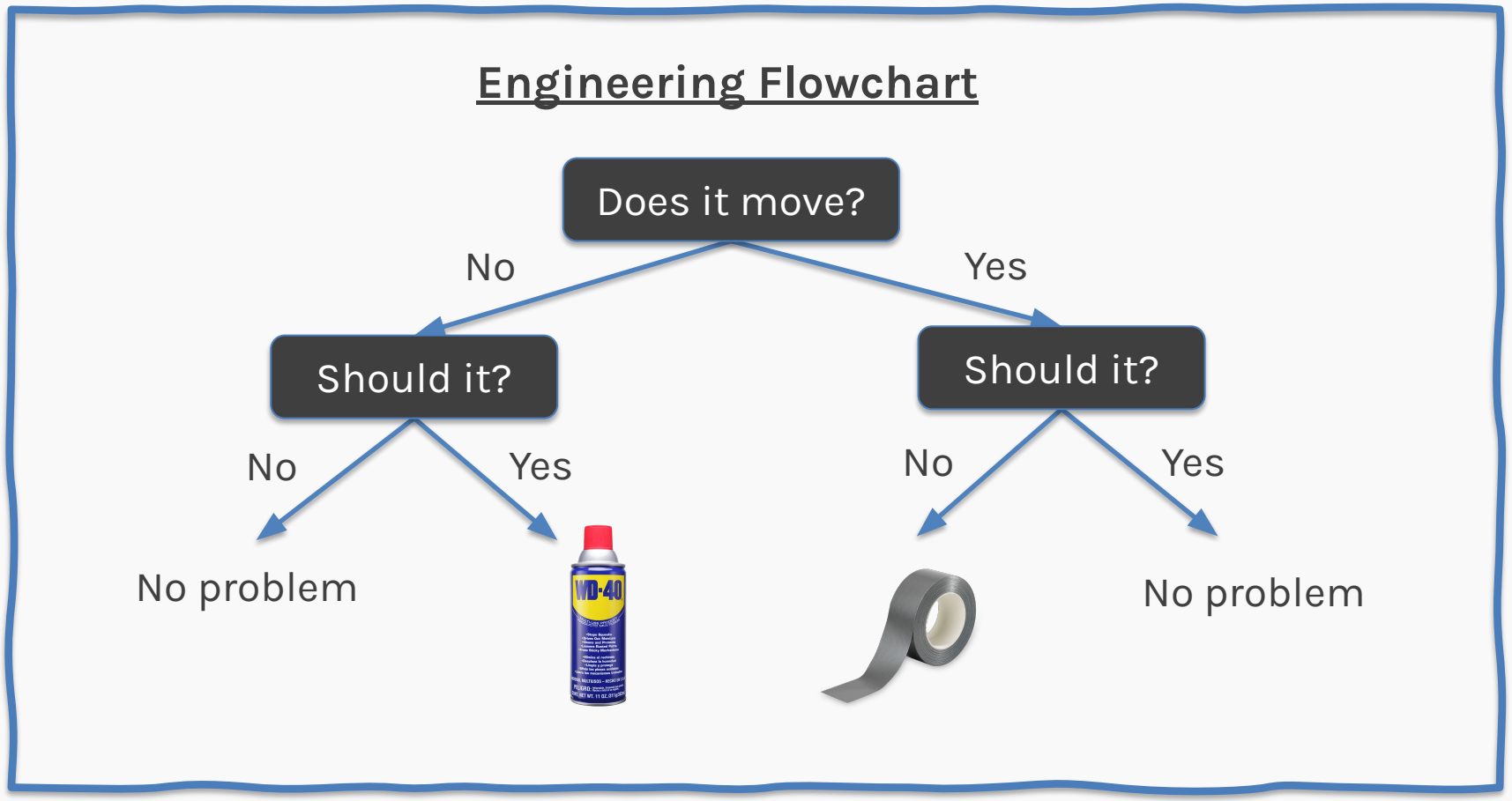
Motivation

It would be desirable to build models that:

1. Allow for ***complex decision boundaries***.
2. Are also ***easy to interpret***.

Interpretable Models

People in every walk of life have long been using interpretable models for differentiating between classes of objects and phenomena:



Interpretable Models

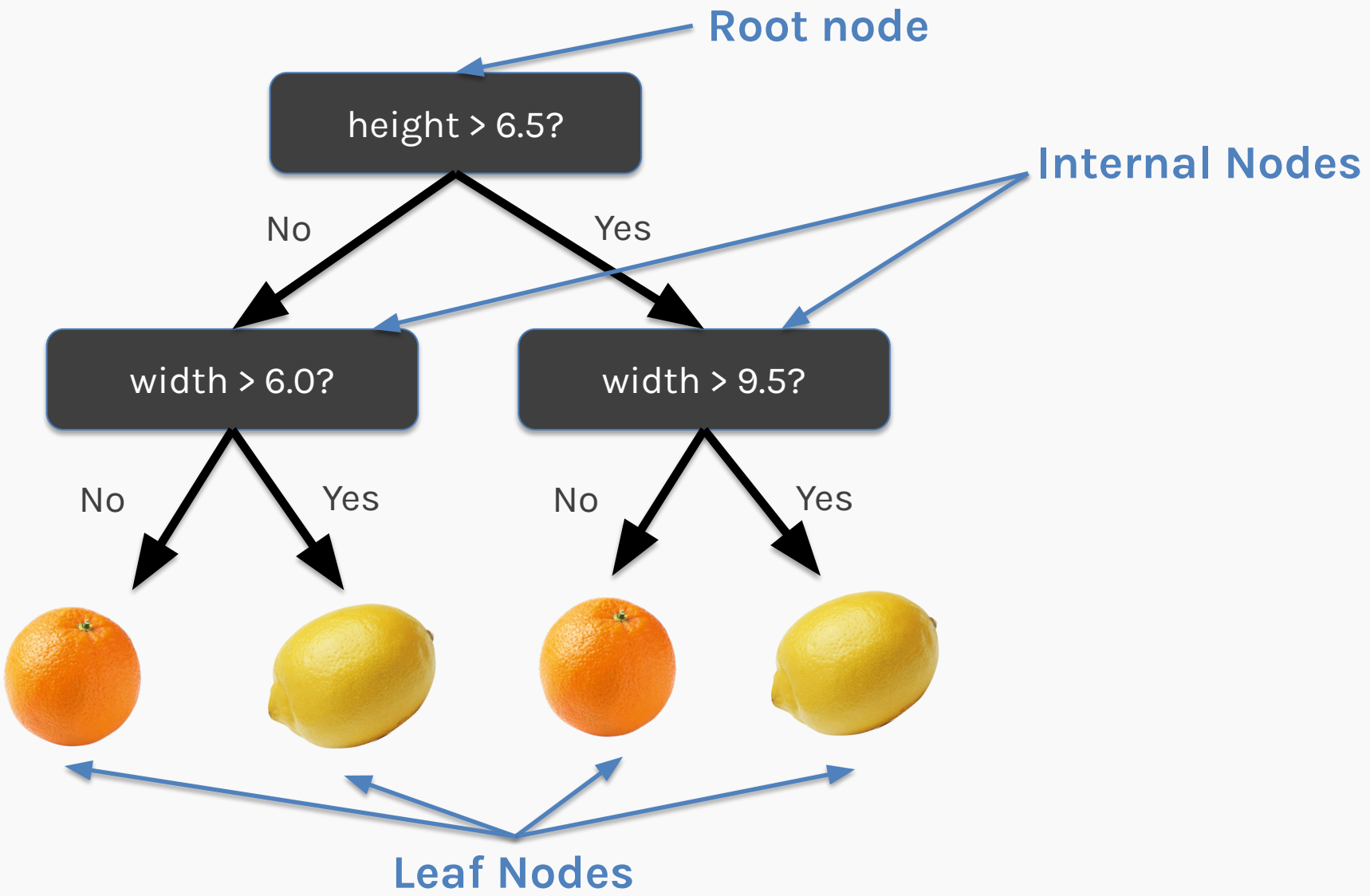
It turns out that the simple flow charts in our examples can be formulated as mathematical models for classification and these models have the properties we desire; they are:

1. Interpretable by humans
2. Have sufficiently complex decision boundaries
3. The decision boundaries are locally linear, each component of the decision boundary is simple to describe mathematically.

Outline

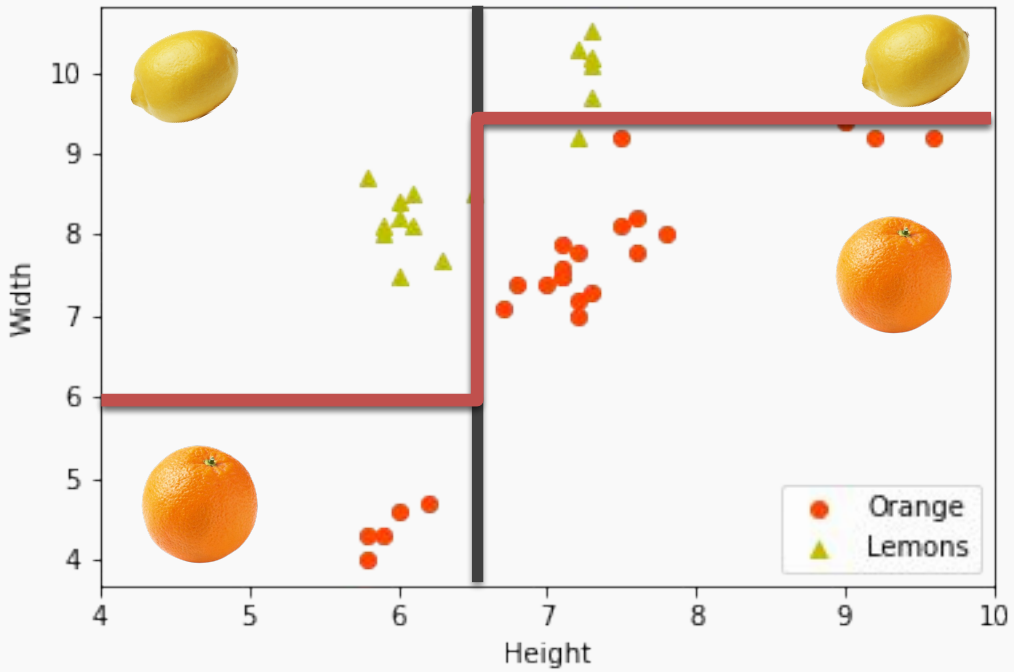
- Motivation
- **Decision Trees - Classification**
- Splitting Criteria
- Stopping Conditions
- Decision Trees – Regression
- Numerical vs Categorical Attributes
- Pruning

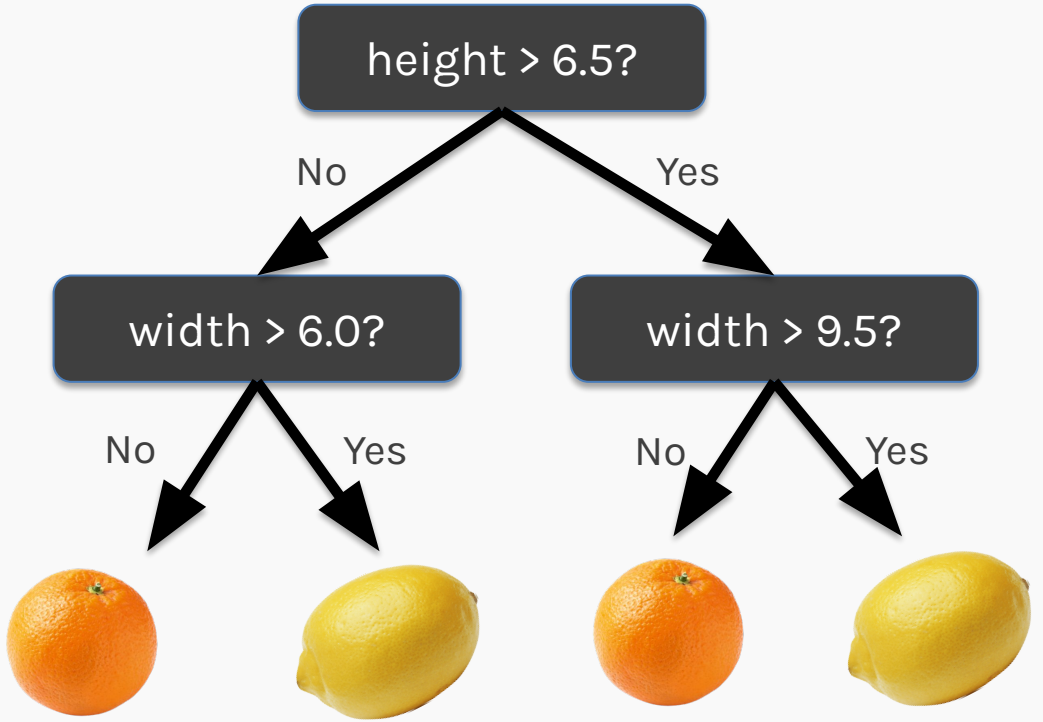
Example: Lemons and Oranges



The Geometry of Flow Charts

Every flow chart tree corresponds to a partition of the feature space by **axis aligned lines or (hyper) planes**. Conversely, every such partition can be written as a flow chart tree.





Learning the Model

Learning the ‘optimal’ decision tree for any given set of data is NP complete (intractable) for numerous simple definitions of ‘optimal’. Instead, we will use a greedy algorithm.

1. Start with an empty decision tree (undivided feature space)
2. Choose the ‘optimal’ predictor on which to split and choose the ‘optimal’ threshold value for splitting.
3. Recurse on each new node until *stopping condition* is met.
4. For the case of classification, predict each region to have a class label based on the largest class of the training points in that region (Bayes’ classifier).

Learning the Model

Learning the 'optimal' decision tree for any given set of data is NP complete (intractable) for numerous simple definitions of 'optimal'. Instead, we will seek a reasonably model using a greedy algorithm.

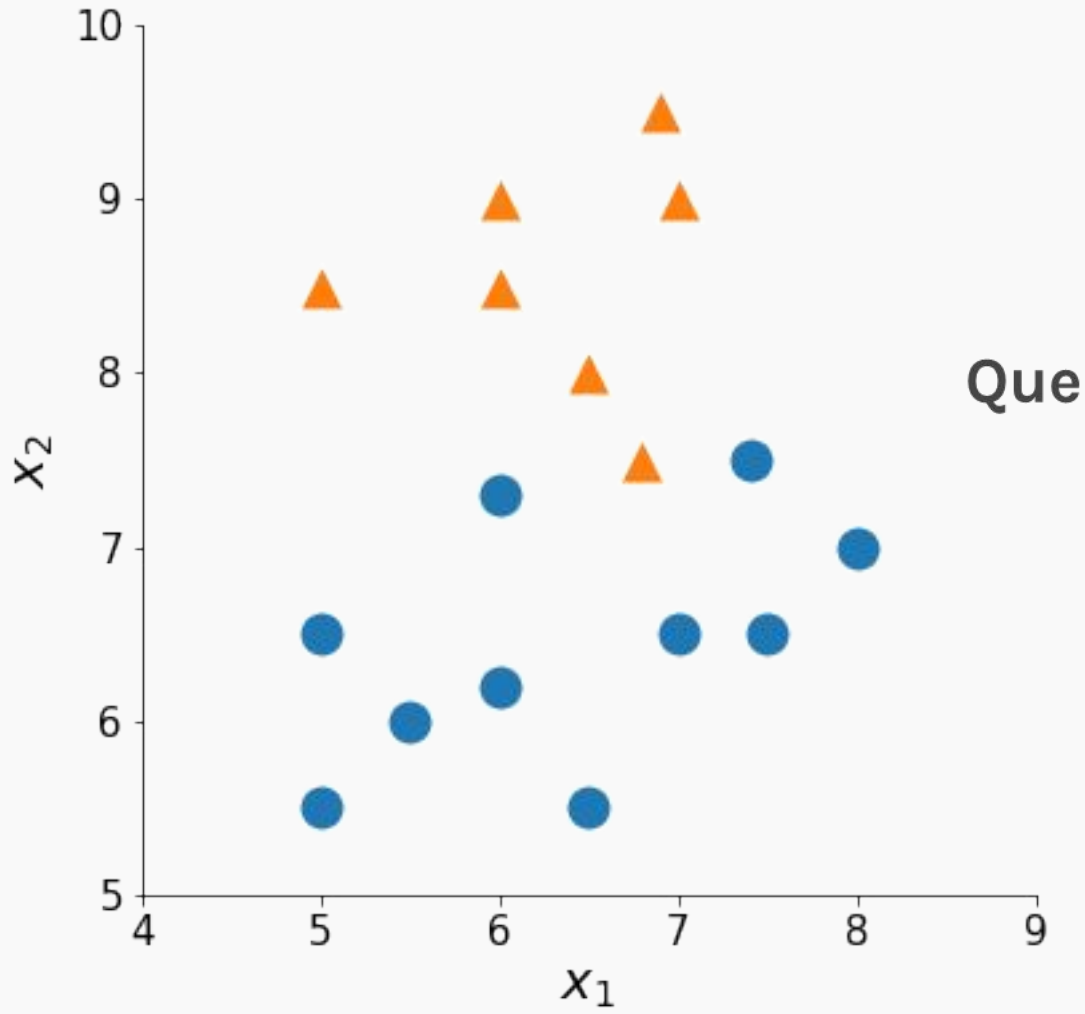
1. Start with an empty decision tree (undivided feature space)
2. **Choose the 'optimal' predictor on which to split and choose the 'optimal' threshold value for splitting.**

How do we choose the optimal variable and threshold?

Outline

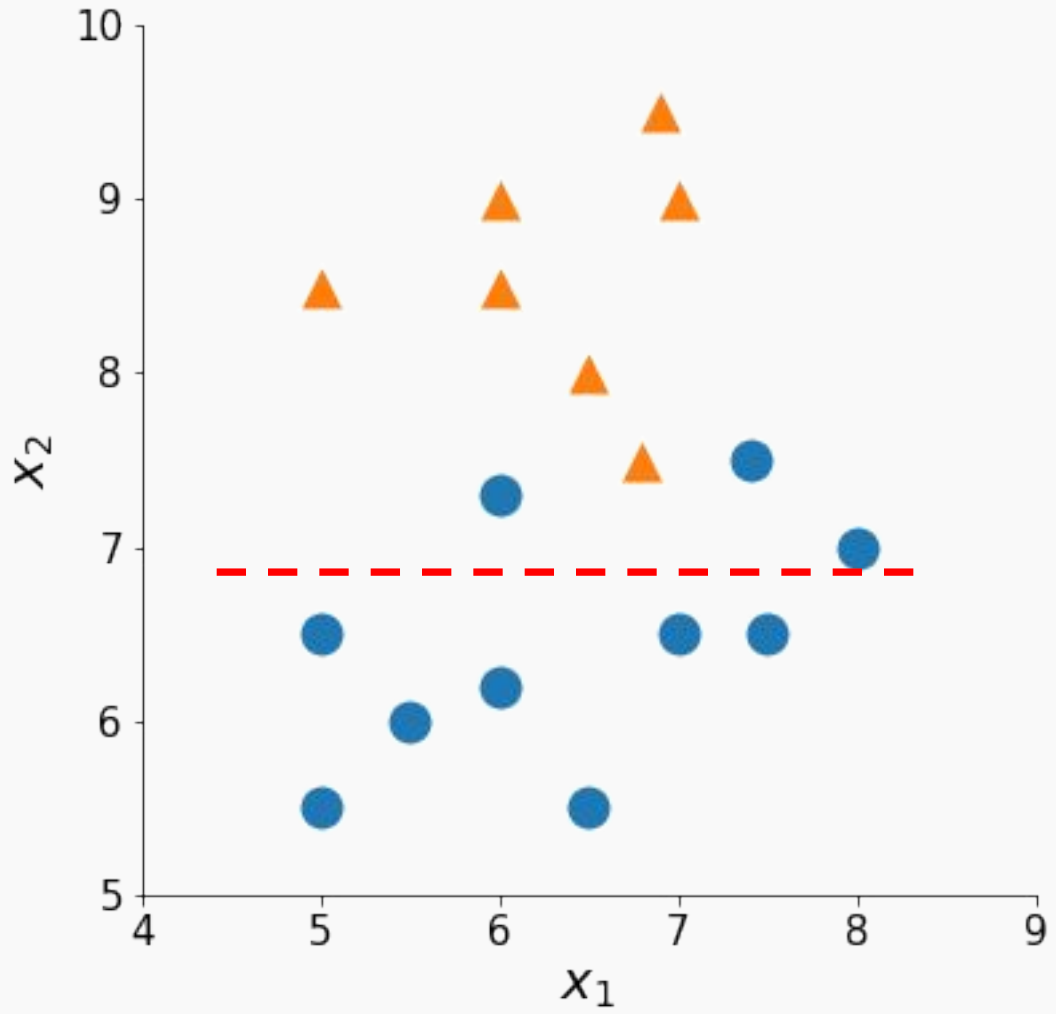
- Motivation
- Decision Trees - Classification
- **Splitting Criteria**
- Stopping Conditions
- Decision Trees – Regression
- Numerical vs Categorical Attributes
- Pruning

Splitting Criteria - Which gives a better split?



Question: Should it be split along x_1 or x_2?

Splitting Criteria - Which gives a better split?



