# Decision Trees

Rebecca Kurtz-Garcia

12/11/2021

# What is a decision tree?

A **decision tree** is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes.

# Decision Tree Examples

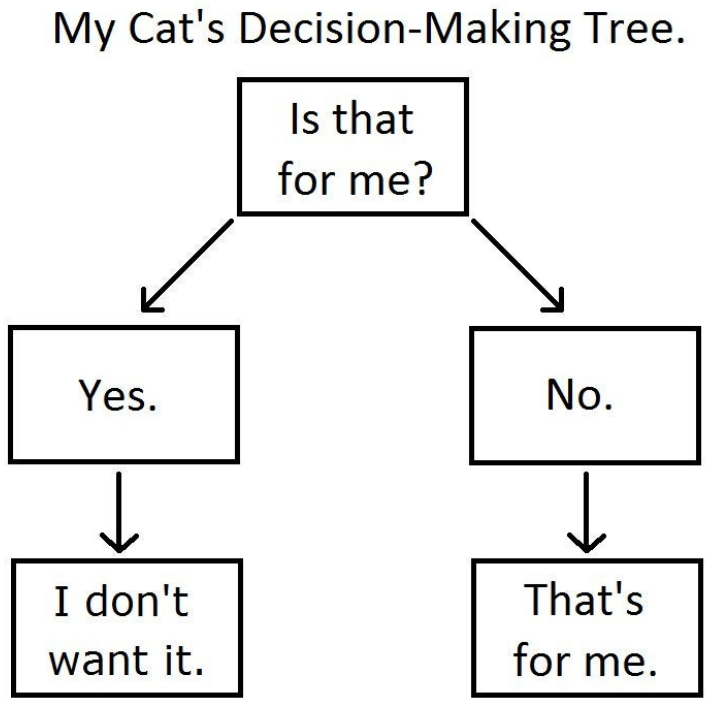



Figure 1: https://www.reddit.com/r/funny/comments/1j4gf7/cats_decision_tree/