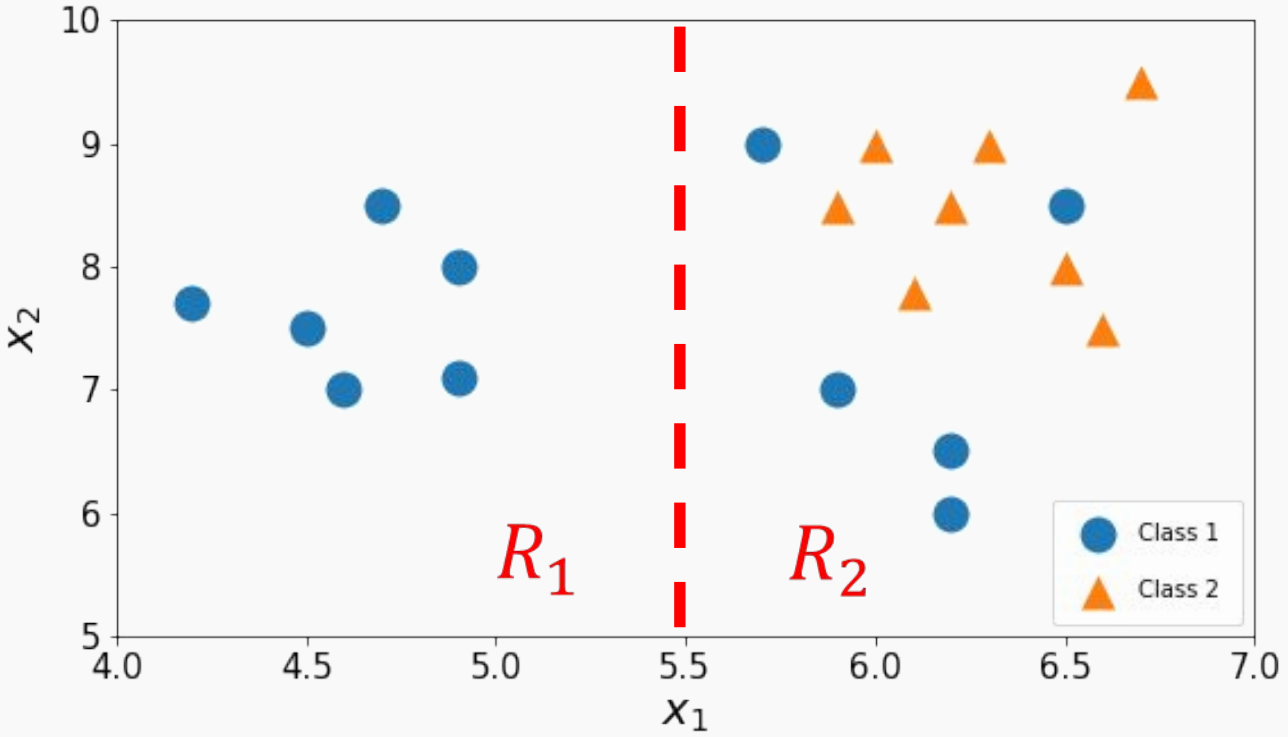
Question: Where should we split?

Optimality of Splitting

While there is no 'correct' way to define an optimal split, there are some common sensical guidelines for every splitting criterion:

- The regions in the feature space should grow progressively **purer** with the number of splits. That is, we should see that each region 'specializes' towards a single class.
- The fitness metric of a split should take a **differentiable form** (making optimization possible).
- We shouldn't end up with **empty regions** - regions containing no training points.

Classification Error



Consider region R_2:

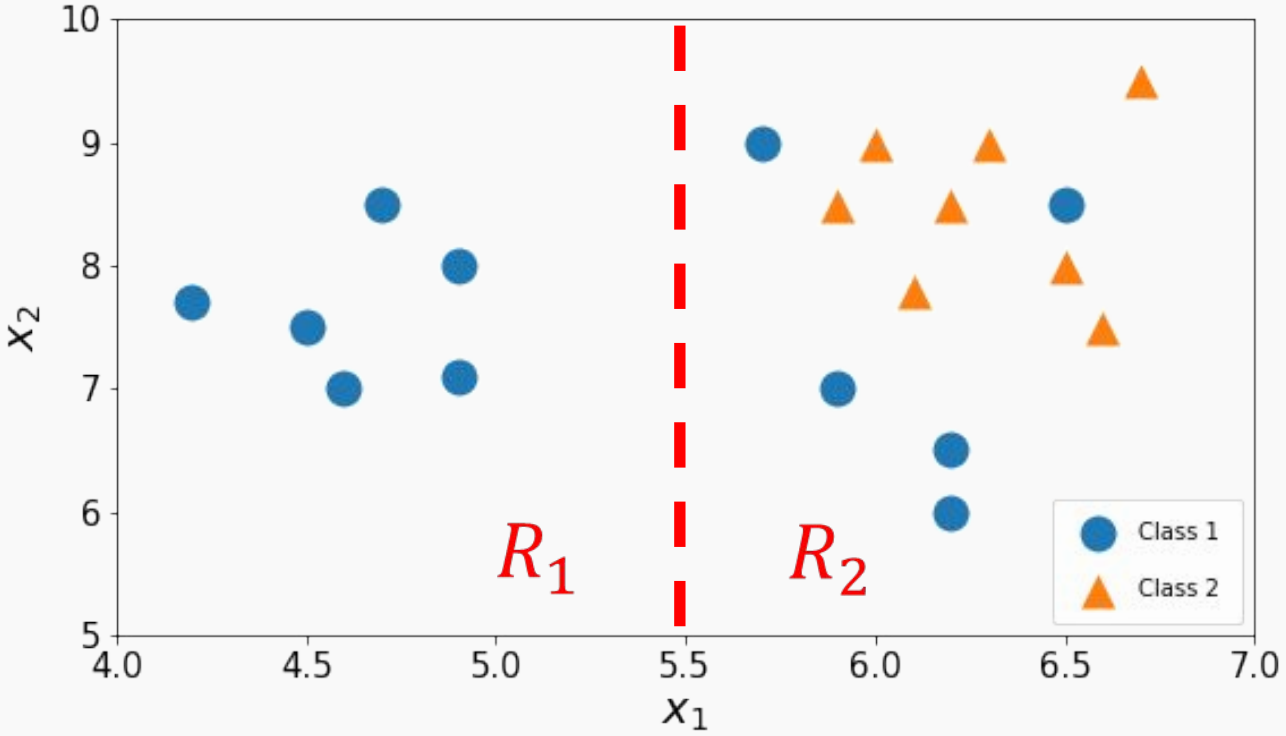
$$\text{Classification error} = \frac{error}{total}$$

$$= \frac{Number\ of\ minority\ data\ points}{Total\ number\ of\ data\ points}$$

$$= \frac{Number\ of\ \bullet}{Total\ number\ of\ data\ points}$$

$$= 1 - \frac{Number\ of\ majority\ data\ points}{Total\ number\ of\ data\ points}$$

Classification Error



Consider region R_2:

$$= 1 - \frac{Number\ of\ majority\ data\ points}{Total\ number\ of\ data\ points}$$

$$= 1 - \Psi(\blacktriangle | R_2)$$

$$= 1 - \max_k(\Psi(k|R_r))$$

where $\Psi(k|R_r)$ is the proportion of training points in R_2 that are labeled class k.

Classification Error

In general:

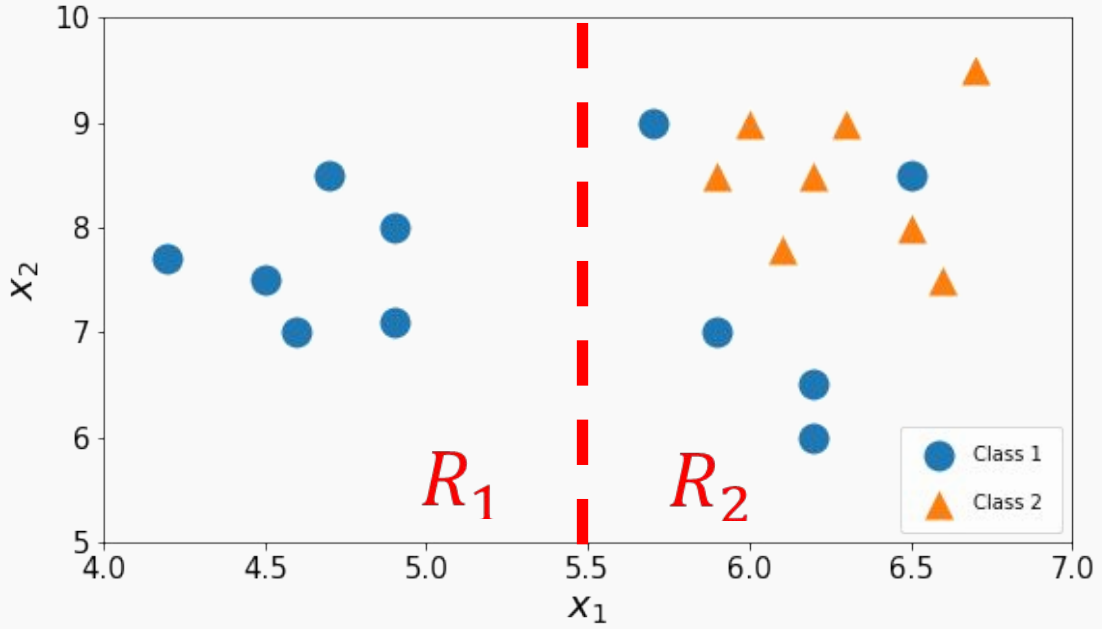
Assume we have P **predictors** and K **classes**. Suppose we select the p^{th} predictor and split a region along the **threshold** $t_p \in \mathbb{R}$.

We can assess the quality of this split by measuring the **classification error** made by each newly created region by calculating:

$$Error\ (R_r | \ p, t_p) = 1\ - \max_k(\Psi(k|R_r))$$

where $\Psi(k|R_r)$ is the proportion of training points in R_r that are labeled class k.

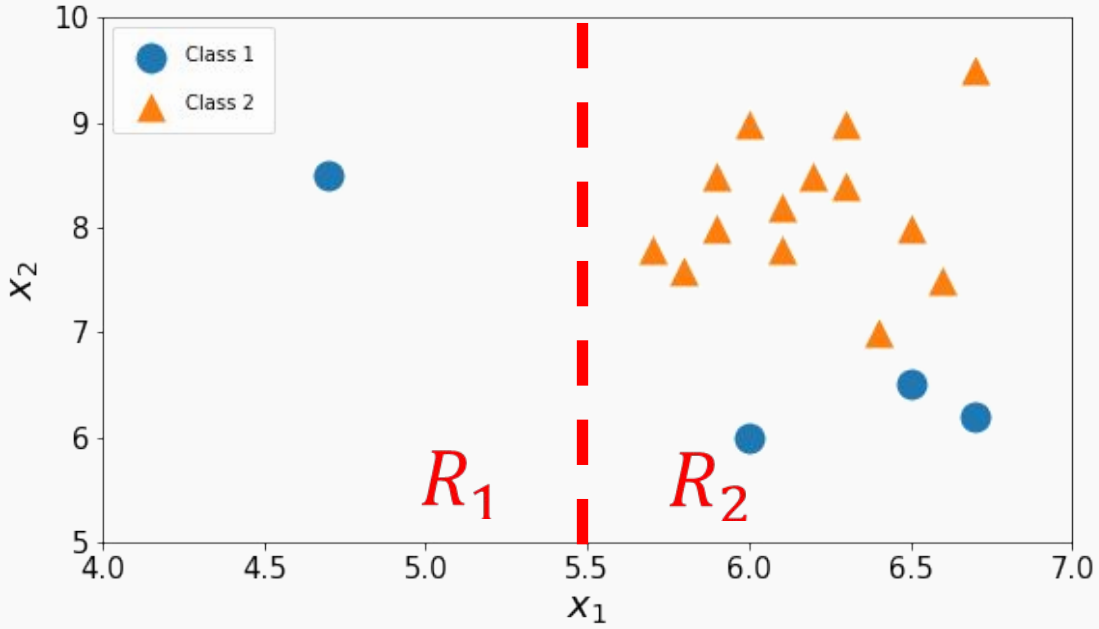
Example: Classification Error



6	0	$1 - max\left(\frac{6}{6}, \frac{0}{6}\right) = 0$
5	8	$1 - max\left(\frac{5}{13}, \frac{8}{13}\right) = \frac{5}{13}$

Classification Error

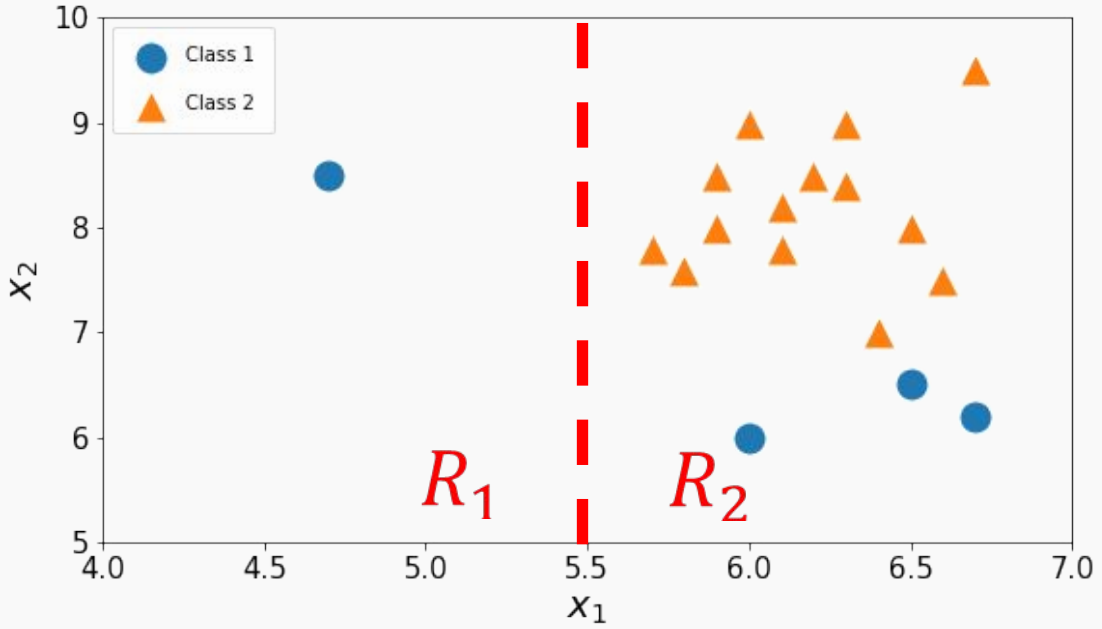
Now calculate the error for the following split:



1	0	$1 - max\left(\frac{1}{1}, \frac{0}{1}\right) = 0$
3	14	$1 - max\left(\frac{3}{17}, \frac{14}{17}\right) = \frac{3}{17}$

R_1 has a smaller error that R_2.
Does that mean this is a good split?

Classification Error



We need to take the **weighted average** over both regions so the number of points in each region is taken into consideration:

$$\min_{p,\, t_p} \left[\frac{N_1}{N} Error\left(R_1 \middle| p\,, t_p\right) + \frac{N_2}{N} Error\left(R_2 \middle| p\,, t_p\right) \right]$$

where N_r is the number of training points inside region R_r.

Gini Index

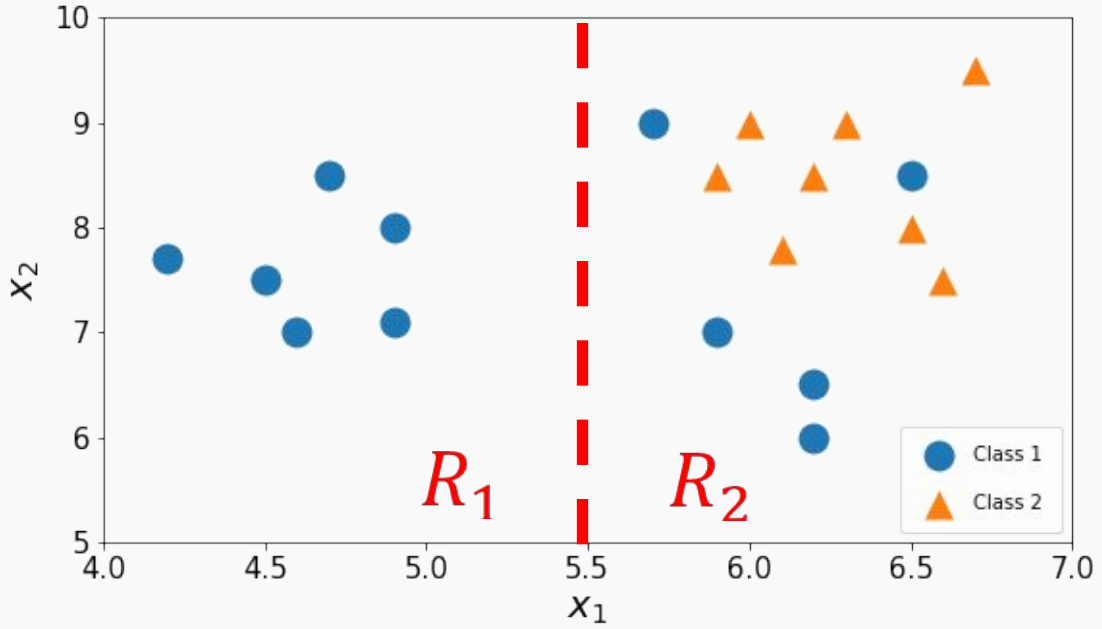
Assume we have P **predictors** and K **classes**. Suppose we select the p^{th} predictor and split a region along the **threshold** $t_p \in \mathbb{R}$.

We can assess the quality of this split by measuring the **Gini Index** made by each newly created region by calculating:

$$Gini\ (R_r|\ p, t_p) = 1\ - \sum_k \Psi(k|R_r)^2$$

Question: What is the effect of squaring the proportions of each class? What is the effect of summing the squared proportions of classes within each region?

Gini Index



	●	▲	
	6	0	$1-\left[\left(\frac{6}{6}\right)^2+\left(\frac{0}{6}\right)^2\right]=0$
	5	8	$1-\left[\left(\frac{5}{13}\right)^2+\left(\frac{8}{13}\right)^2\right]=\frac{80}{169}$

Gini Index

We can now try to find the predictor p and the threshold t_p that minimizes the **weighted average Gini Index** over the two regions:

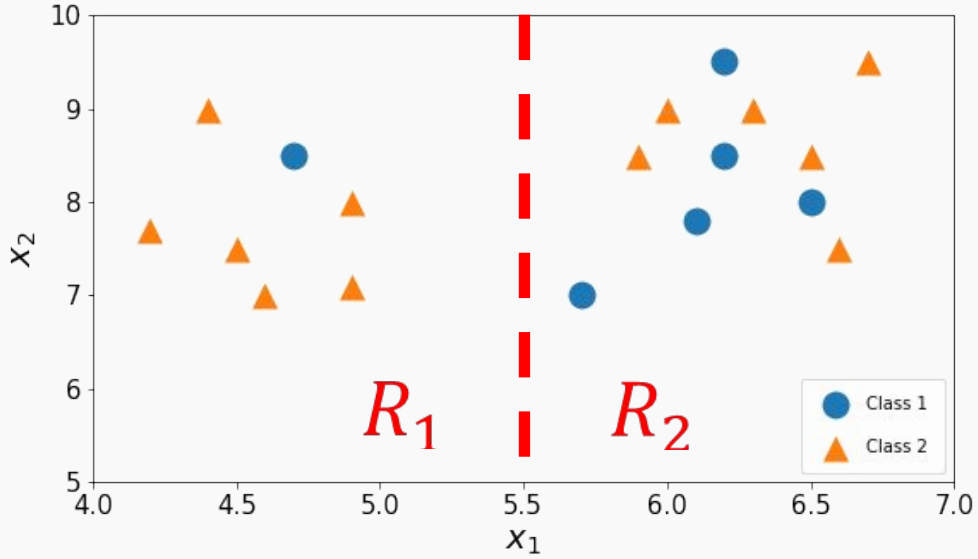
$$\min_{p,t_p}\left[\frac{N_1}{N}Gini\left(R_1\middle|p\,,t_p\right)+\frac{N_2}{N}Gini\left(R_2\middle|p\,,t_p\right)\right]$$

where N_r is the number of training points inside region R_r.

Information Theory

The last metric for evaluating the quality of a split is motivated by metrics of uncertainty in information theory.

Question: In the below plot, which region is 'purer'?



While both regions are impure, R_1 is clearly sending a **stronger 'signal'** for class 2 than R_2.

Information Theory

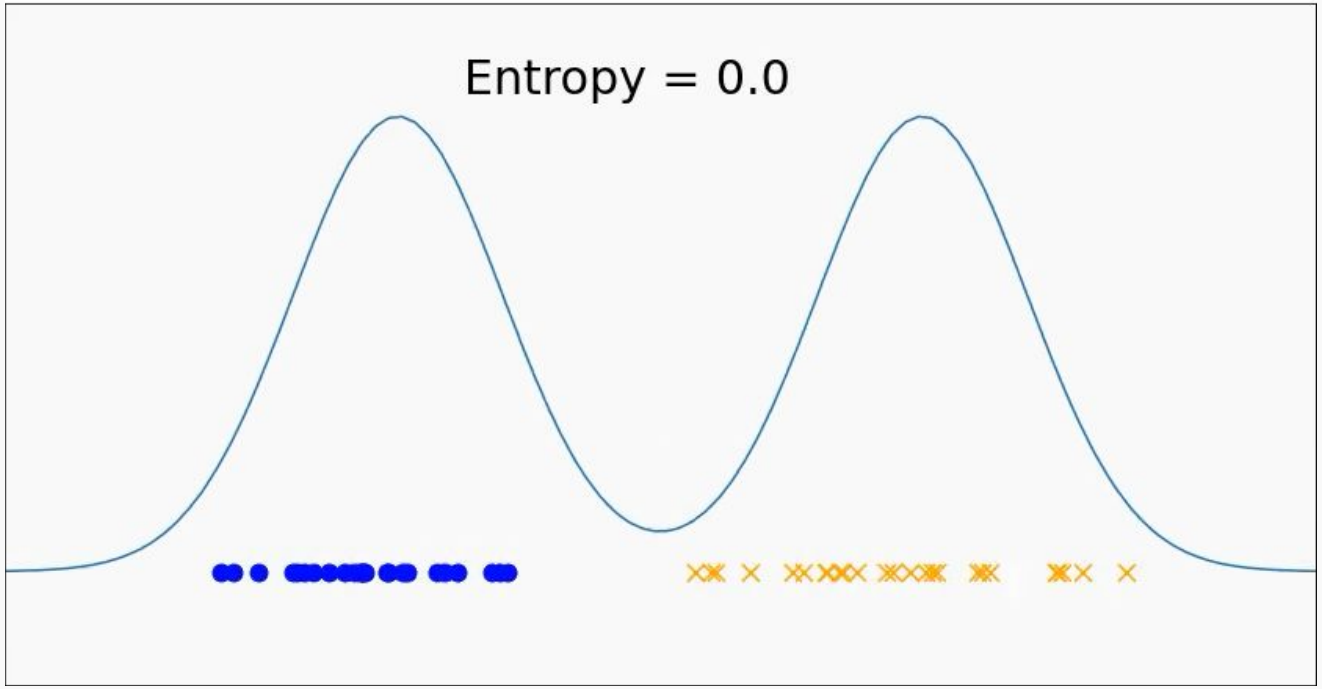
One way to quantify the strength of a signal in a particular region is to analyze the **distribution of classes** within the region. We compute the **entropy** of this distribution.

For a random variable with a discrete distribution, the entropy is computed by:

$$H(x) = -\sum_{x \epsilon X} \Psi(x) \log_2 \Psi(x)$$

Information Theory

$$H(x) = -\sum_{x \epsilon X} \Psi(x) \log_2 \Psi(x)$$



Entropy

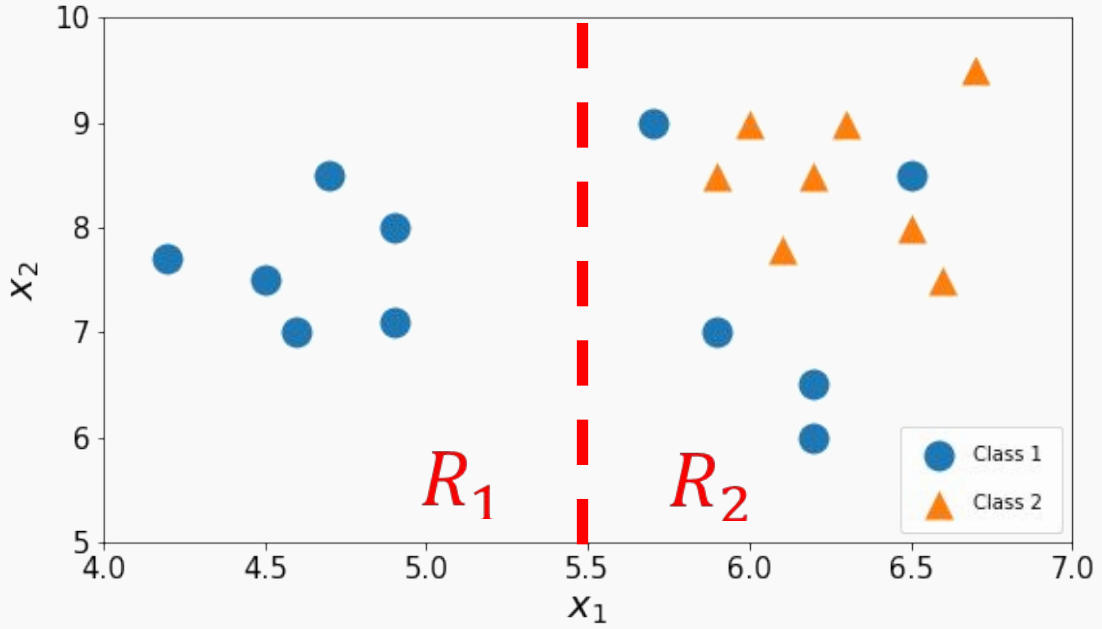
Assume we have P **predictors** and K **classes**. Suppose we select the p^{th} predictor and split a region along the **threshold** $t_p \in \mathbb{R}$.

We can assess the quality of this split by measuring the **entropy of the class distribution** in each newly created region by calculating:

$$Entropy\,(R_r|\,p,\,t_p\,) = -\sum_k \Psi(k|R_r)\log_2\Psi(k|R_r)$$

Note: We are actually computing the conditional entropy of the distribution of training points amongst the K classes given that the point is in region r.

Entropy - Example



	●	▲	
	6	0	$-\left[\frac{6}{6}\log_2\frac{6}{6}+\frac{0}{6}\log_2\frac{0}{6}\right]=0$
	5	8	$\left[\frac{5}{13}\log_2\frac{5}{13}+\frac{8}{13}\log_2\frac{8}{13}\right]\approx 1.3$

Entropy

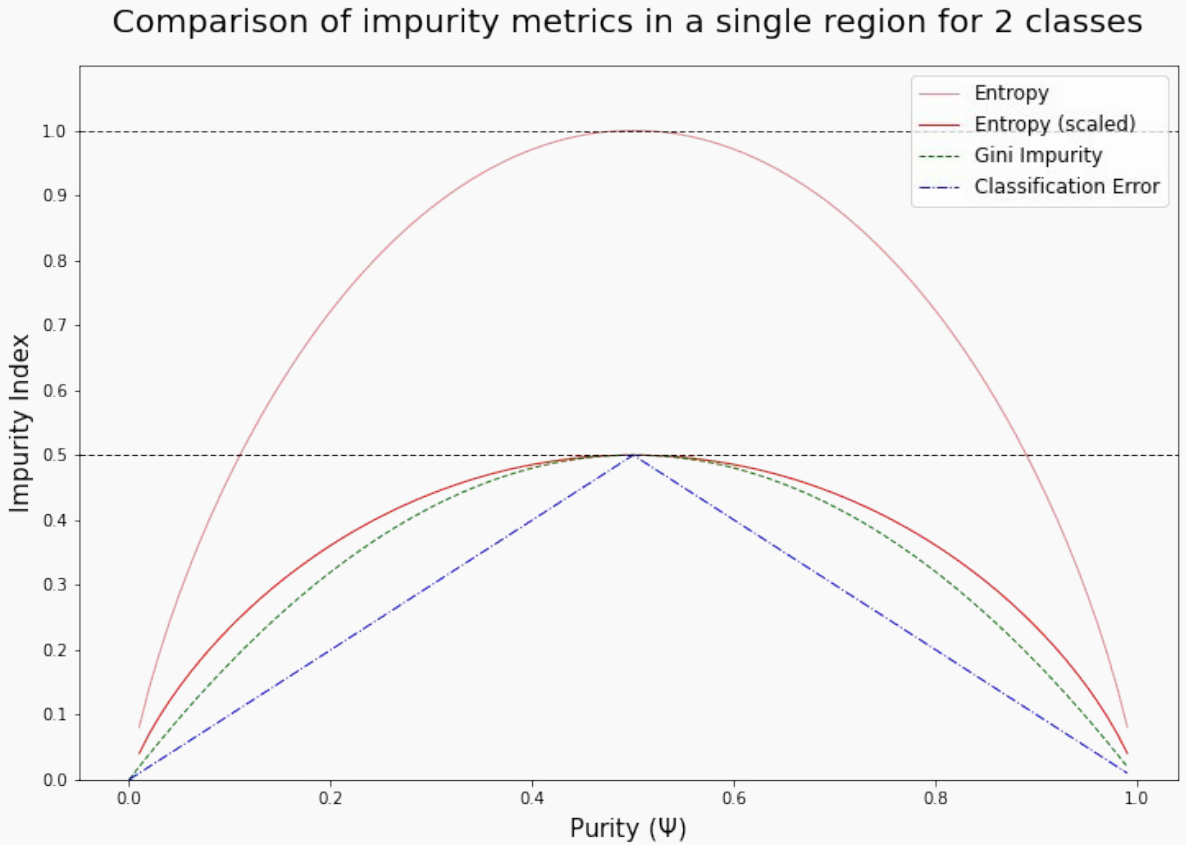
We can now try to find the predictor p and the threshold t_p that minimizes the **weighted average entropy** over the two regions:

$$\min_{p,t_p}\left[\frac{N_1}{N}Entropy\left(R_1\middle|p\ ,t_p\right)+\frac{N_2}{N}Entropy\left(R_2\middle|p\ ,t_p\right)\right]$$

where N_r is the number of training points inside region R_r.

Comparison of Criteria

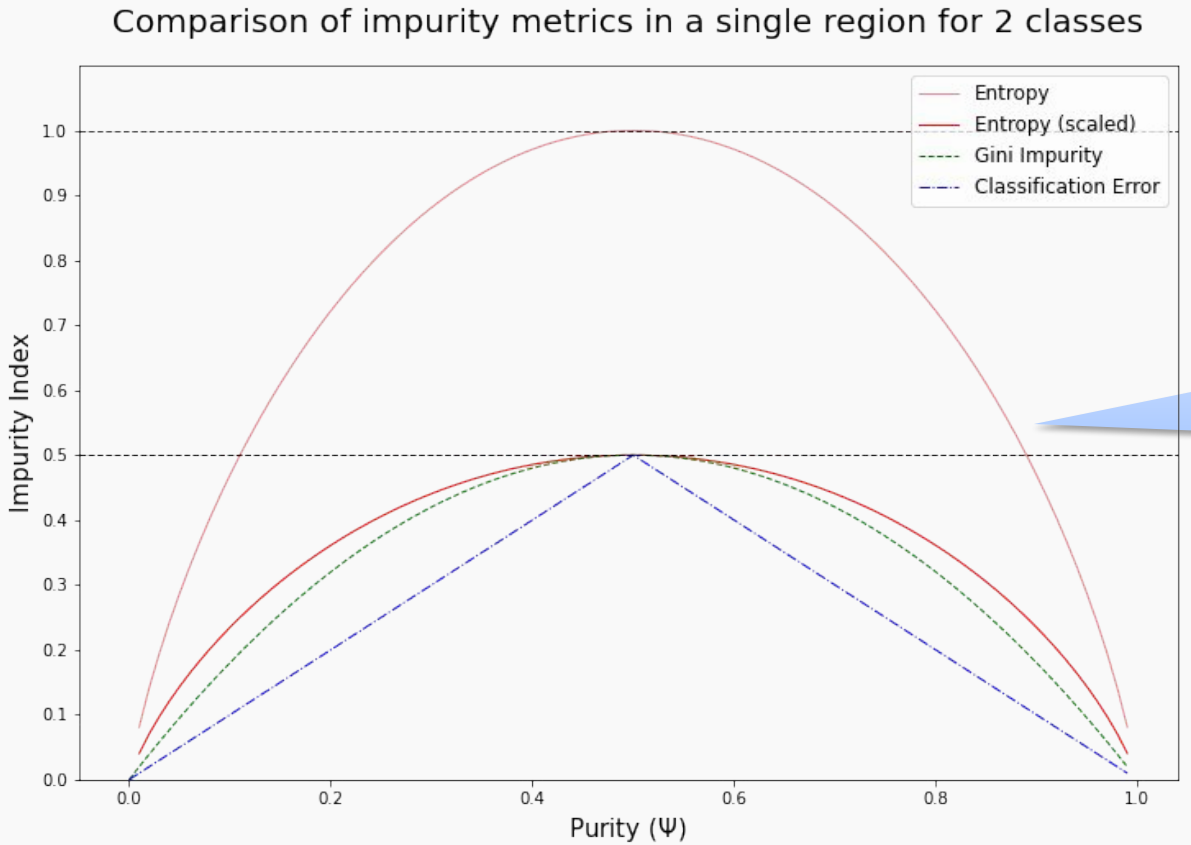
Question: Which of the three criteria fits our guideline the best?



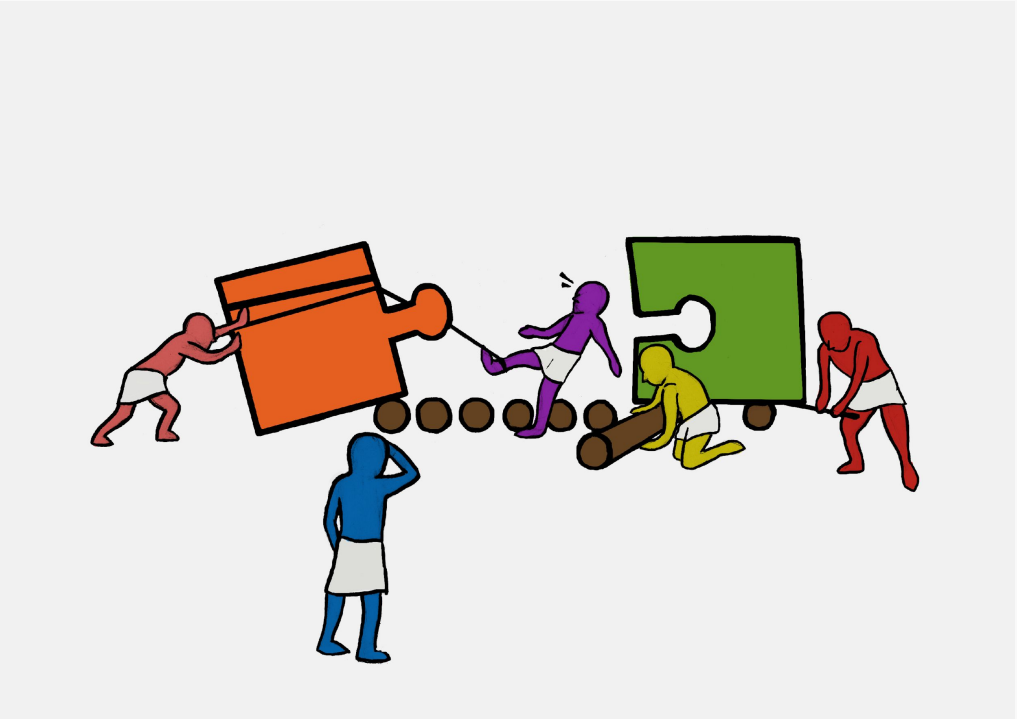
Note: Sklearn uses the unscaled entropy as a splitting criteria

Comparison of Criteria

Question: Which of the three criteria fits our guideline the best?

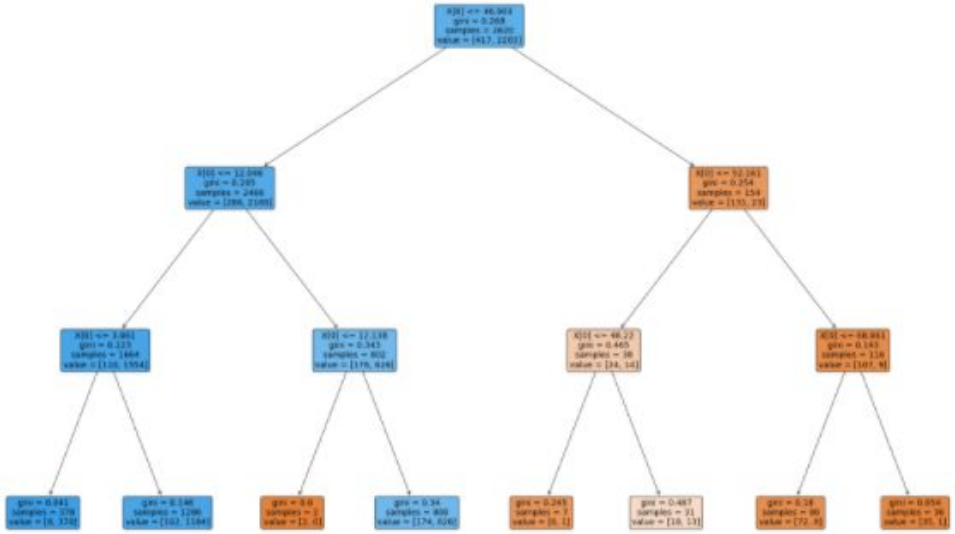


Entropy **penalizes impurity the most**



👨‍🏫 Exercise: Visualizing a Decision Tree

The aim of this exercise is to visualize the decision tree that is created when performing Decision Tree Classification or Regression. The tree will look similar to the one given below.



Dataset Description:

We are trying to predict the winner of the 2016 Presidential election (Trump vs. Clinton) in each county in the US. To do this, we will

Decision Trees

Part B – Stopping Conditions

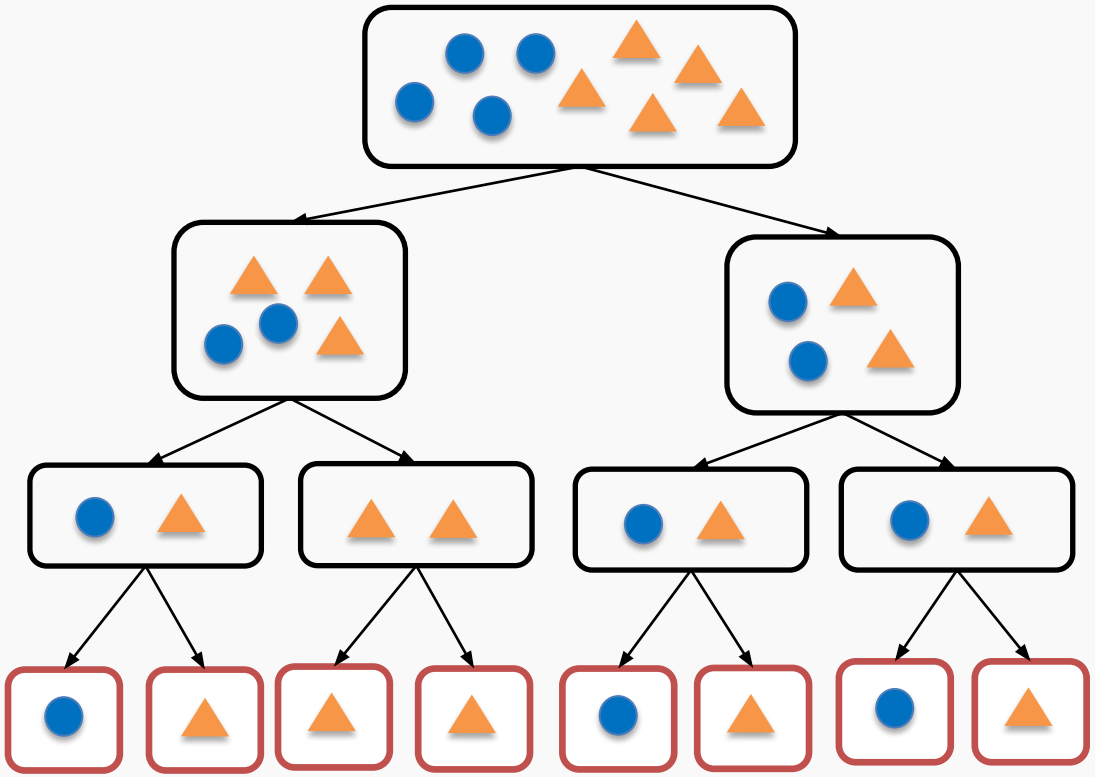
Pavlos Protopapas

Outline

- Motivation
- Decision Trees - Classification
- Splitting Criteria
- **Stopping Conditions**
- Decision Trees – Regression
- Numerical vs Categorical Attributes
- Pruning

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.

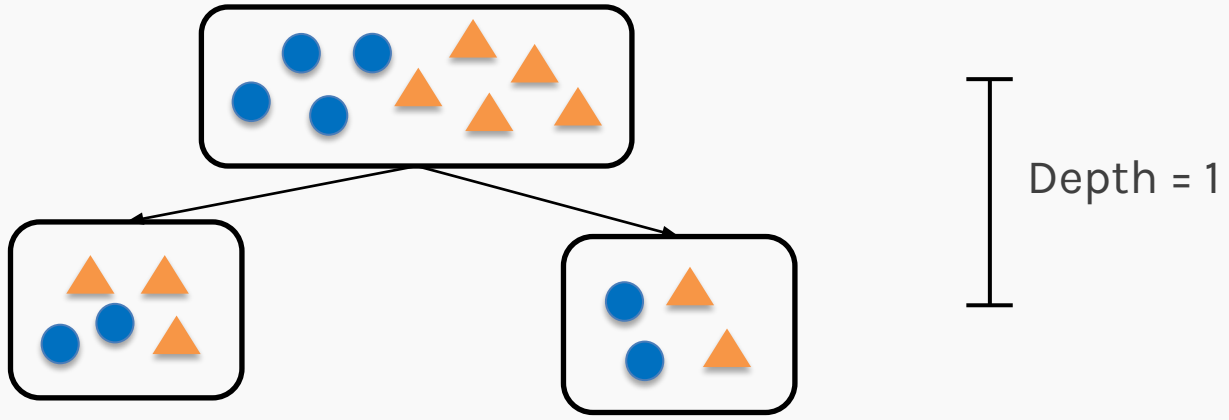
Stopping Conditions

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.