# Decision Tree Examples

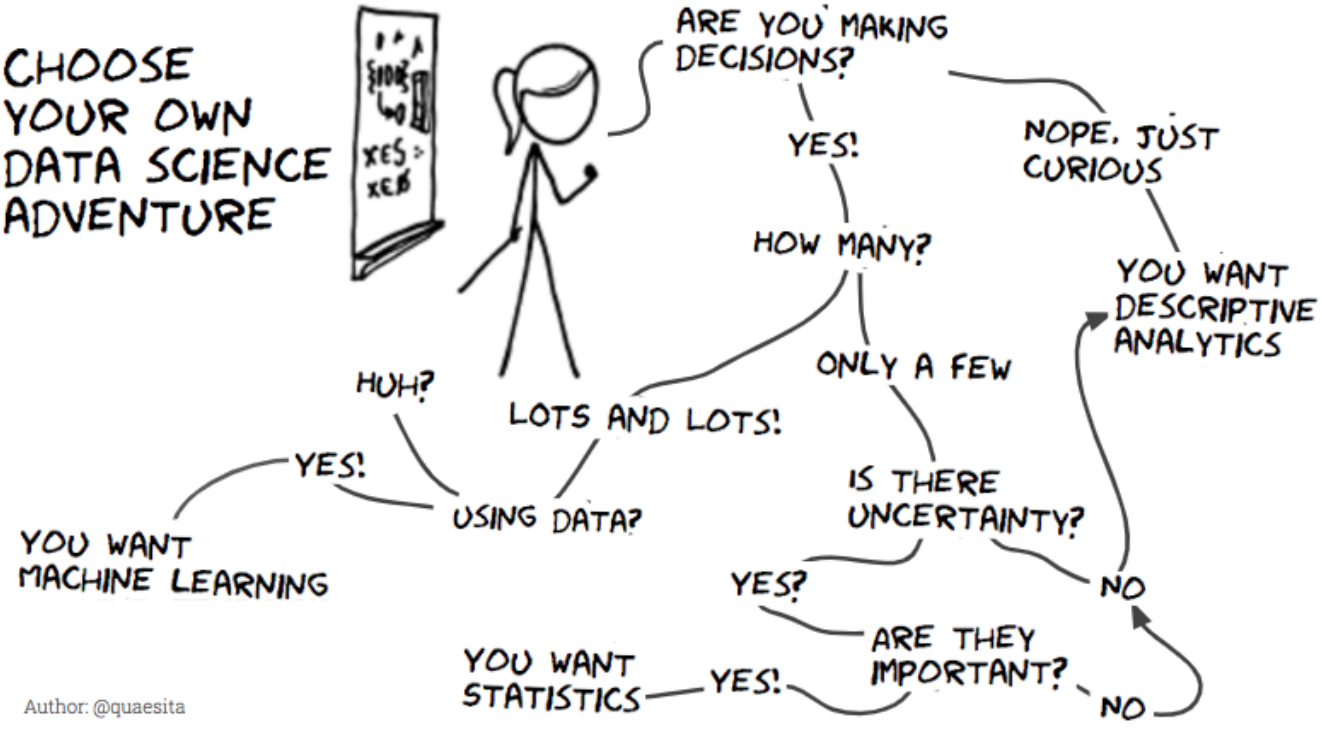



Figure 2: https://hackernoon.com/what-on-earth-is-data-science-eb1237d8cb37

# Decision Tree Examples

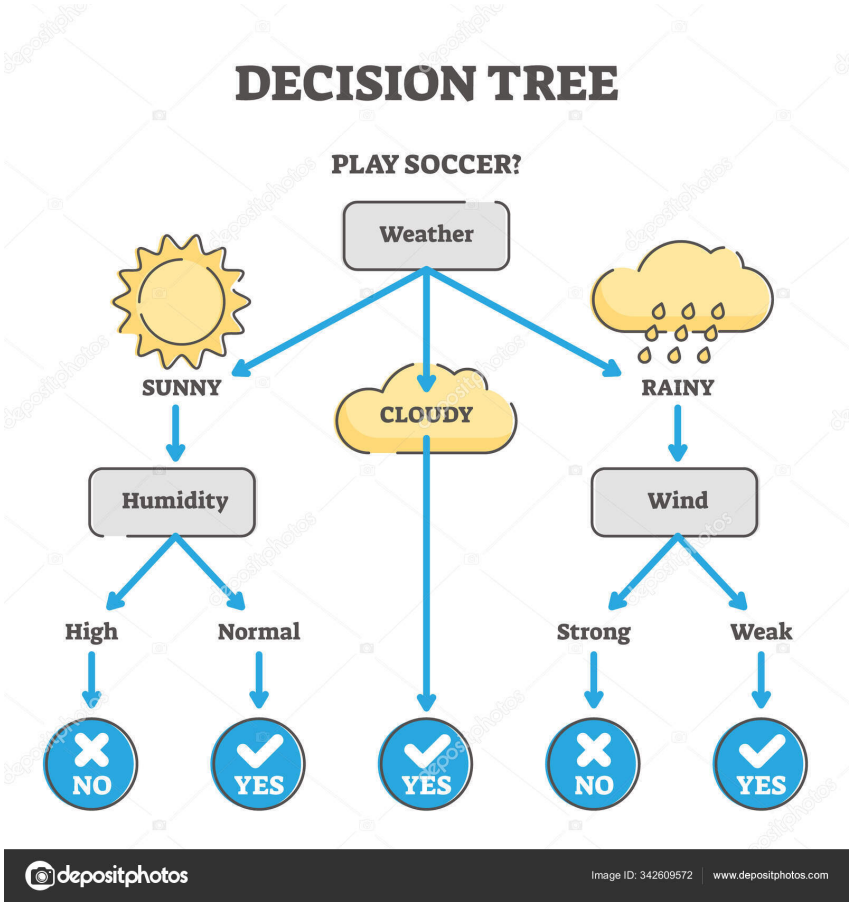



Figure 3: https://depositphotos.com/342609572/stock-illustration-decision-tree-example-diagram-vector.html