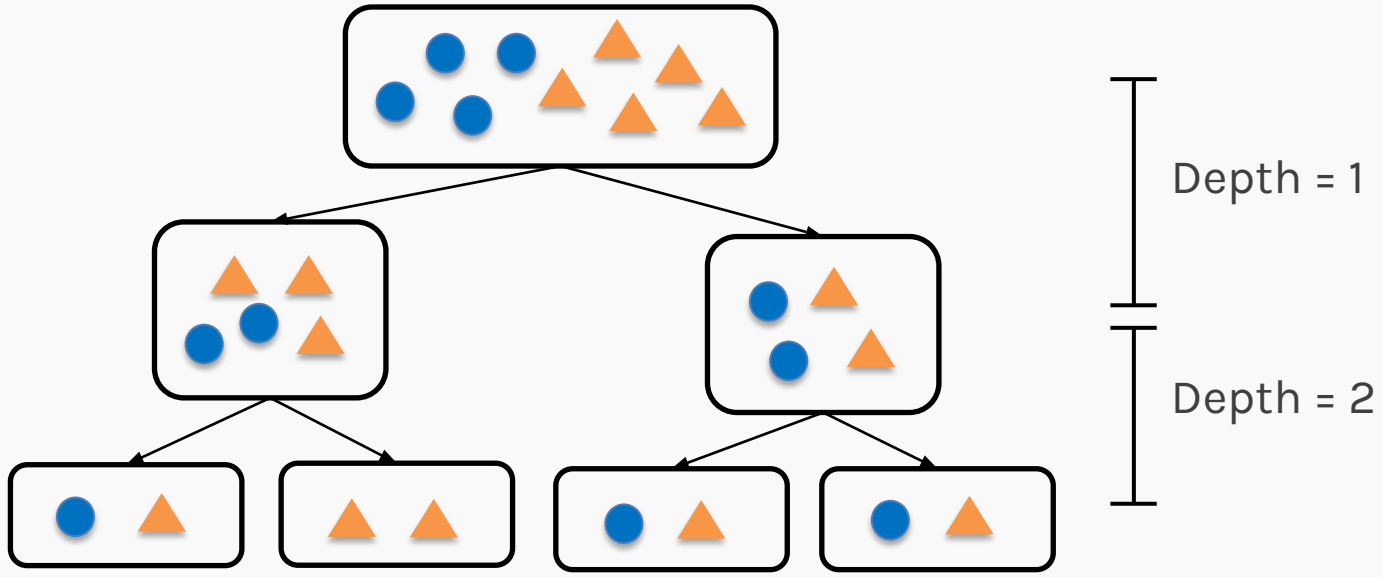
max_depth = 1



Stopping Conditions

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

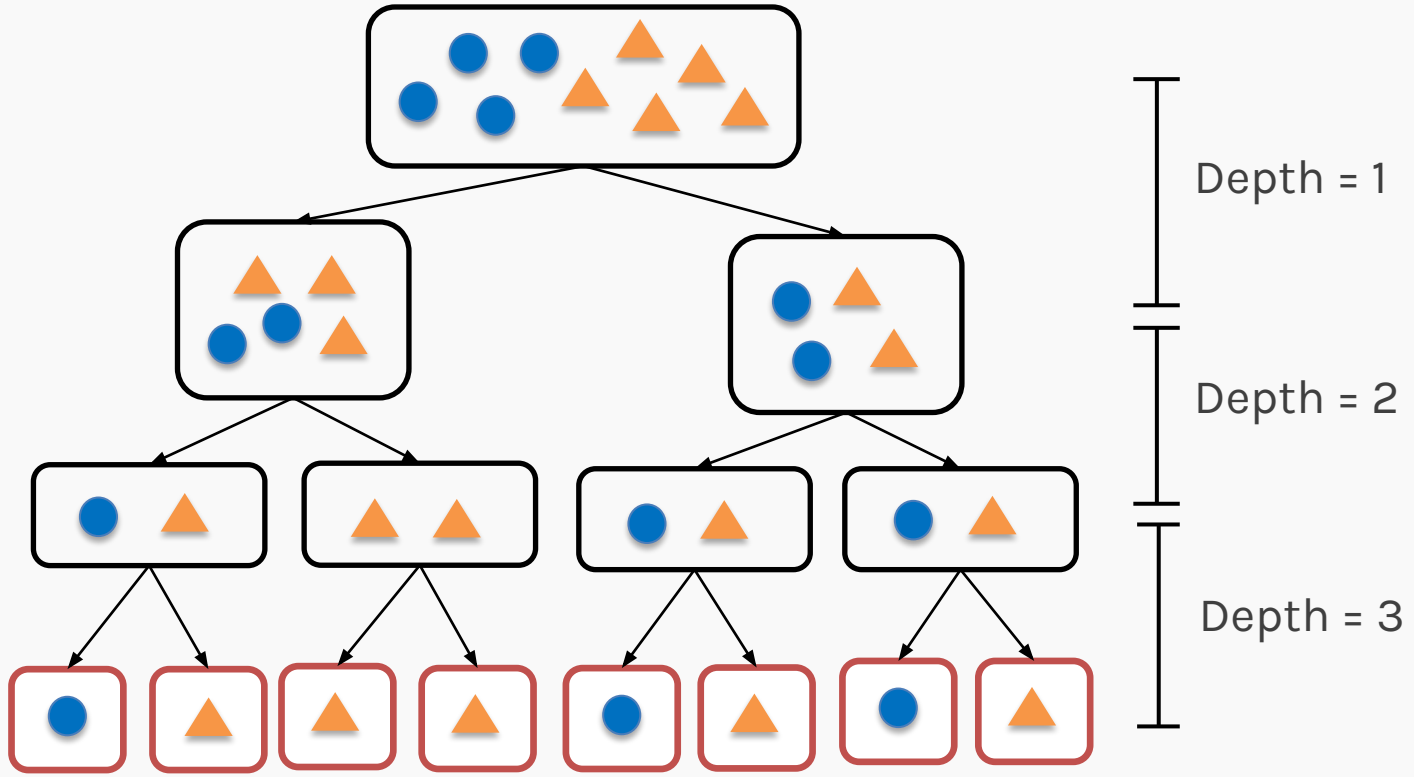
max_depth = 2



Stopping Conditions

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

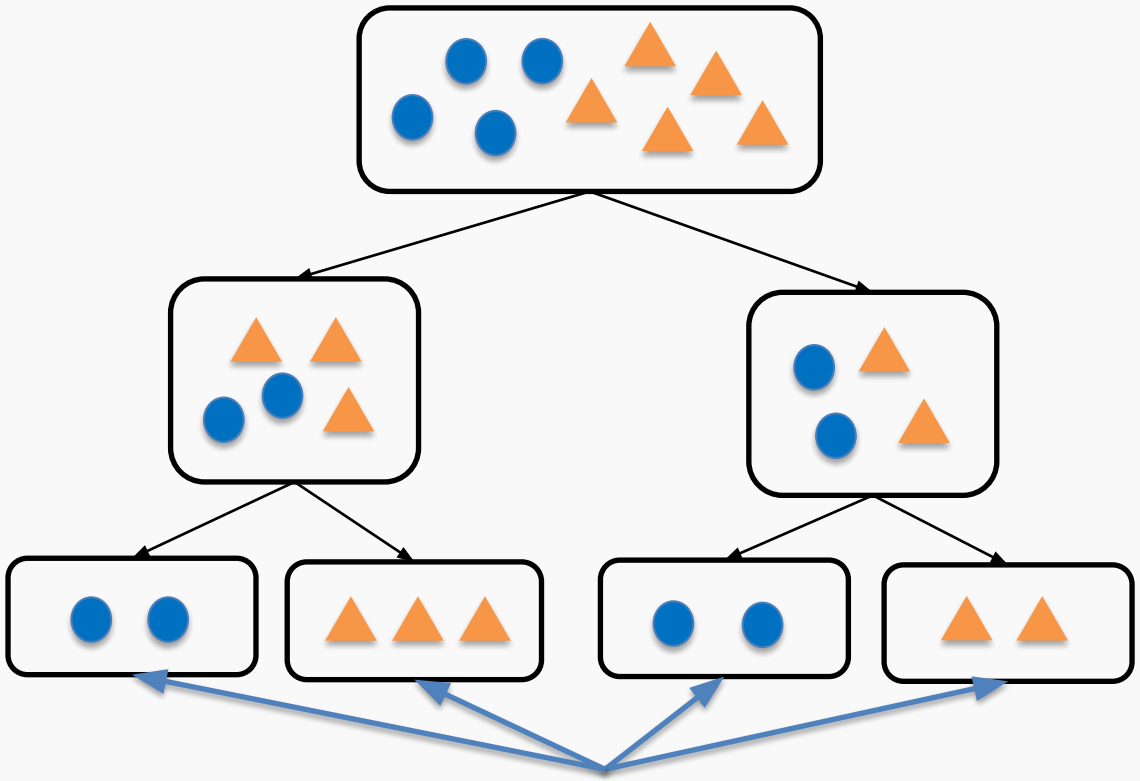
max_depth = 3



Stopping Conditions

Other common simple stopping conditions are:

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

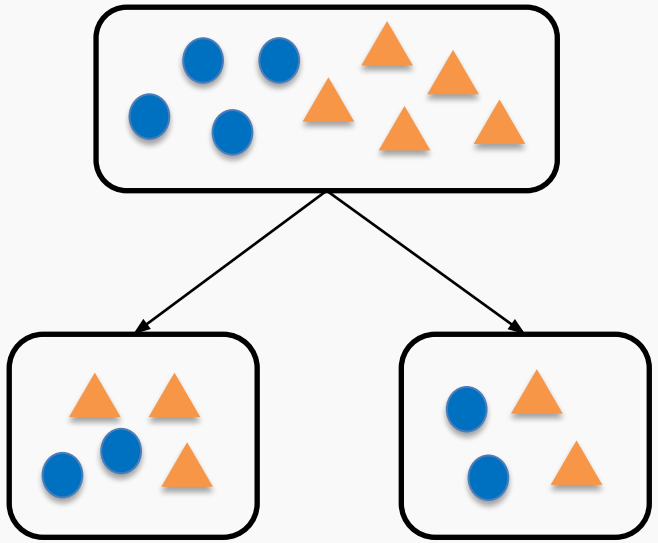


Stopping Conditions

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

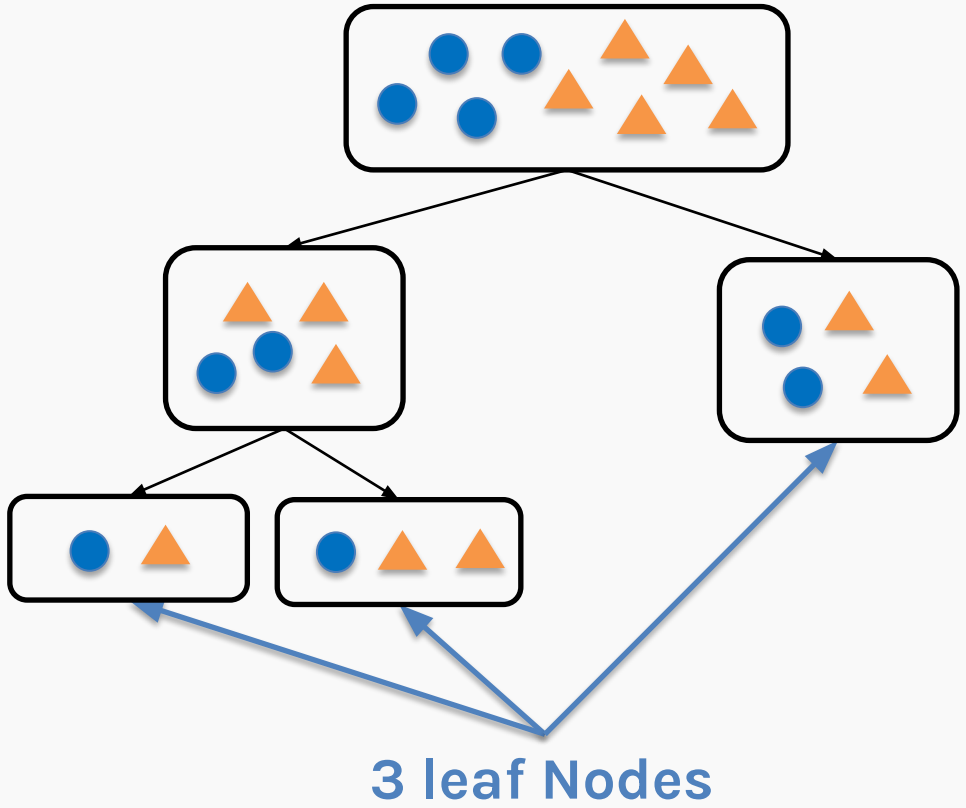


Stopping Conditions

Other common simple stopping conditions are:

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

max_leaf_nodes = 3



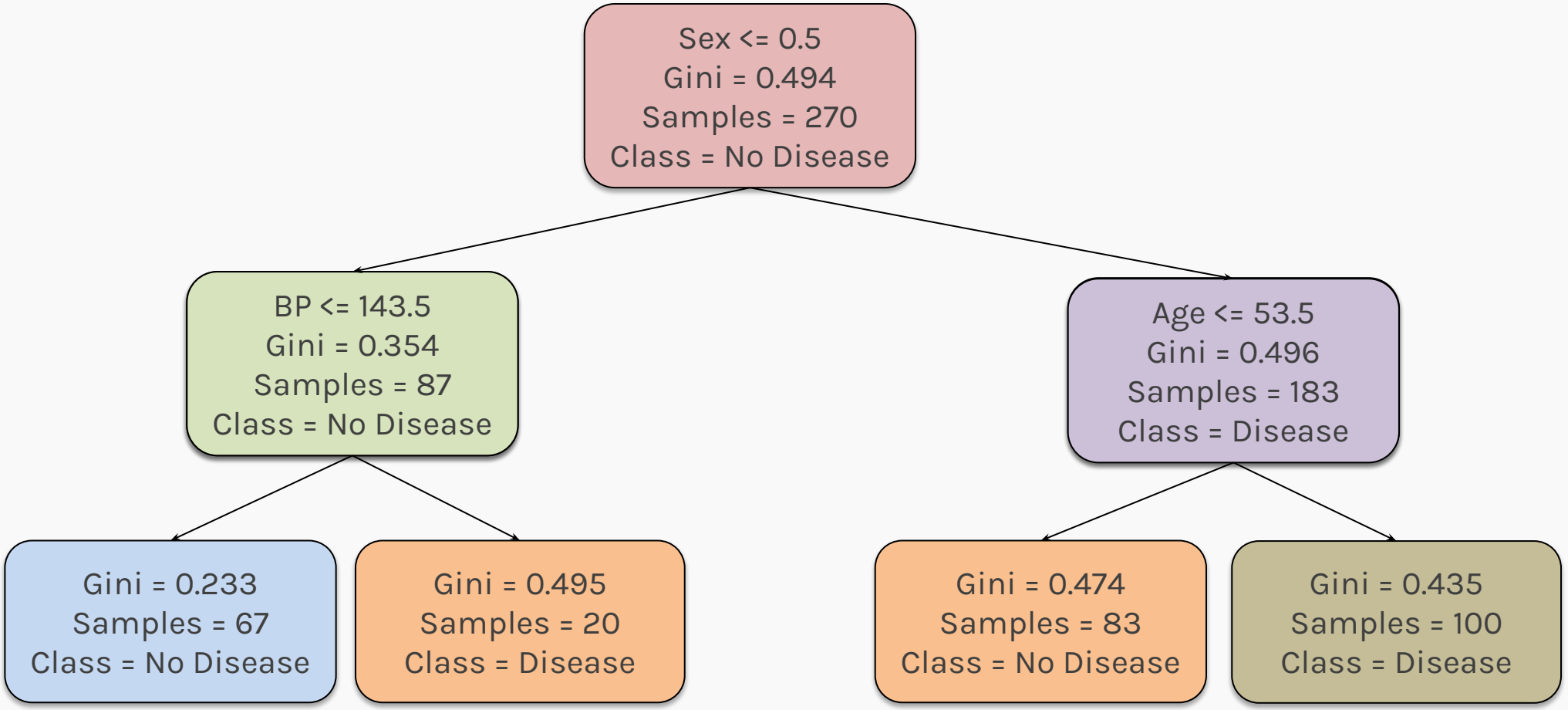
Stopping Conditions

Sklearn grows trees in **depth-first** fashion unless *max_leaf_nodes* is specified.
When *max_leaf_nodes* is specified, it is grown in a **best-first** fashion.

But what is depth-first and best-first fashion?

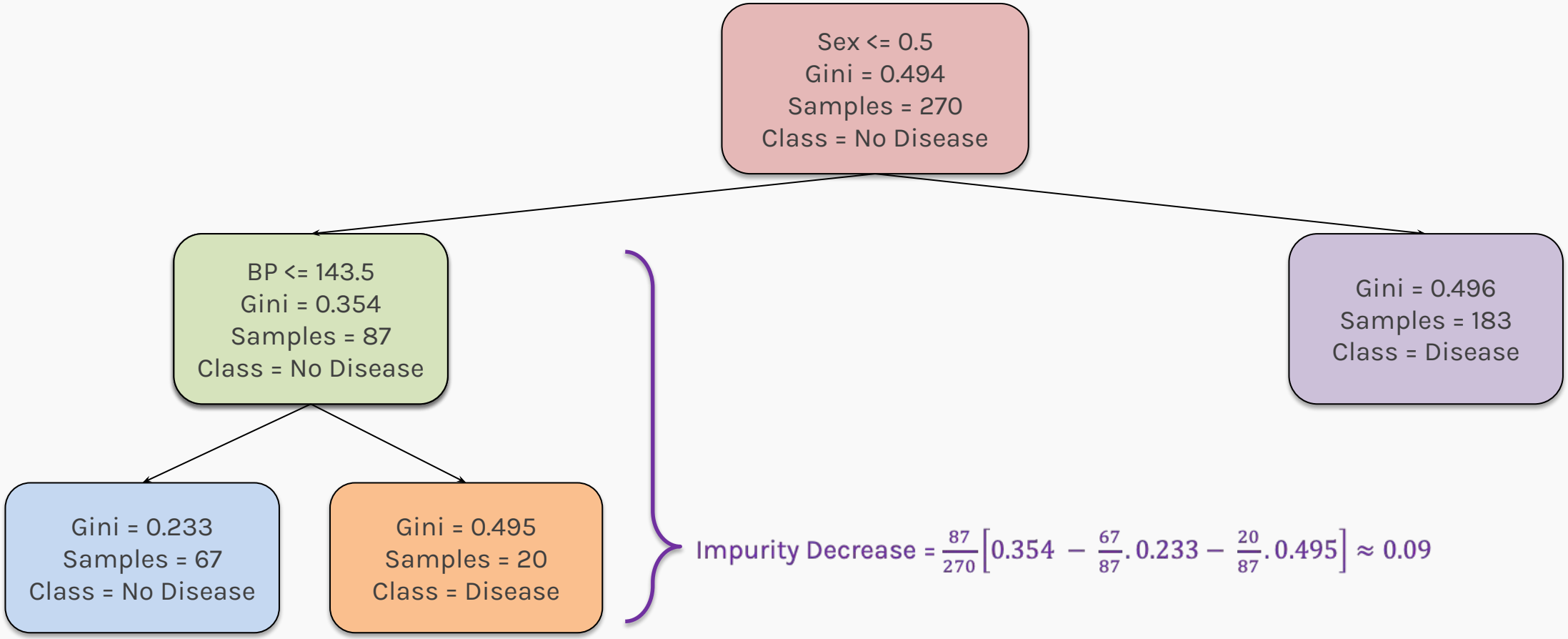
Example 1: Depth-first

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

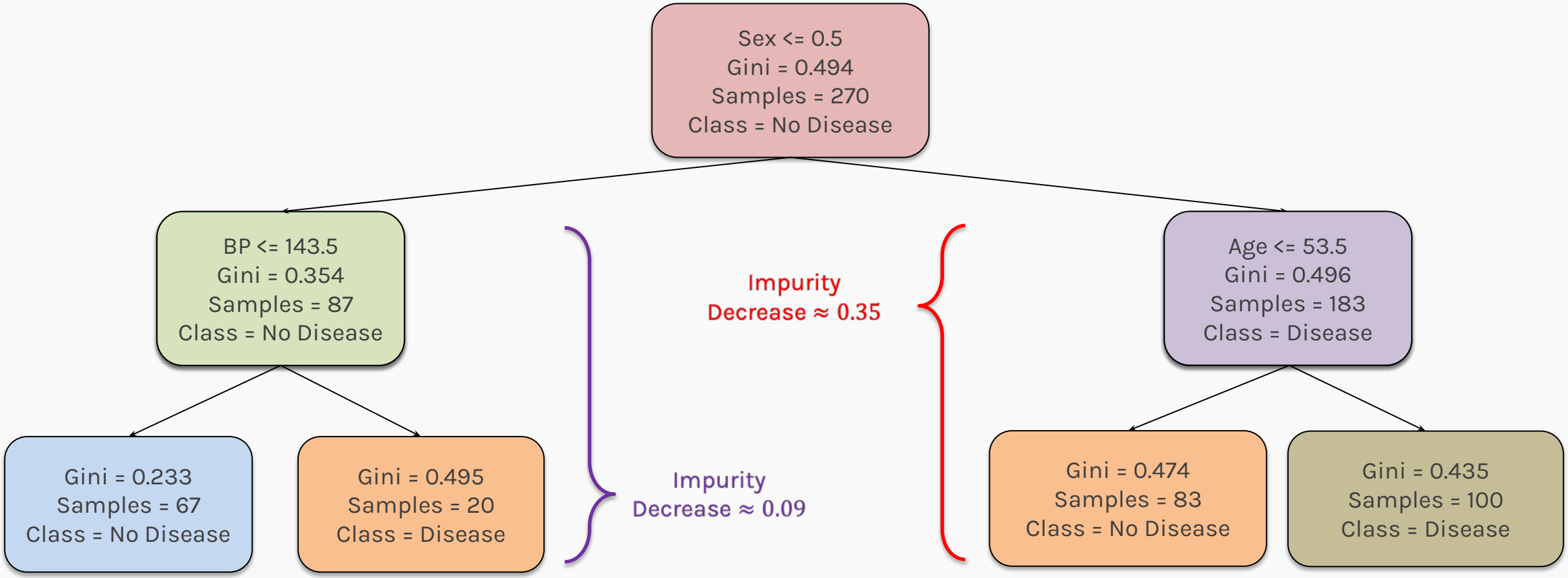


Example 1: Best-first growth

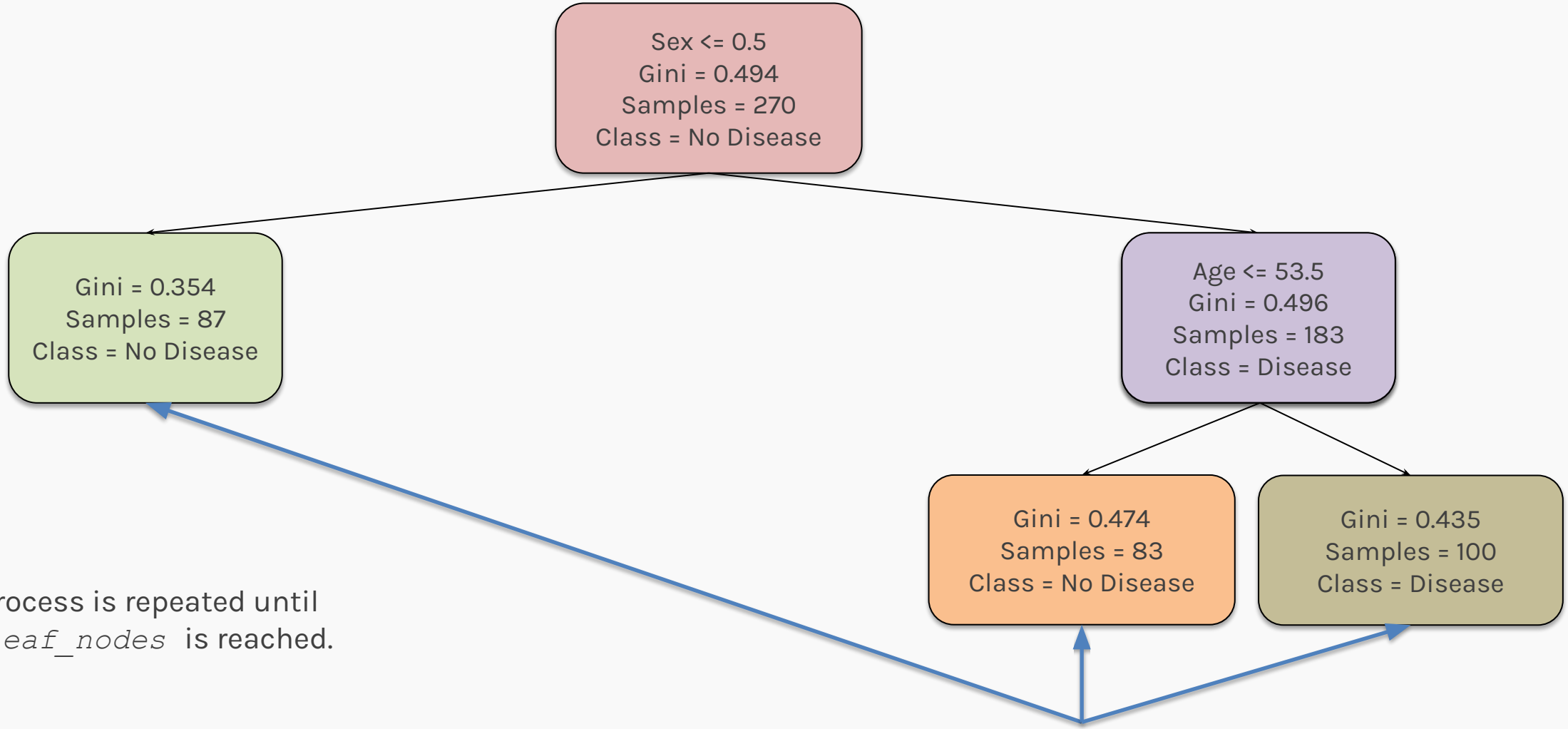
Sklearn determines the best split based on **impurity decrease**. The resulting tree will be the same when fully grown, just the order in which it is built is different.



Example 1: Best-first growth



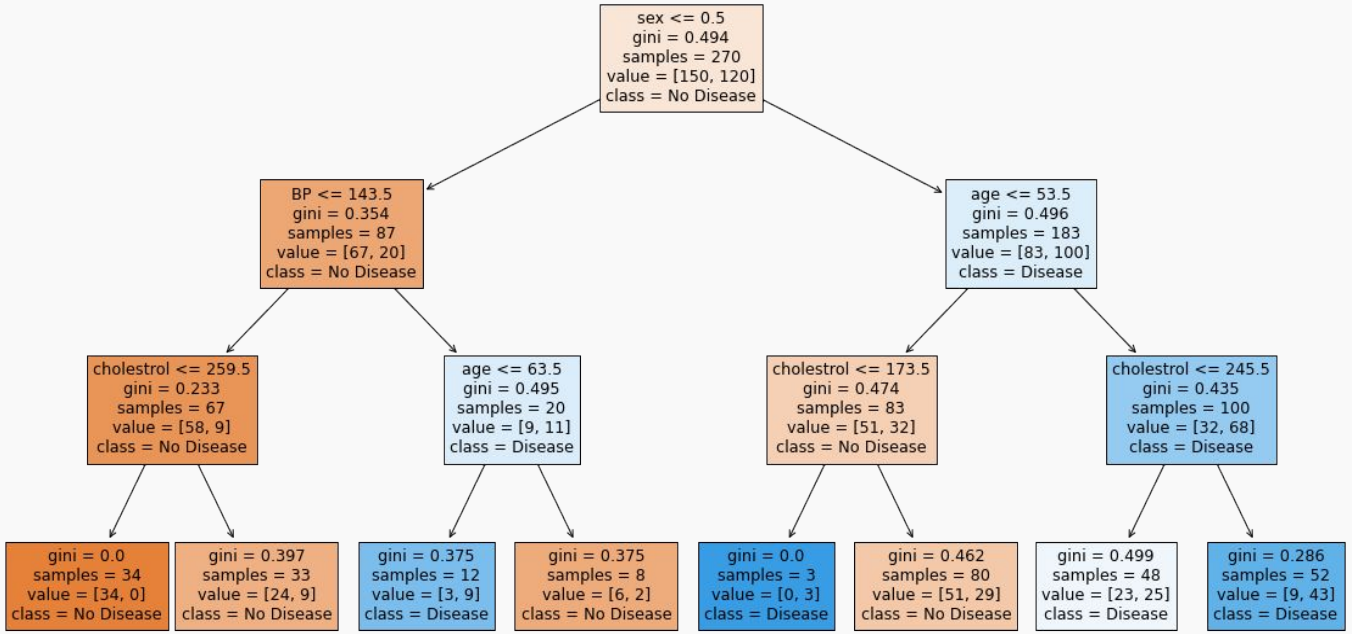
Example 1: Best-first growth



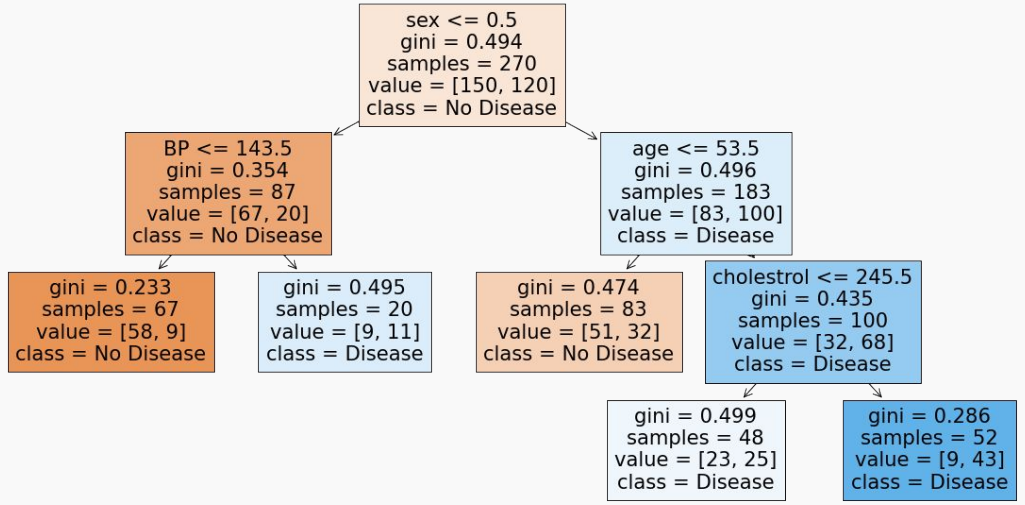
This process is repeated until *max_leaf_nodes* is reached.

Now we compare the impurity decrease of these 3 nodes!

Example 2: Depth-first vs Best-first growth



max_depth = 3



max_leaf_nodes = 5

Stopping Conditions

A more restrictive stopping condition is:

Compute the gain in purity, information or reduction in entropy of splitting a region R into R_1 and R_2:

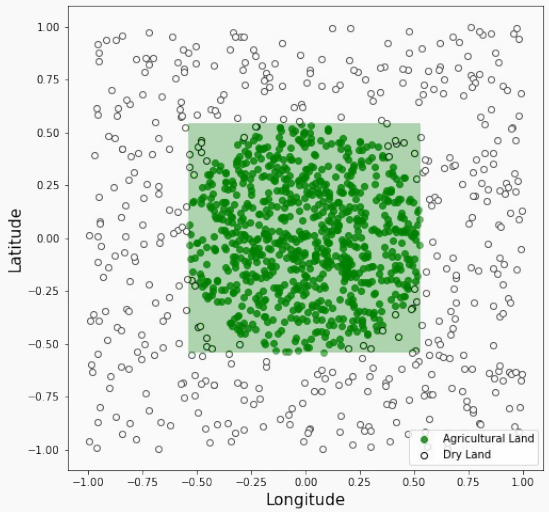
$$Gain(R) = \Delta(R) = m(R) - \frac{N_1}{N} m(R_1) - \frac{N_2}{N} m(R_2)$$

Classification Error/Gini Index/Entropy

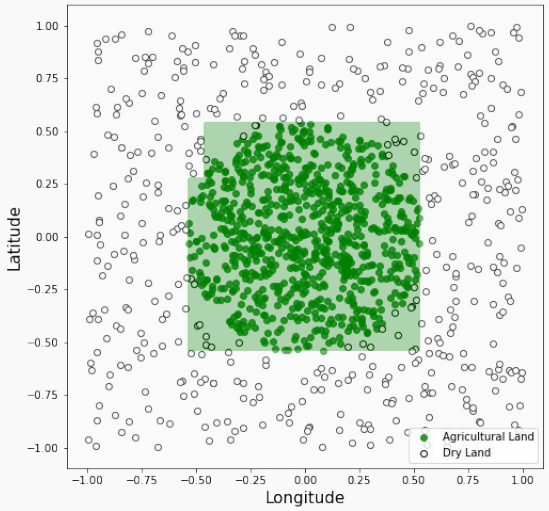
Don't split if the gain is less than some pre-defined threshold (`min_impurity_decrease`).

Variance vs Bias

max_depth = 4

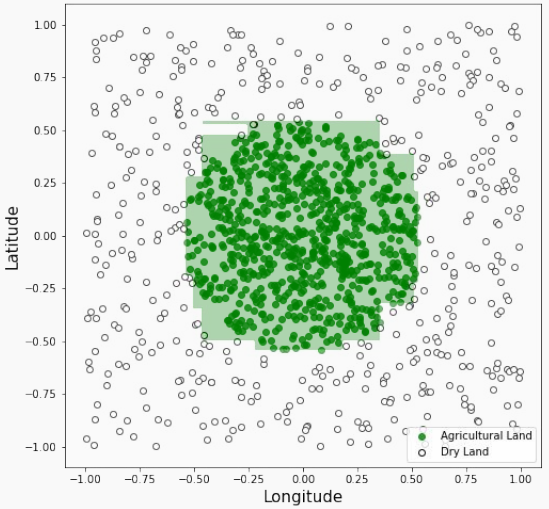


max_depth = 6



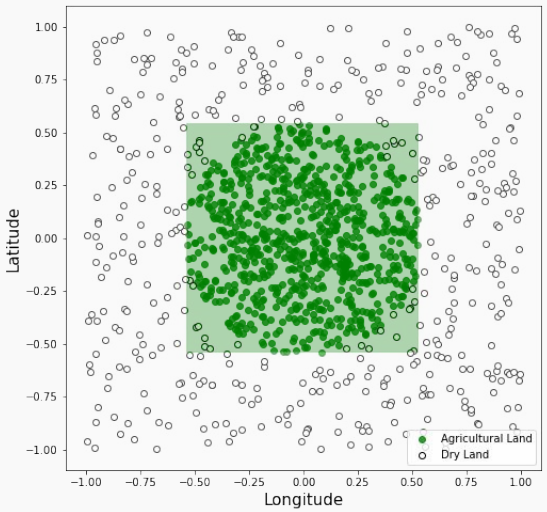
• • • •

max_depth = 5000

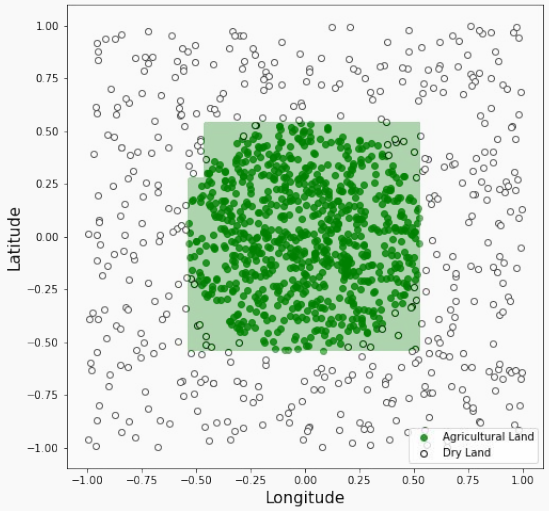


Variance vs Bias

max_depth = 4

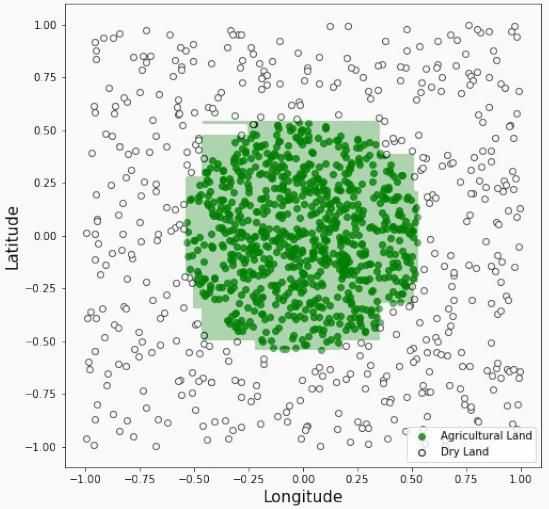


max_depth = 6



• • • •

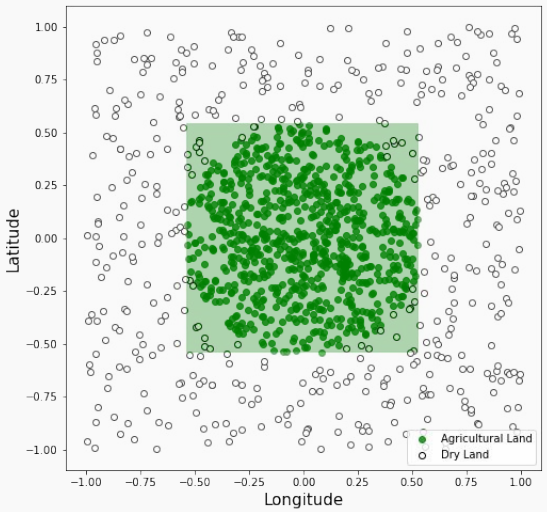
max_depth = 5000



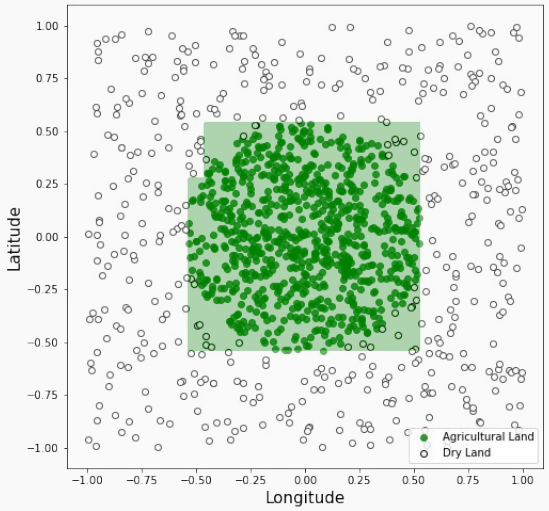
Bias decreases (can overfits)

Variance vs Bias

max_depth = 4

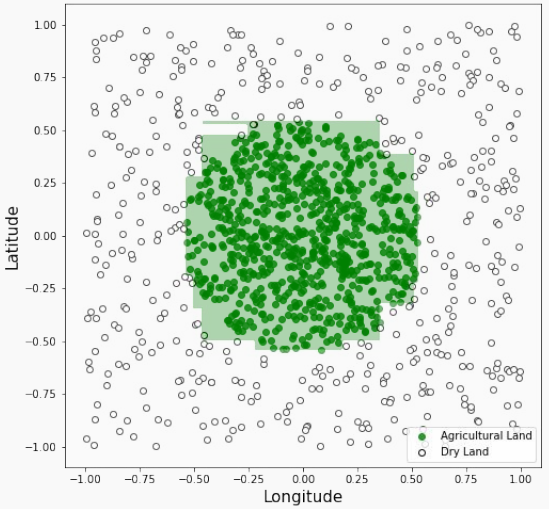


max_depth = 6



• • • •

max_depth = 5000

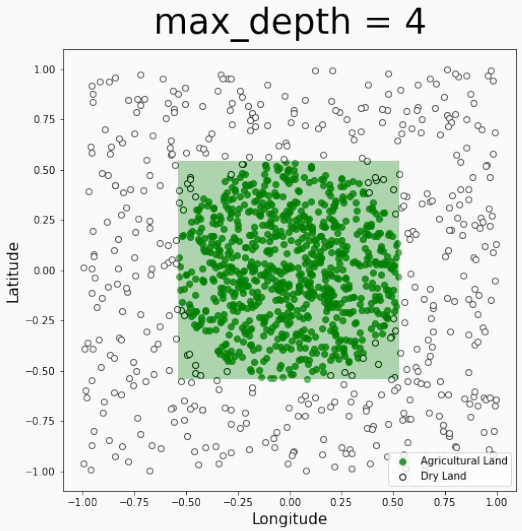


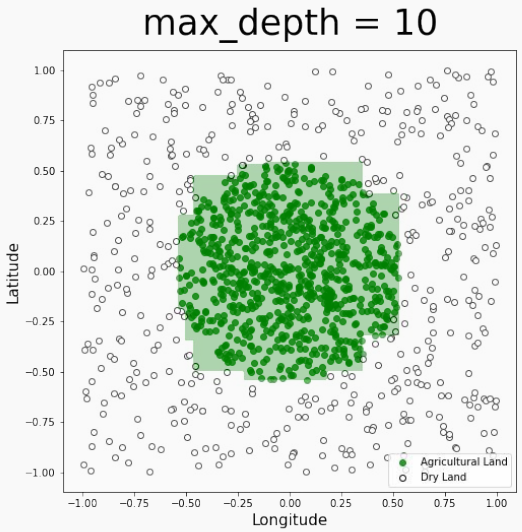
Bias decreases (can overfits)

Variance decreases (can underfit)