# Decision Tree Examples

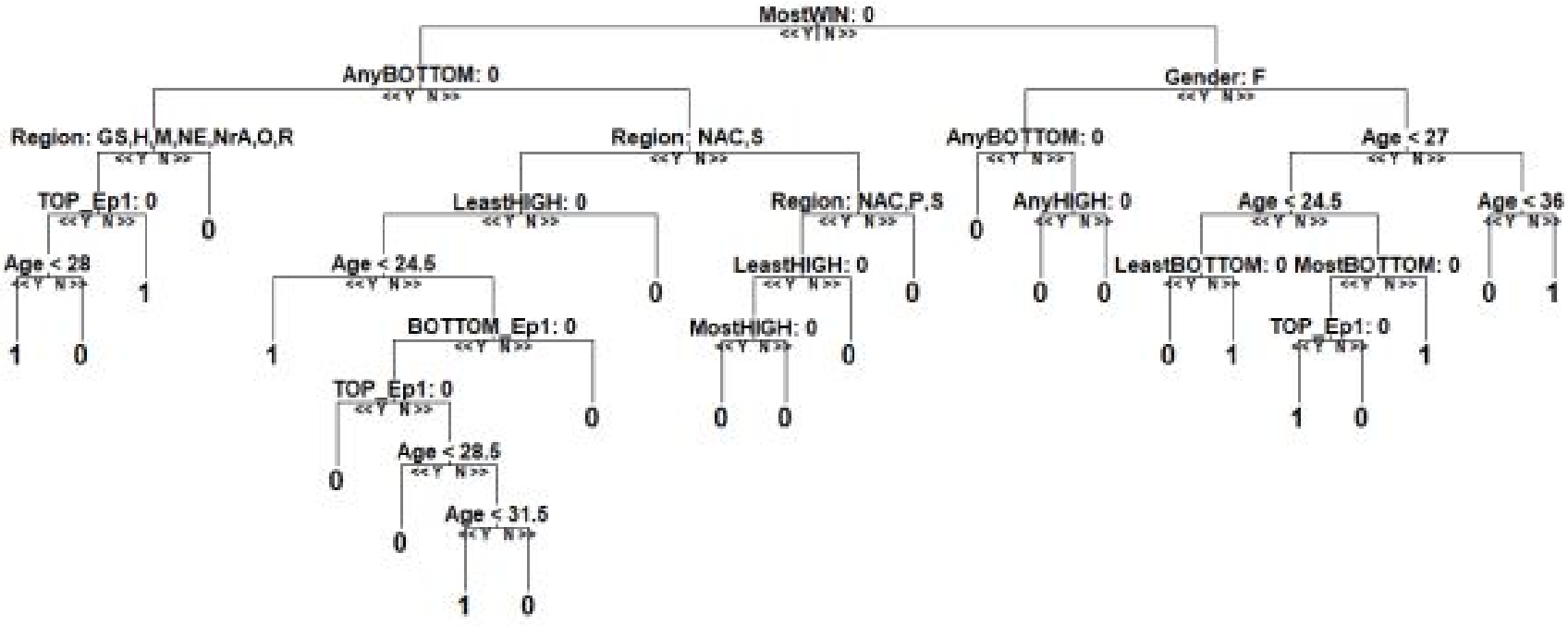



Figure 4: I made this one :)

# Data Types

- Before getting started on decision trees we need to review properties of data.
- In a typical data set each row corresponds to an *observation*, and each column refers to a *feature* or a *variable*.
- Features can either be **categorical** or **numeric**. We use a lot of other ways to describe features: qualitative vs quantitative, discrete vs continuous, etc.
  - Examples of categorical features: hair color, species, pet (cat, dog, fish)
  - Examples of numeric features: age, height, blood pressure
- We pick a categorical feature to be the focus of our analysis, or the **response** feature.

# Data Types

| Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species |
|---|---|---|---|---|
| 5.10 | 3.50 | 1.40 | 0.20 | setosa |
| 4.90 | 3.00 | 1.40 | 0.20 | setosa |
| 4.70 | 3.20 | 1.30 | 0.20 | setosa |
| 4.60 | 3.10 | 1.50 | 0.20 | setosa |
| 5.00 | 3.60 | 1.40 | 0.20 | setosa |
| 5.40 | 3.90 | 1.70 | 0.40 | setosa |

# Decision Trees

- In most of our examples we made tree splits based on logic and outside knowledge.
- When we have a lot of data (or do not know much about our data) we want to find a impartial splitting rule based on mathematical properties.
- We use **splitting rules** to segment the predictor space. These splitting rules can be summarized using graphs that look like trees.

# Decision Trees

- So the goal is to separate the data into different species variables:

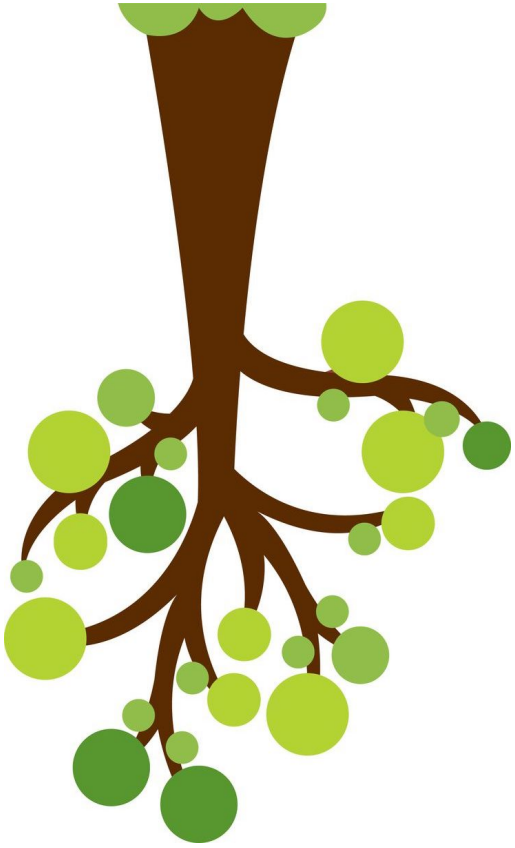
  `Sepal.Length`, `Sepal.Width`, `Petal.Length`, and `Petal.Width`.

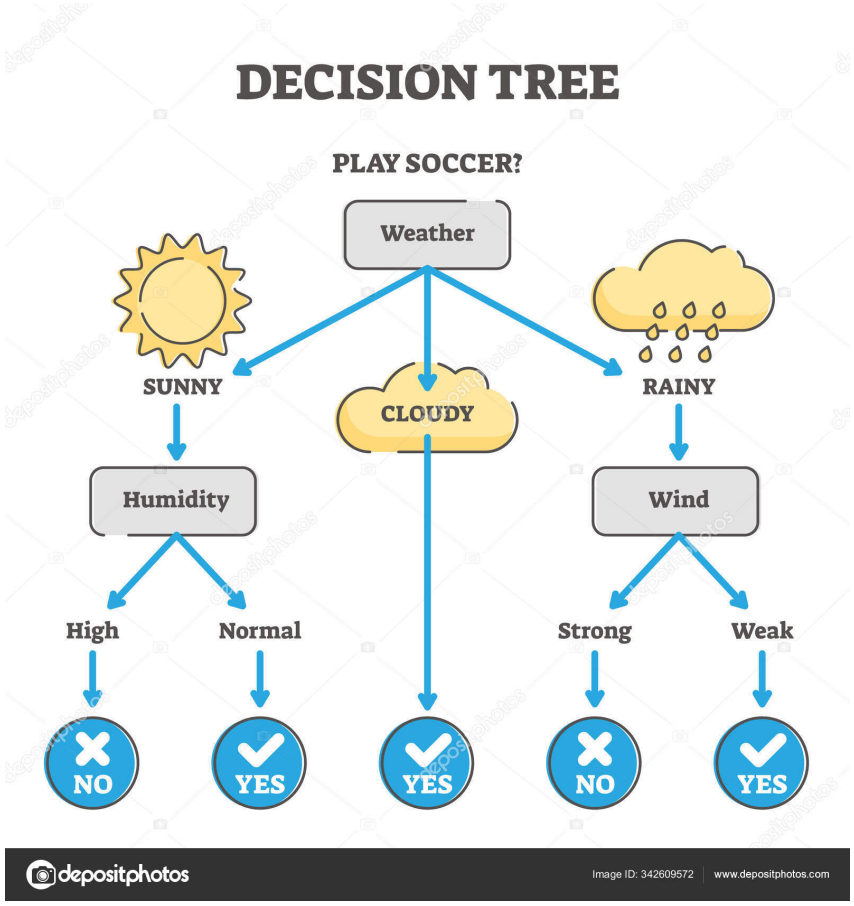
- Why would we want to build a tree?
  - To *predict* what type of species we have for a new unknown flower
  - To *assess* the relationships between features of a data set.

    i.e. How are sepals, petals, and species all related?

# Decision Trees

- The graphs we create look like trees.
- The final regions are known as **terminal nodes** or **leaves**
- Trees are typically drawn upside down.