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

Variance vs Bias





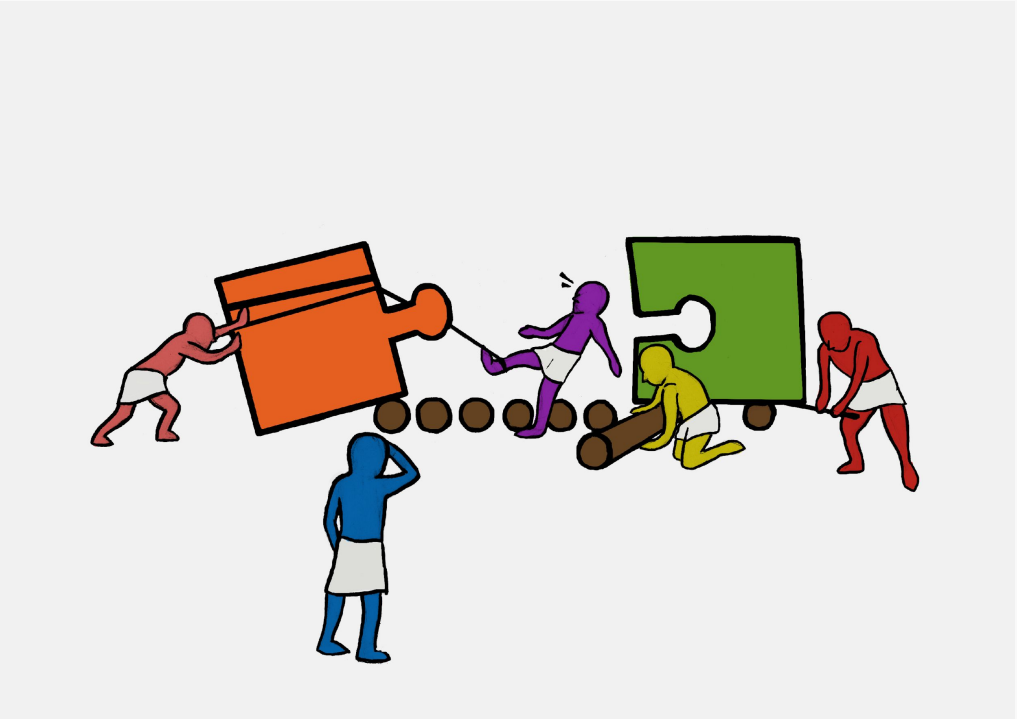
- **High Bias:** Trees of low depth are not a good fit for the training data - it's unable to capture the nonlinear boundary separating the two classes.
- **Low Variance:** Trees of low depth are robust to slight perturbations in the training data - the square carved out by the model is stable if you 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).
- **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

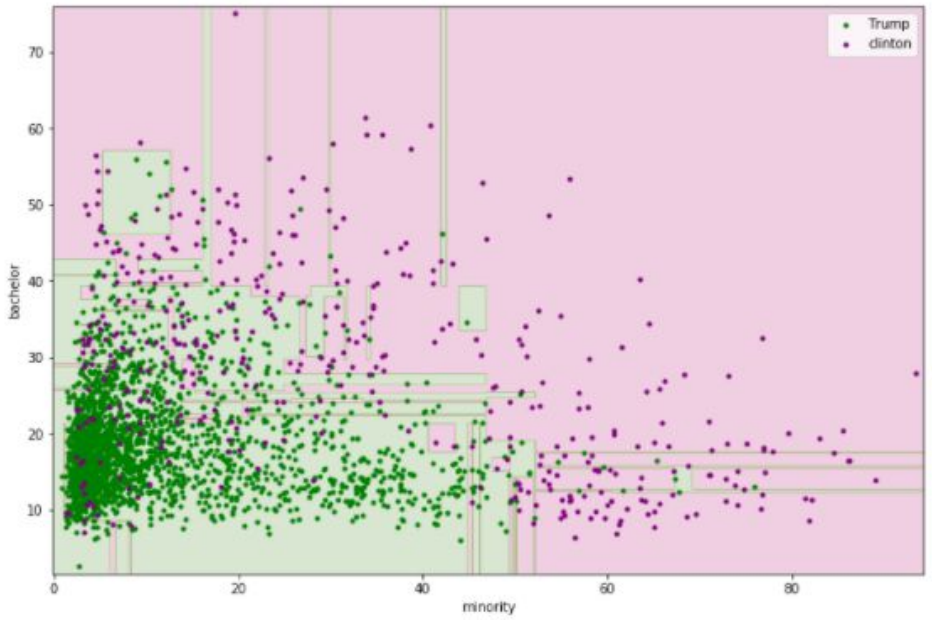
The appropriate parameters can be determined by evaluating the model on a validation data set or, better yet, with..

cross-validation



Exercise - Classification using Decision Tree

The goal of this exercise is to get comfortable using Decision Trees for classification in `sklearn`. Eventually, you will produce a plot similar to the one given below:



Instructions:

- Read the train and test datafile as Pandas data frame.
- Use `minority` and `bachelor` as the predictor variables and `won` as the response.

Decision Trees

Part C – Regression using trees

Pavlos Protopapas

Outline

- Motivation
- Decision Trees - Classification
- Splitting Criteria
- Stopping Conditions
- **Decision Trees – Regression**
- Numerical vs Categorical Attributes
- Pruning

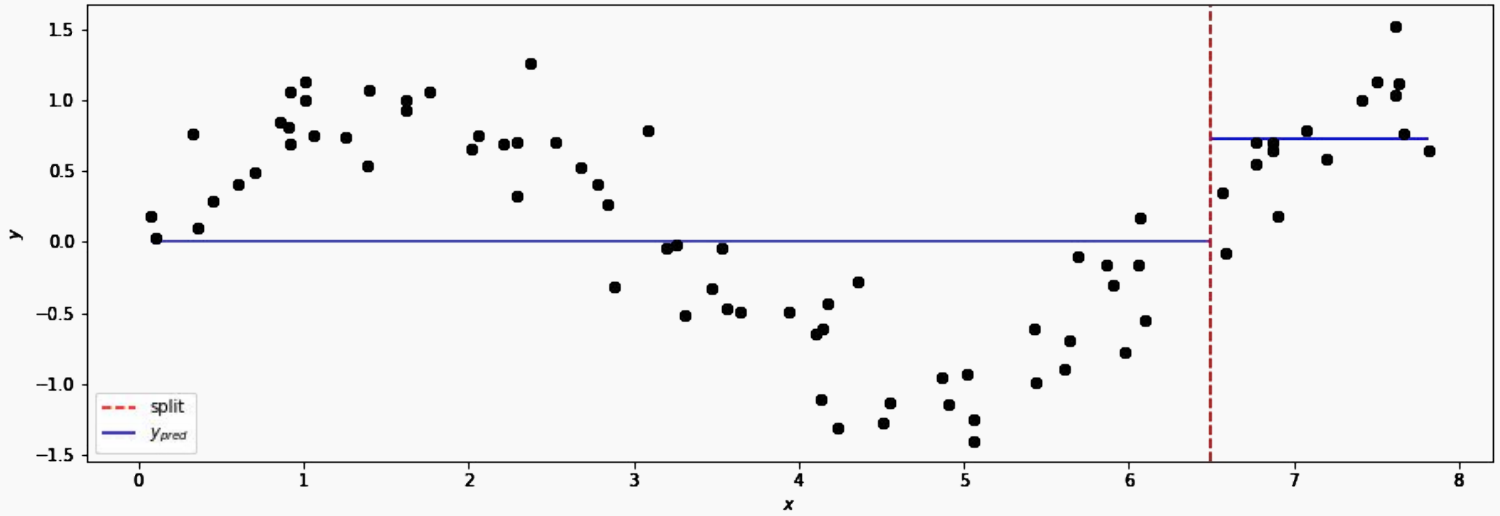
Regression Trees

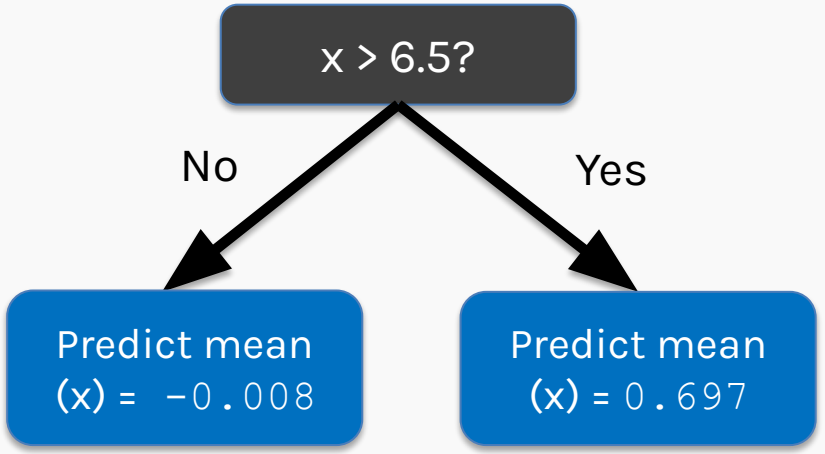
How can this decision tree approach apply to a ***regression problem*** (quantitative outcome)?

Questions to consider:

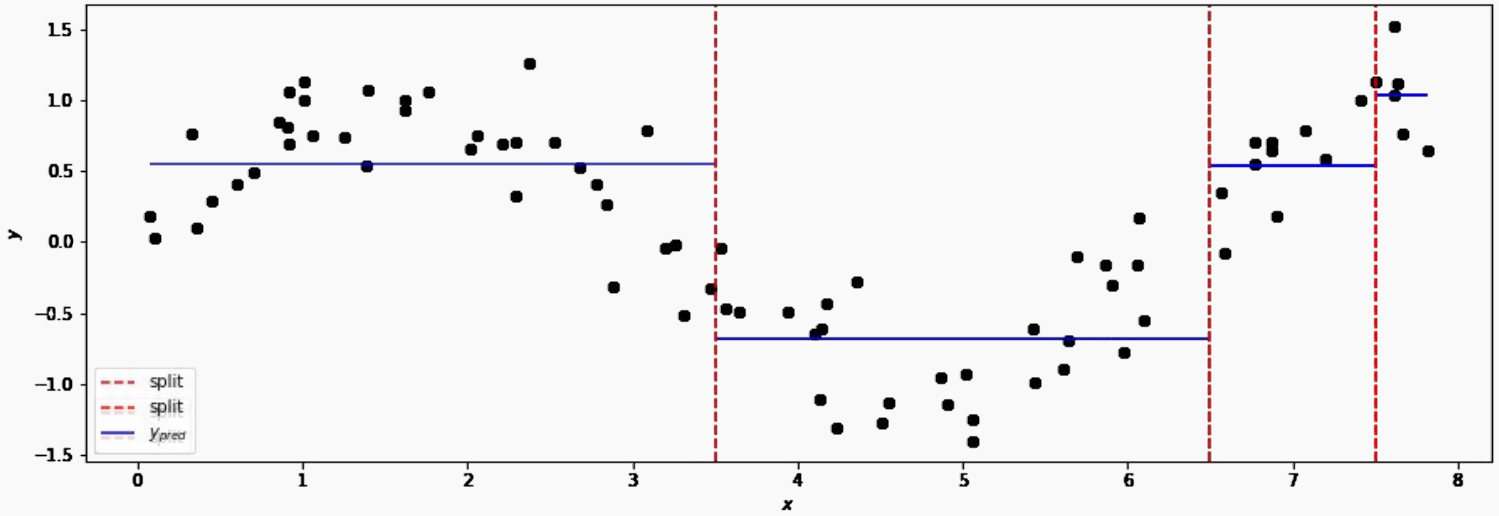
- How would you determine any splitting criteria?
- What would be a reasonable objective function?
- How would you perform prediction in each leaf?

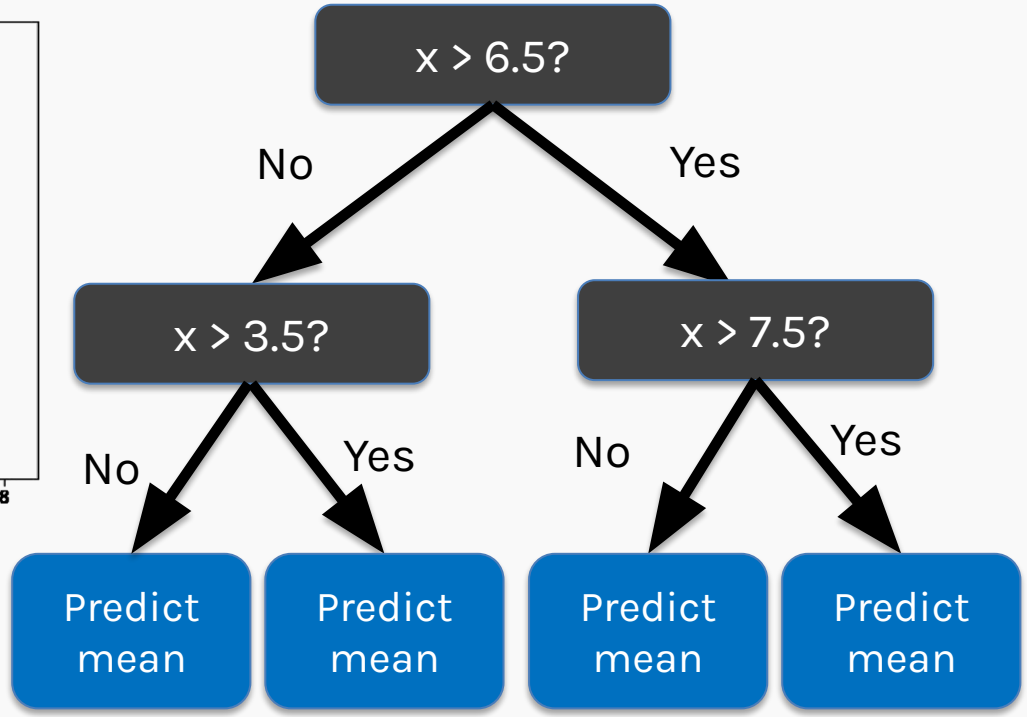
Regression Tree (max_depth = 1)



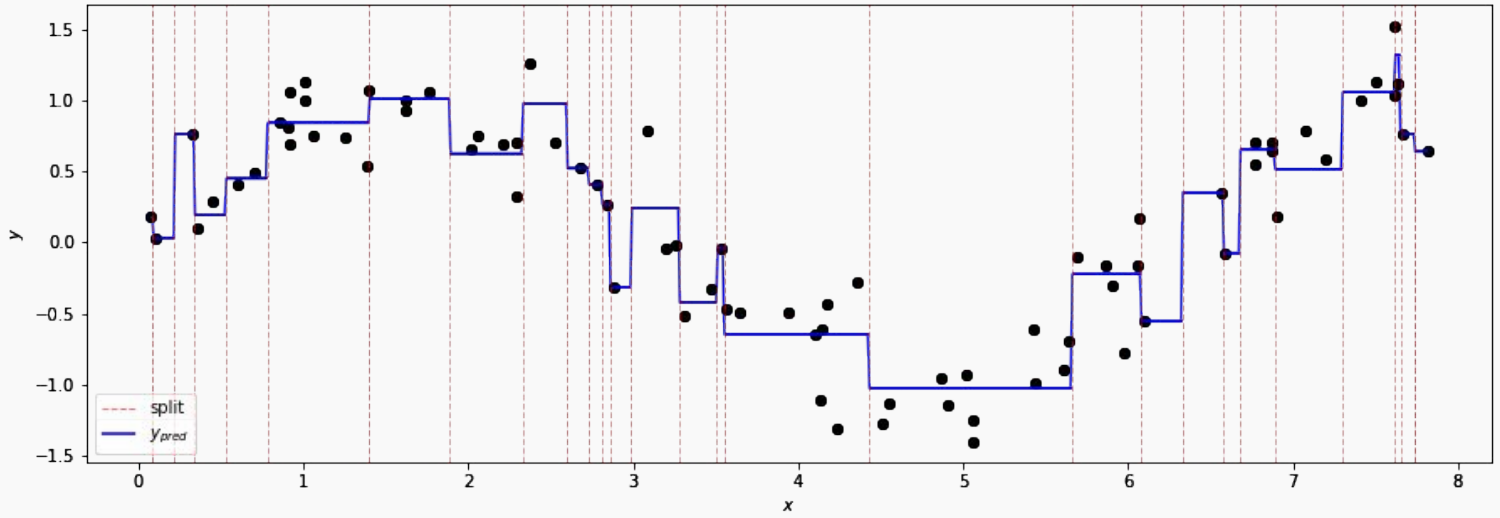


Regression Tree (max_depth = 2)

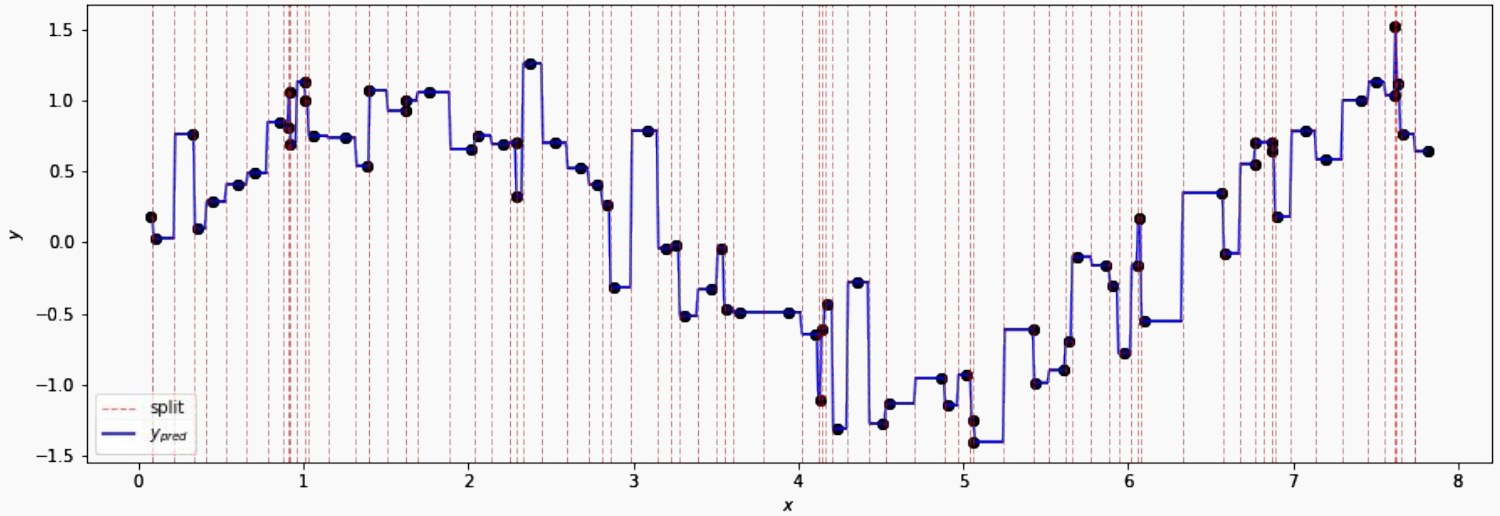




Regression Tree (max_depth = 5)



Regression Tree (max_depth = 10)



Regression Trees Prediction

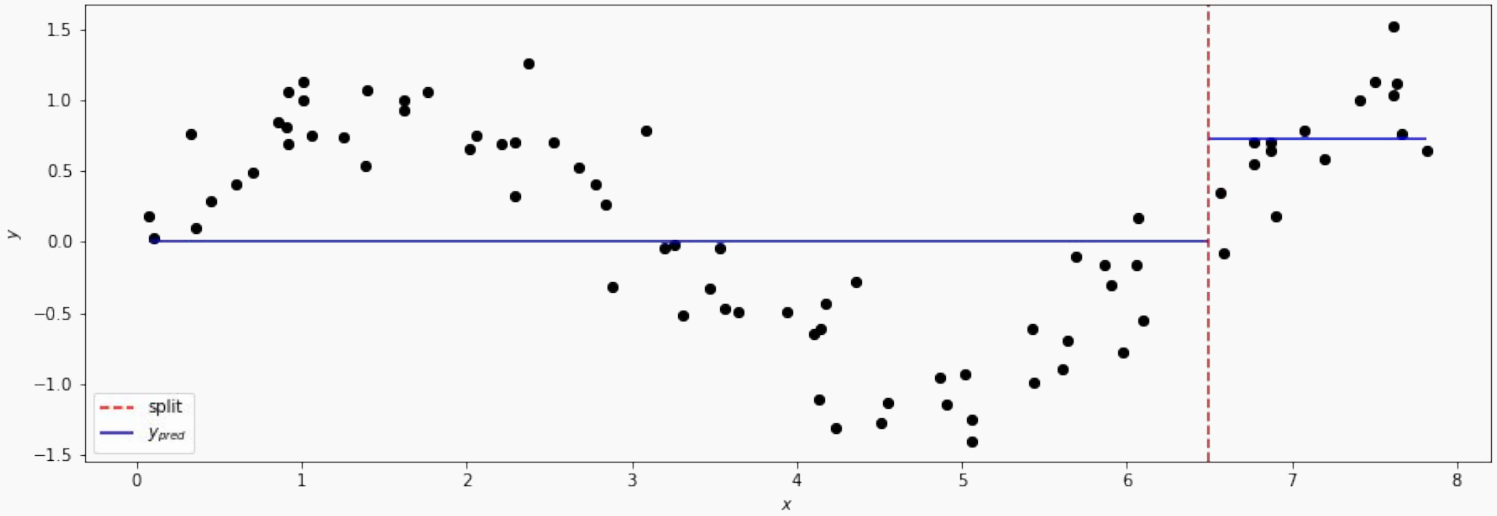
For any data point x_i

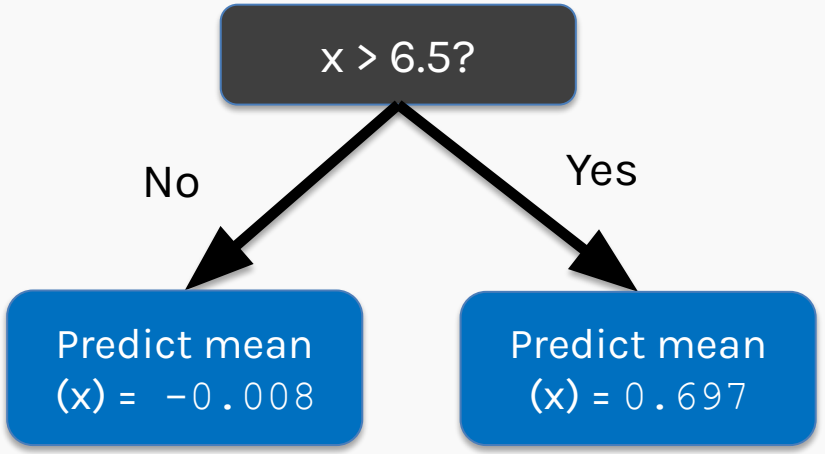
1. Traverse the tree until we reach a leaf node.