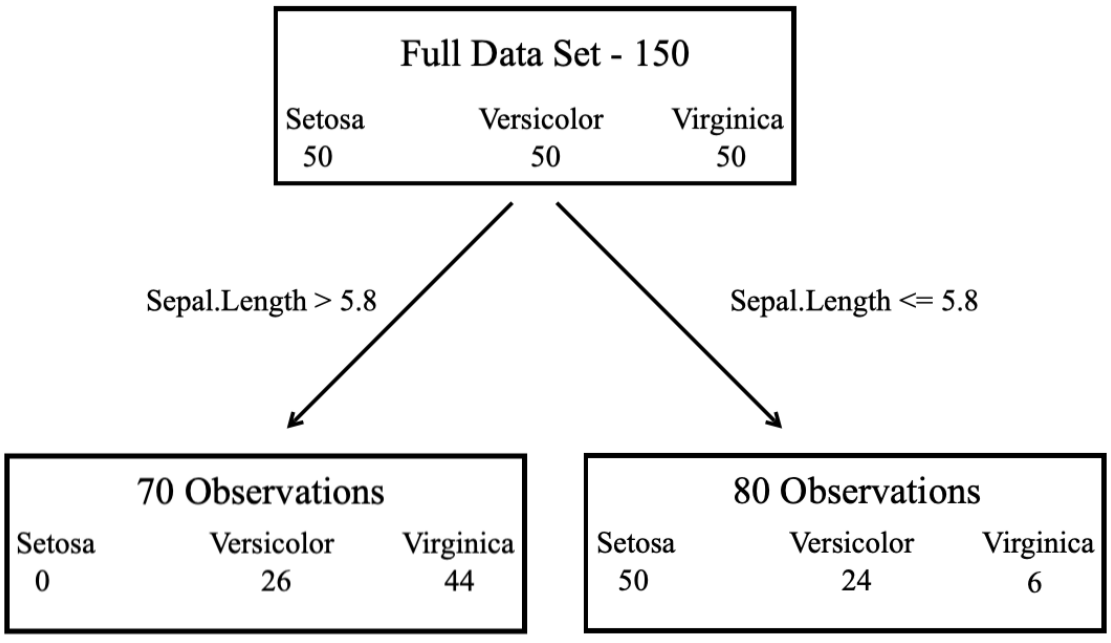
# Decision Trees

- Can we always create a decision tree? Not necessarily.
- We must assume that all observations are **independent**. That is, we don't repeatedly sample an observations, have time series data, etc.

# Making Predictions

- We make predictions based on the data that is in the terminal nodes and the feature we are interested in predicting.
- When we are interested in predicting a categorical variable we use most common value (the mode).

# Growing a Tree




| Sepal.Length | ... | Speices |
|---|---|---|
| 5.1 | | Setosa |
| 4.9 | | Setosa |
| 5.0 | | Setosa |
| . | | . |
| . | | . |
| . | | . |
| 5.7 | | Virginica |
| 6.2 | | Virginica |
| 5.9 | | Virginica |

# A few notes about nodes

- Terminal nodes are usually not *pure*, meaning the values in the terminal node are mixed.
- Values in terminal nodes are repeated.

# Growing a Tree

- How do we grow a good tree?
- It is usually unfeasible to consider every possible partition.
- We typically use a **top-down/greedy/recursive-binary-splitting** approach.
  - At each split take the best possible split at that particular point.

# The Best Split

- What criteria is used to determine the best split? Depends on the type of response variable.
- For categorical data: **error rate**, entropy, information, gini impurity, chi-square