2. Averaged value of the response variable y's in the leaf (this is from the training set) is the $\widehat{y_i}$.

Splitting Criteria

Consider the example we saw before:

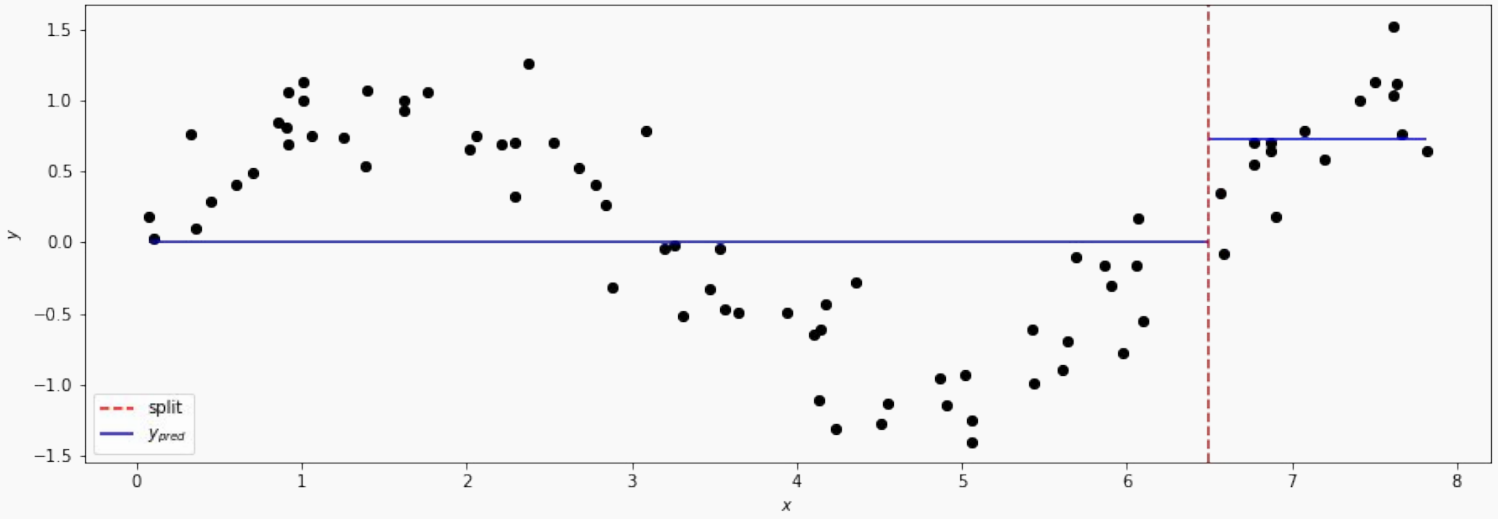




Question: How did we choose the splitting criteria for this regression tree?

Splitting Criteria

We can assess the quality of this split by measuring the **mean squared error** made by each newly created region by calculating:



$$MSE(R_r) = \frac{1}{n} \sum_{i \in R_{wr}} (y_i - \bar{y}_{R_r})^2$$

Note: This is the same as the variance!

Splitting Criteria

We need to take the **weighted average** over both regions so the number of points in each region is taken into consideration:

$$\min_{p,\, t_p} \left[\frac{N_1}{N} MSE(R_1) + \frac{N_2}{N} MSE(R_2) \right]$$

Stopping Conditions

Most of the stopping conditions we saw for classification trees like **maximum depth** or **minimum number of points** in region can still be applied.

In the place of purity gain, we can instead compute **accuracy gain** for splitting a region R and stop the tree when the gain is less than some pre-defined threshold.

$$\text{Gain}(R) = \Delta(R) = MSE(R) - \frac{N_1}{N} MSE(R_1) - \frac{N_2}{N} MSE(R_2)$$

Outline

- Motivation
- Decision Trees - Classification
- Splitting Criteria
- Stopping Conditions
- Decision Trees – Regression
- **Numerical vs Categorical Attributes**
- Pruning

Numerical vs Categorical Attributes

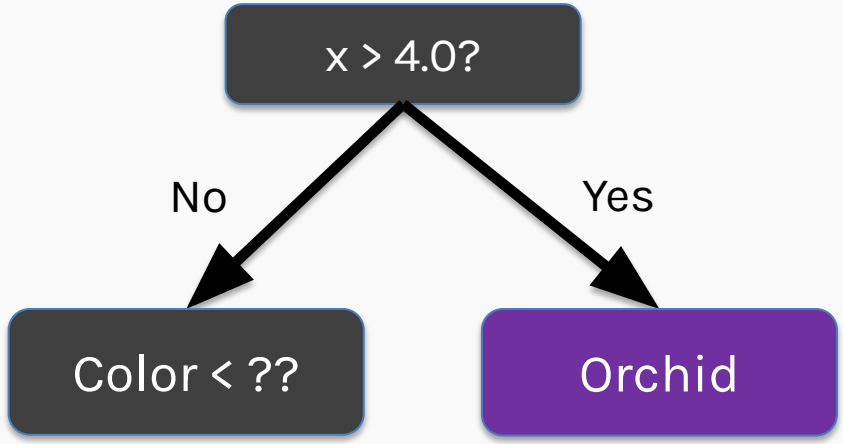
Consider the following data:

Sepal width	Color	Flower
3.0 mm	Yellow	Sunflower
3.5 mm	Red	Rose
4.5 mm	Purple	Orchid
3.7 mm	Purple	Tulip

Question: How do we construct a decision tree for this data?

Numerical vs Categorical Attributes

Sepal width	Color	Flower
3.0 mm	Yellow	Sunflower
3.5 mm	Red	Rose
4.5 mm	Purple	Orchid
3.7 mm	Purple	Tulip



Note that the 'compare and branch' method by which we defined classification tree works well for numerical features.

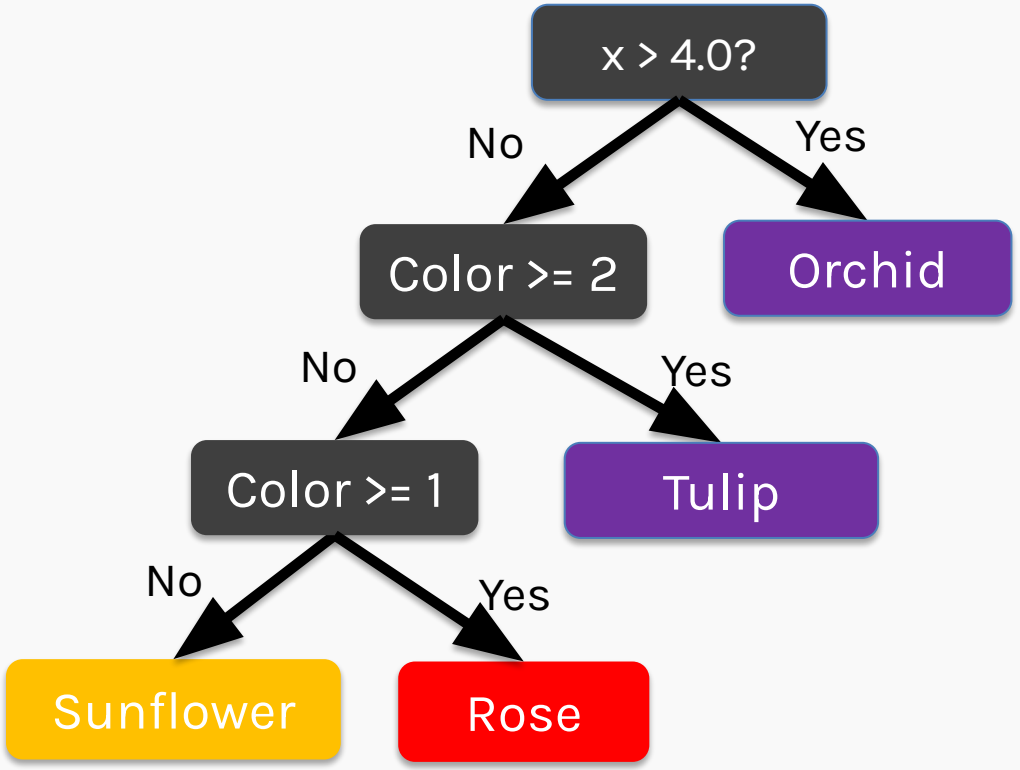
If a feature is categorical (with more than two possible values), comparisons like *feature* < *threshold* does not make sense.

Numerical vs Categorical Attributes

A simple solution is to encode the values of a categorical feature using numbers and treat this feature like a numerical variable.

If we encode **Yellow** = 0, **Red** = 1, **Purple** = 2, our decision tree can be:

Sepal width	Color	Flower
3.0 mm	0	Sunflower
3.5 mm	1	Rose
4.5 mm	2	Orchid
3.7 mm	2	Tulip



Numerical vs Categorical Attributes

In the example, we encoded:

Yellow = 0, **Red** = 1, **Purple** = 2

Then the possible non-trivial splits on **color** are:

{{**Yellow**}, {**Red**, **Purple**}} and {{**Yellow**, **Red**},{**Purple**}}

But if we encode the categories numerically as:

Yellow = 2, **Red** = 0, **Purple** = 1

The possible splits are:

{{**Red**}, {**Yellow**, **Purple**}} and {{**Red**, **Purple**}, {**Yellow**}}

Numerical vs Categorical Attributes

{{Yellow}, {Red, Purple}} and {{Yellow, Red},{Purple}}

{{Red}, {Yellow, Purple}} and {{Red, Purple}, {Yellow}}

Depending on the encoding, the splits we optimize over can be different!

In practice, the effect of our choice of naive encoding of categorical variables are often negligible - models resulting from different choices of encoding will perform comparably.

Numerical vs Categorical Attributes

In the example, we used ordinal encoding. If your categorical data is not ordinal, this is not good - you'll end up with splits that do not make sense. How do we encode this?

One-hot-encoding or dummy encoding!

It is computationally more expensive but is implemented in several computational libraries (e.g. `R's randomForest, H20, XGBoost`).

Numerical vs Categorical Attributes

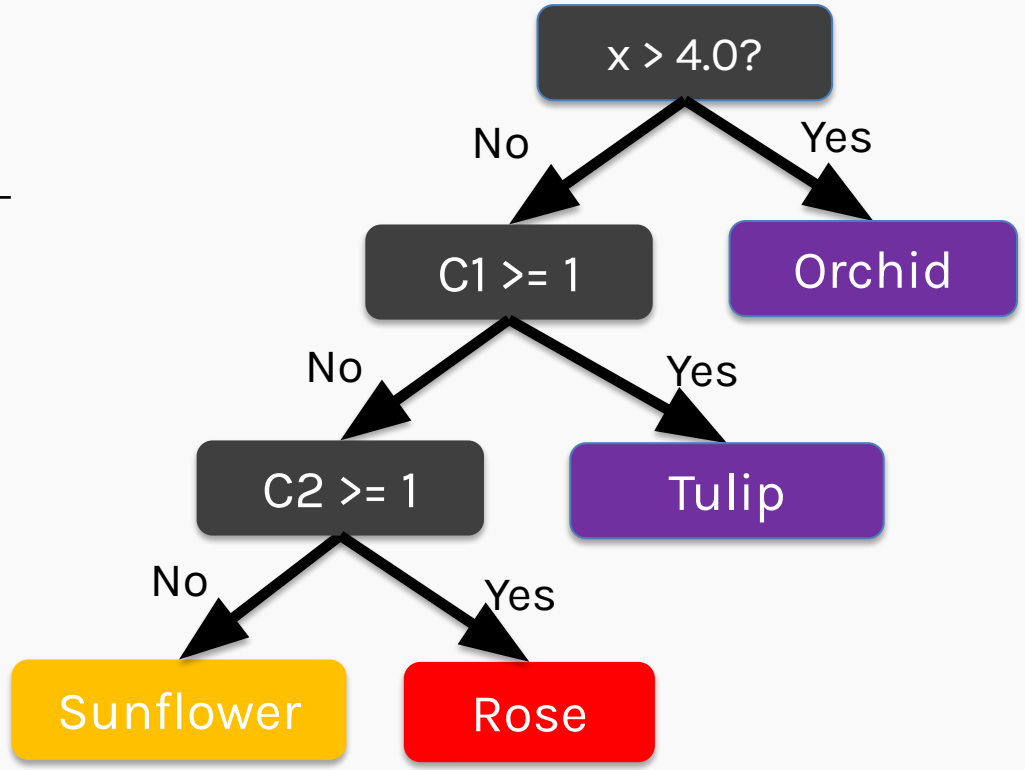
Sepal width	Color	Flower
3.0 mm	Yellow	Sunflower
3.5 mm	Red	Rose
4.5 mm	Purple	Orchid
3.7 mm	Purple	Tulip

OHE →

Sepal width	C1	C2	C3	Flower
3.0 mm	0	0	1	Sunflower
3.5 mm	0	1	0	Rose
4.5 mm	1	0	0	Orchid
3.7 mm	1	0	0	Tulip

Numerical vs Categorical Attributes

Sepal width	C1	C2	C3	Flower
3.0 mm	0	0	1	Sunflower
3.5 mm	0	1	0	Rose
4.5 mm	1	0	0	Orchid
3.7 mm	1	0	0	Tulip



Categorical Predictors

As it stands, sklearn decision trees do not handle categorical data.

From sklearn documentation:
scikit-learn uses an optimized version of the CART algorithm; however, scikit-learn implementation does not support categorical variables for now.

Decision Trees
Part D – Pruning

Pavlos Protopapas

Outline

- Motivation
- Decision Trees - Classification
- Splitting Criteria
- Stopping Conditions
- Decision Trees – Regression
- Numerical vs Categorical Attributes
- **Pruning**

Alternative to Using Stopping Conditions

What is the major issue with pre-specifying a stopping condition?

- You may stop too early or stop too late.

How can we fix this issue?

- Choose several stopping criterion (e.g. set minimal Gain(R) at various levels) and cross-validate which is the best.

What is an alternative approach to this issue?

- Don’t stop. Instead prune back!

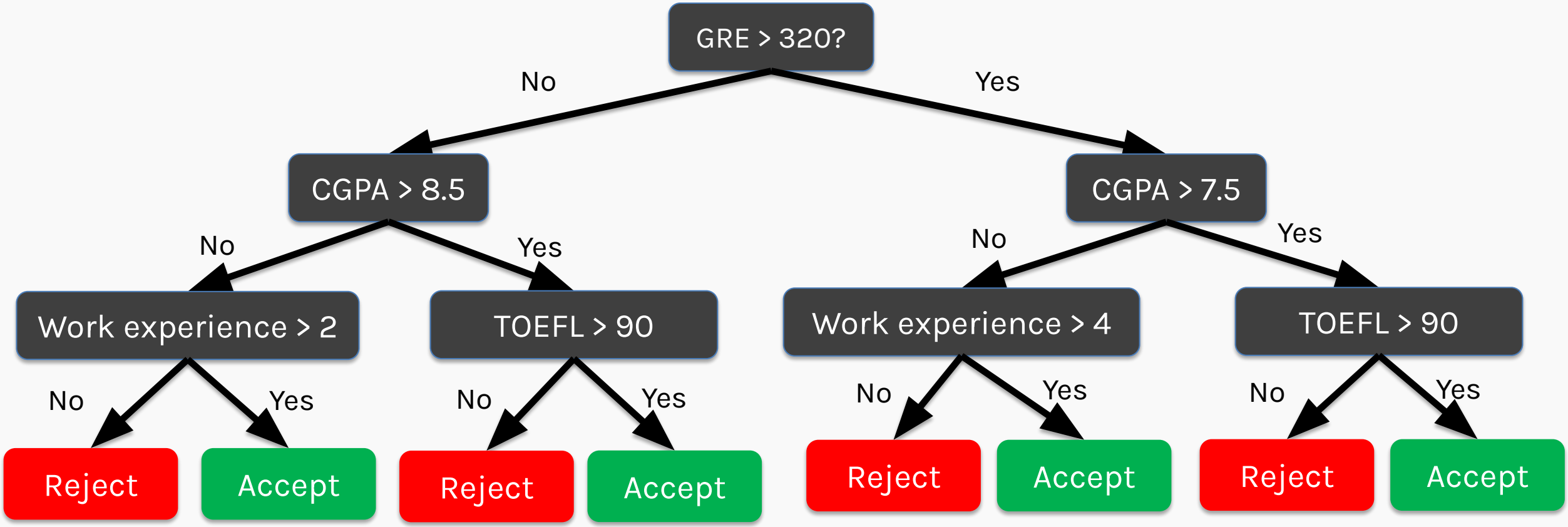
Example: Masters Applications

DISCLAIMER

All trees, variables and thresholds are entirely fictional.

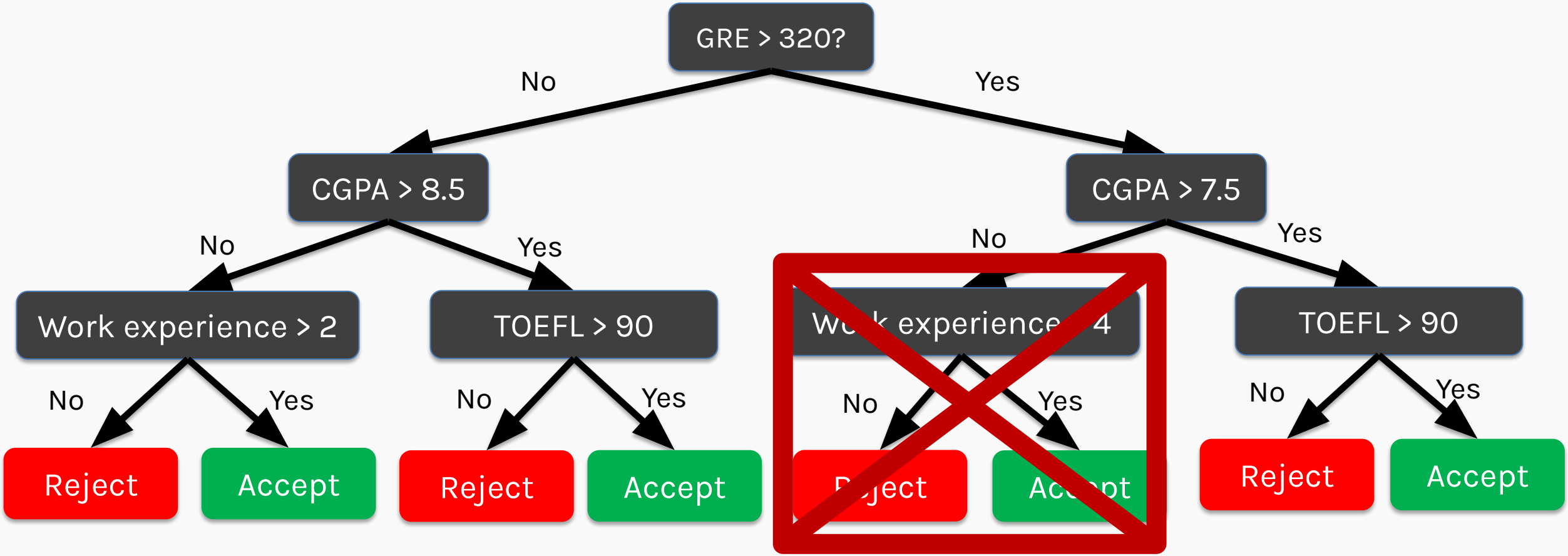
The following representation is only for pedagogical purposes. It does not represent the actual process of an admission committee!

Example: Masters Applications



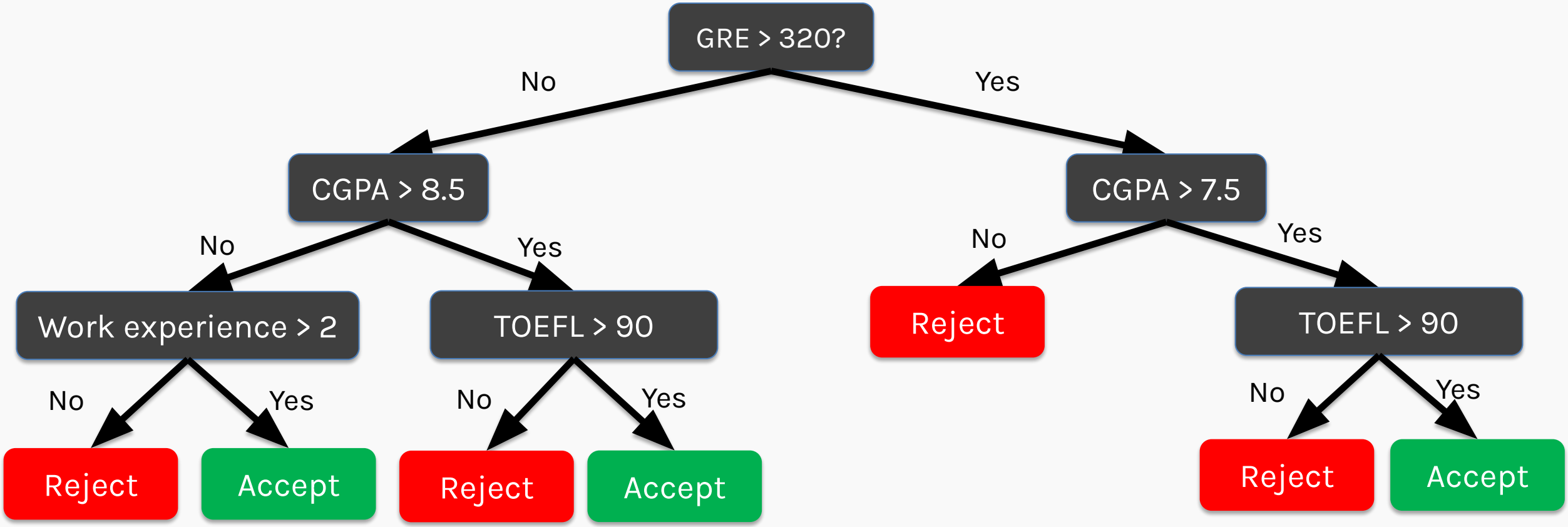
** Above representation is only for pedagogical purposes.

Example: Masters Applications



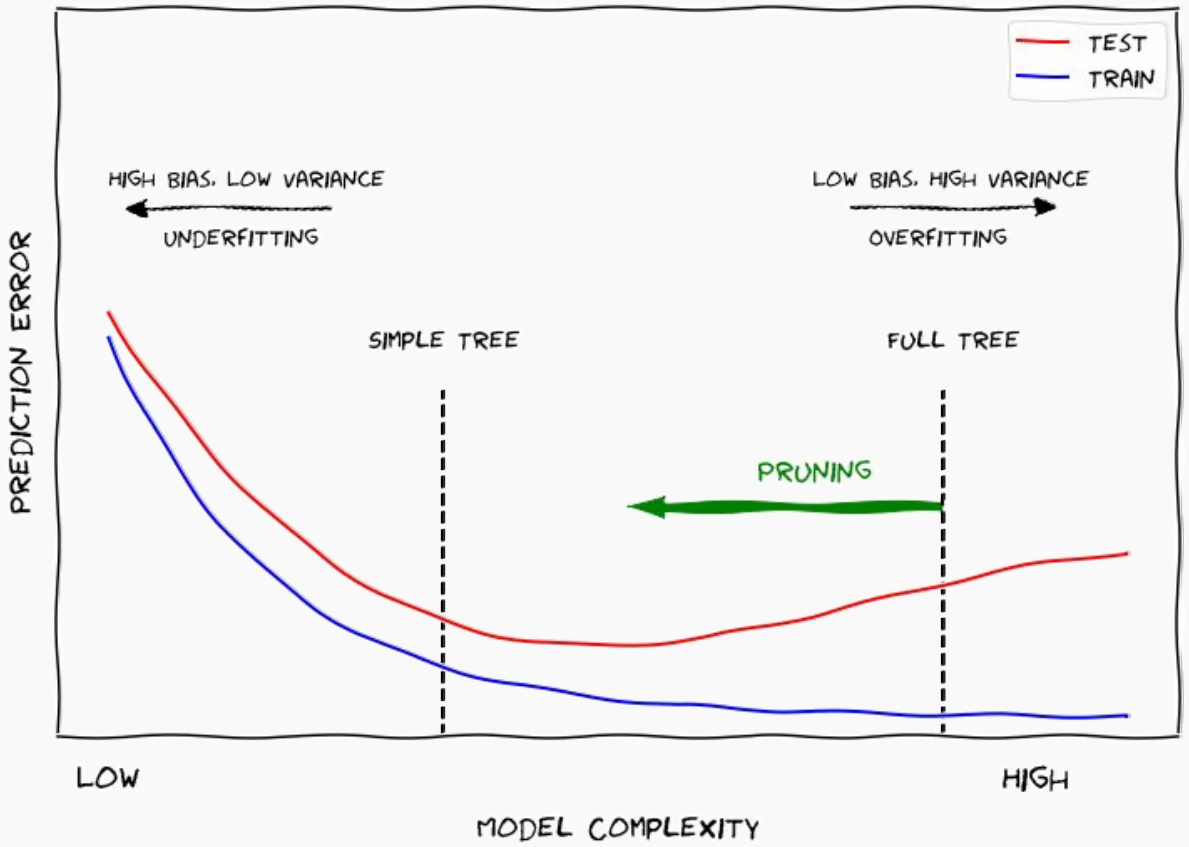
** Above representation is only for pedagogical purposes.

Example: Masters Applications



** Above representation is only for pedagogical purposes.

Motivation for Pruning



Rather than preventing a complex tree from growing, we can obtain a simpler tree by 'pruning' a complex one.

Pruning

There are many methods of pruning, a common one is **cost complexity pruning**:

That is, we add a 'regularization' term:

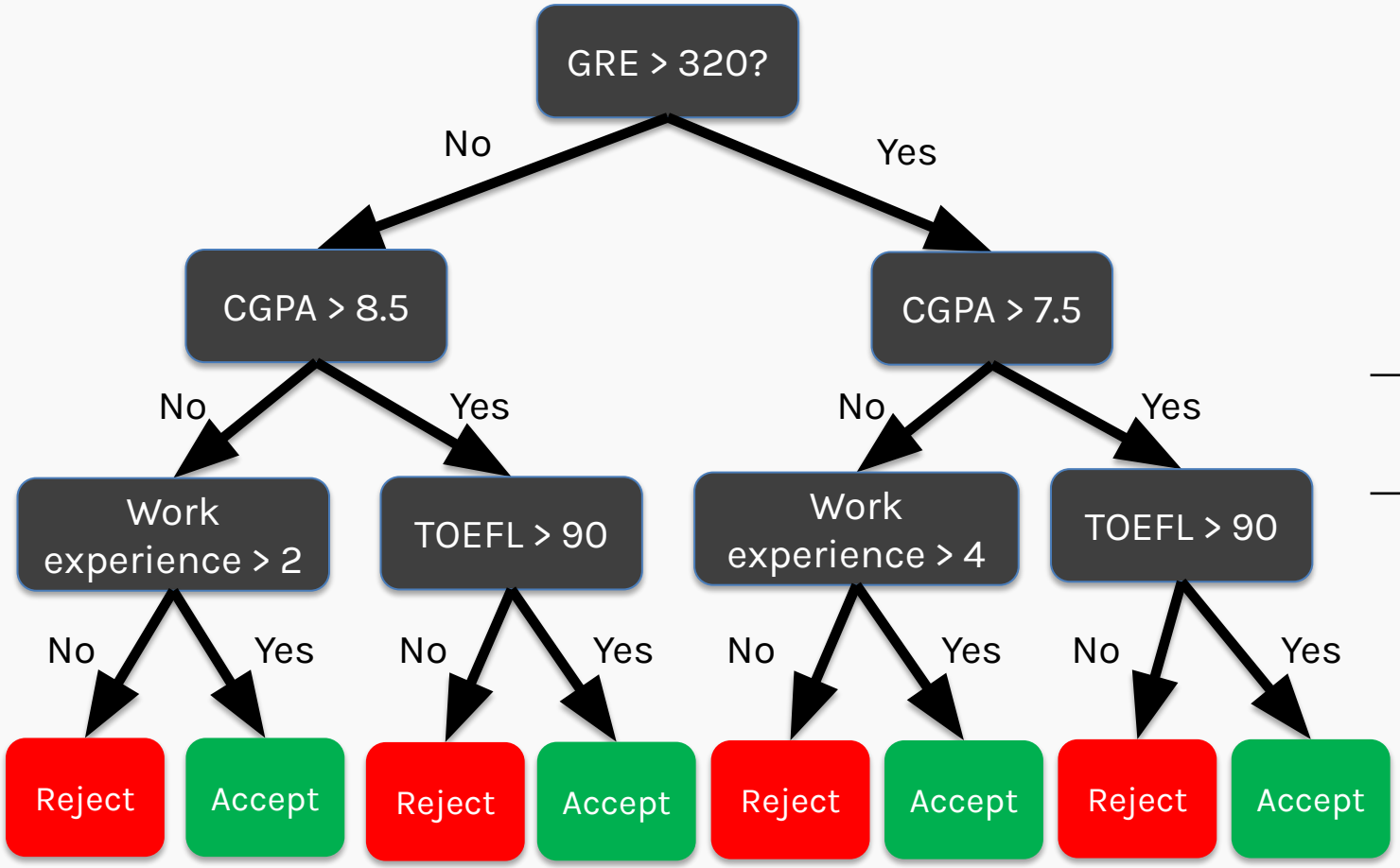
$$C(T) = Error(T) + \alpha\,|T|$$

Decision tree

Complexity Parameter

Number of leaves in the tree

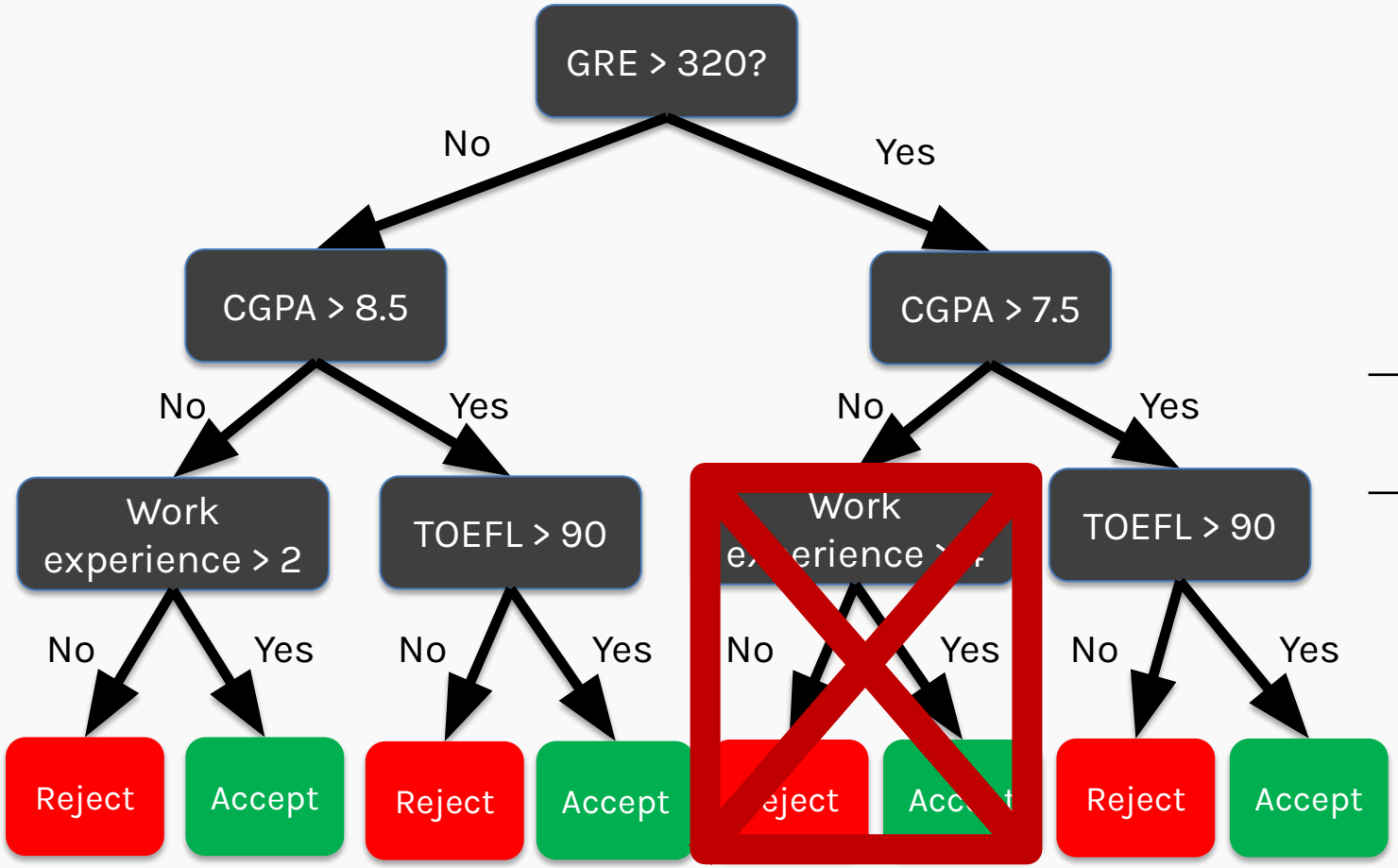
Pruning



$$\alpha = 0.2$$

Tree	Error(T)	| T |	
	0.32	8	0.32 + 0.2*8 = 1.92

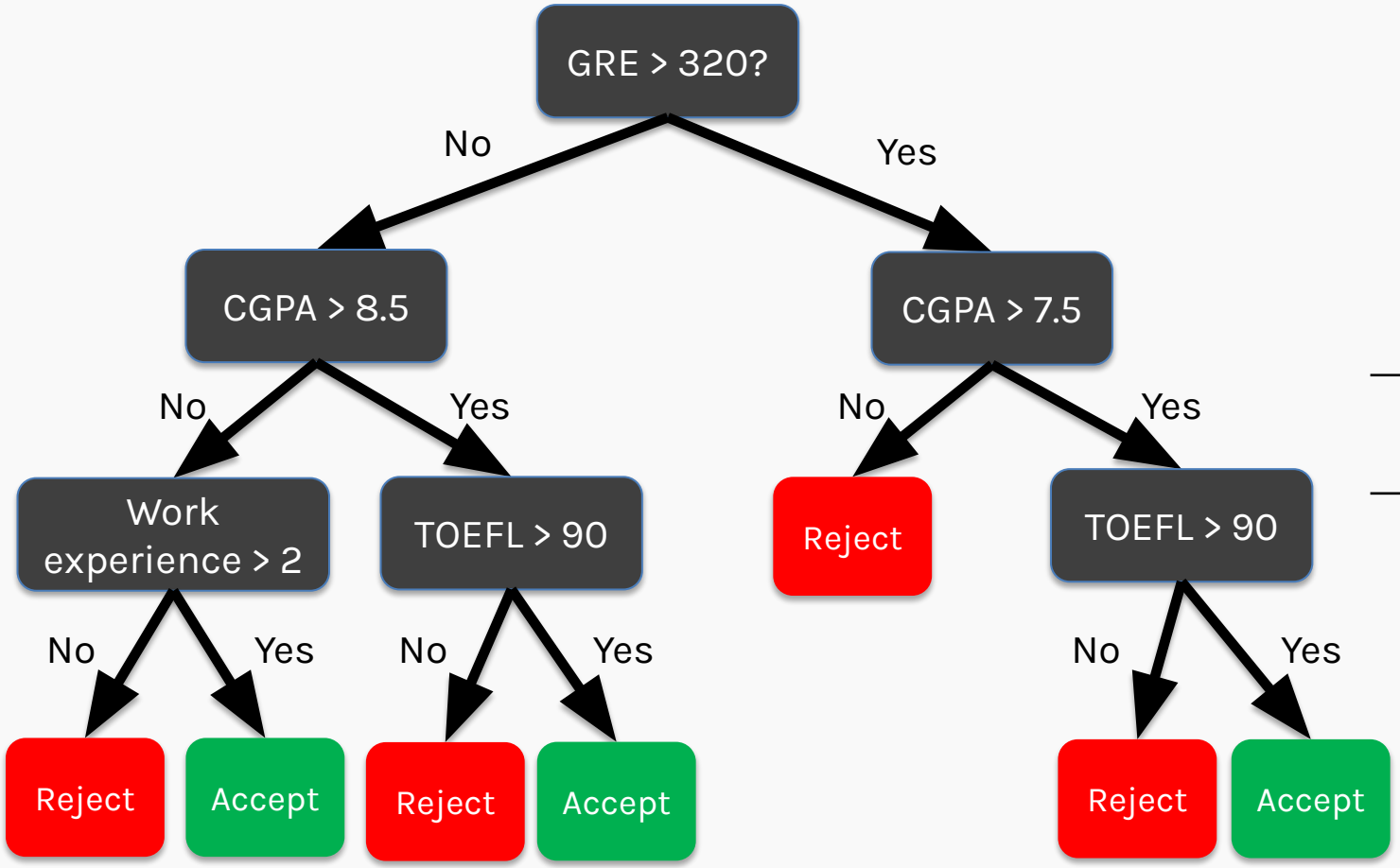
Pruning



$\alpha = 0.2$

Tree	Error(T)	| T |	
	0.32	8	1.92

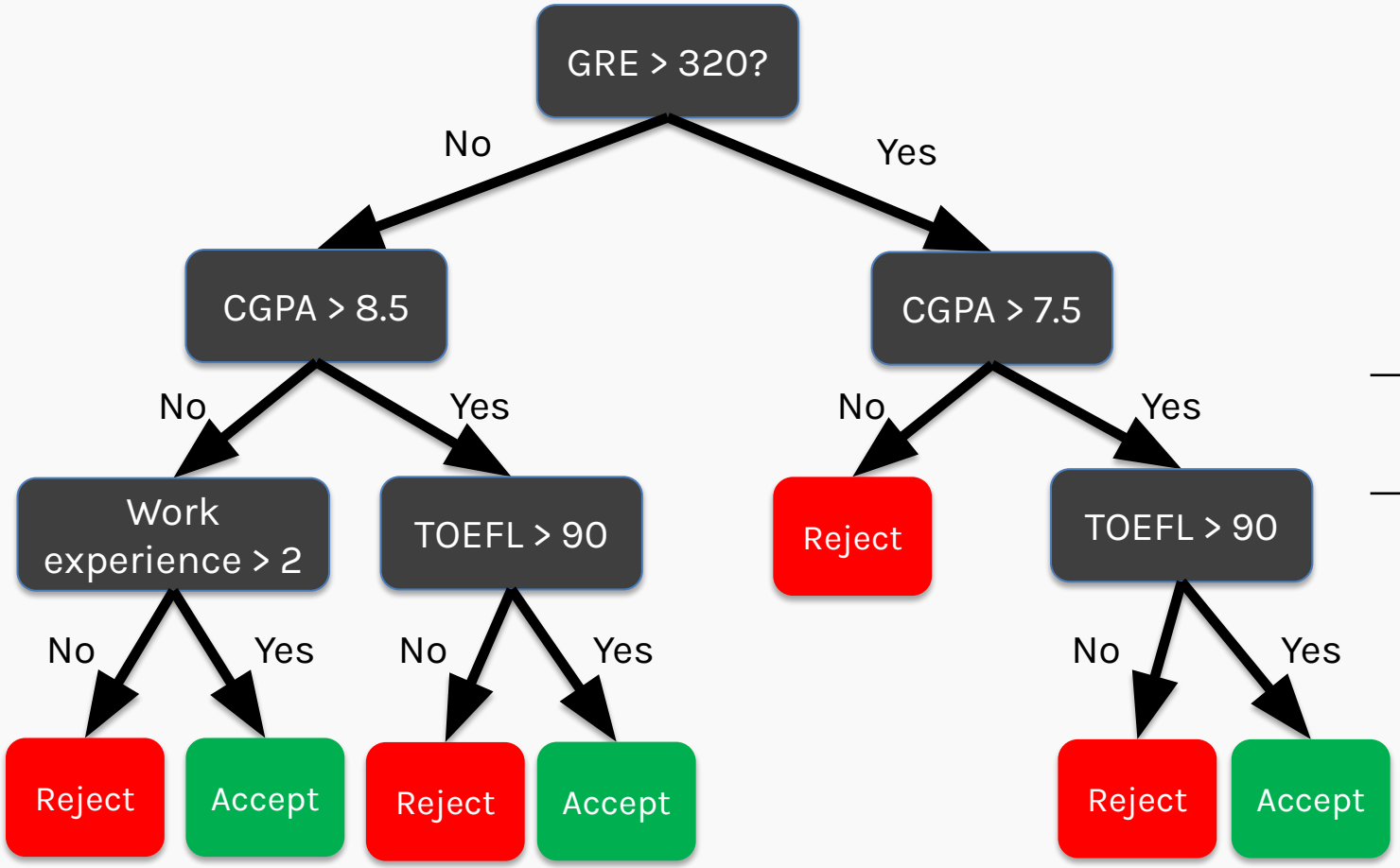
Pruning



$\alpha = 0.2$

Tree	Error(T)	| T |	
	0.32	8	1.92
	0.33	7	0.33 + 0.2*7 = 1.73

Pruning



$$\alpha = 0.2$$

Tree	Error(T)	| T |	
	0.32	8	1.92
	0.33	7	1.73

The smaller tree has a larger error `Error(T)` but smaller complexity score `C(T)` .

Pruning

Question: In the example, we set α = 0.2. But how do we find the best α?

$$C(T) = Error(T) + \alpha|T|$$

1. Fix α.

2. Find best tree for a given α and based on cost complexity C.

3. Find best α using CV (what should be the error measure?)

Pruning

The pruning algorithm:

1. Start with a full tree T_0 (each leaf node is pure)
2. Replace a subtree in T_0 with a leaf node to obtain a pruned tree T_1. This subtree should be selected to minimize

$$\frac{Error(T_0) - Error(T_1)}{|T_0| - |T_1|}$$

3. Iterate this pruning process to obtain $T_0, T_1, \dots, T_L$ where T_L is the tree containing just the root of T_0 . This should follow a bottom-up approach.
4. Select the optimal tree T_i by cross validation.

Note: You might wonder where we are computing the cost-complexity $C(T_l)$. One can prove that this process is equivalent to explicitly optimizing C at each step.