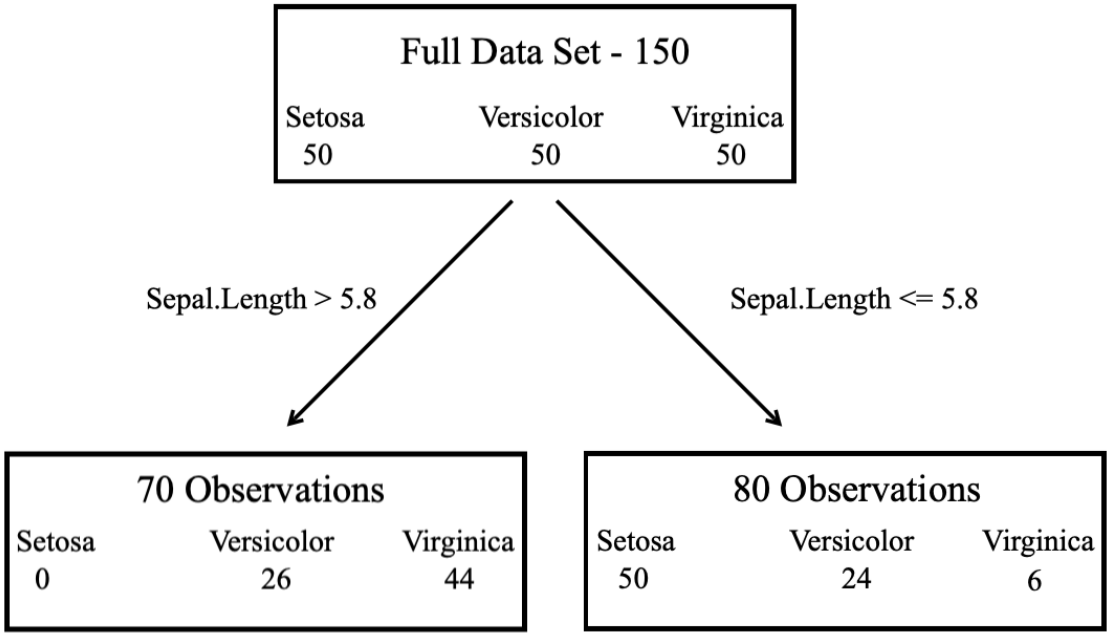
# Growing a Tree

- We can quantify what best separates the data by seeing what would by asking ourselves: If we stopped splitting the data right now, what would be the *error rate (E)* ?

$$E = \frac{\text{Number of Observations Incorrectly Predicted}}{\text{Total Amount of Data}}$$

- We consider every possible split at every opportunity and always pick the one with the lowest error rate.
- This is one of the reasons why this technique become popular when computers became more readily available. Considering every possible type of split is computationally difficult.

# Growing a Tree




| Sepal.Length | ... | Speices |
|---|---|---|
| 5.1 | | Setosa |
| 4.9 | | Setosa |
| 5.0 | | Setosa |
| . | | . |
| . | | . |
| . | | . |
| 5.7 | | Virginica |
| 6.2 | | Virginica |
| 5.9 | | Virginica |

What would be the predicted values if we stopped here?

# Growing a Tree

| Sepal.Length | ... | Speices | Predicted |
|---|---|---|---|
| 5.1 | | Setosa | Setosa |
| 4.9 | | Setosa | Setosa |
| 5.0 | | Setosa | Setosa |
| . | | . | |
| . | | . | |
| . | | . | |
| 5.7 | | Virginica | Setosa |
| 6.2 | | Virginica | Virginica |
| 5.9 | | Virginica | Virginica |

If we stopped now and looked at the classification **error rate**, the error rate would be 37.33%.

# Growing a Tree

- We could continue splitting or growing our tree until...
  - all our nodes our pure, if possible.
  - there are no more possible splits.
- This practice often leads to **overfitting** our data.

# Growing a Tree

- Overfitting occurs when your model or tree becomes too specific to your data set, and not the overall population.
- The tree will not be relevant to new observations that were not used to build the tree.

# Stunting

- There are several techniques to prevent overfitting. The one we will focus on is stunting.
- When growing a tree in real life you can stunt its growth in various ways to prevent it from growing to big. We can also do this with decision trees!
- Recall, a decision tree always tries to do the best possible separation. Subsequent splits tend to be less important for separating the data.
- So we only make our tree stop growing once it reaches a maximum number of splits.

# Stunting

- More ways we could *stunt* a tree:
  - Only split if it results a specific amount of error reduction.
  - Only split if there are a minimum number of observations in the resulting node.
  - Only split if there are a minimum number of observations in the current node.
  - Only consider a maximum number of splits no matter what.
  - etc.

# Stunting

- We "stunt" decision trees to prevent overfitting, however if we are too restrictive we could result in *underfitting*.
- Underfitting occurs when our model/tree is simple and missing important information.
- To determine if we have a good fit with the right balance we can evaluate our model using a *training* set and a *test* set.

# Stunting

- It is very difficult to accurately assess a model using the same data you used to build the model. This typically leads to inaccurate results.
- To assess how well your model is preforming you need to test it with observations that you did not consider when building your model.
- When you first get a large data set you should separate your data set into two parts: **test set** (20%), and **training set** (80%).

# Test Error vs Training Error

- The test data set should be used to assess the model.
- We want to *stunt* our decision tree in order to get the least complicated model that gives us good results.
- This part of the analysis is not an exact science, use your judgment.

# How do we know if we have a good tree?

- One of the most important ways to evaluate your model is by error rate. Was your model better than random guessing (50%).
- You should also investigate where the errors occurred, and what type of error was the most common.
- It is also important to highlight any interesting trends or relationships in your model.

# General Steps

1) Establish which variables will be used for splitting, and which variable are we interested in predicting (the response variable). Is our data *independent*?
2) Divide the data into two parts, a *test set* (20%) and a training set (80%).
3) Build your tree. Consider a stunting method and look at the *test error* to determine how you should stunt your tree.
4) Evaluate and interpret your tree.

# Step 1 & 2: Set up

Step 1:

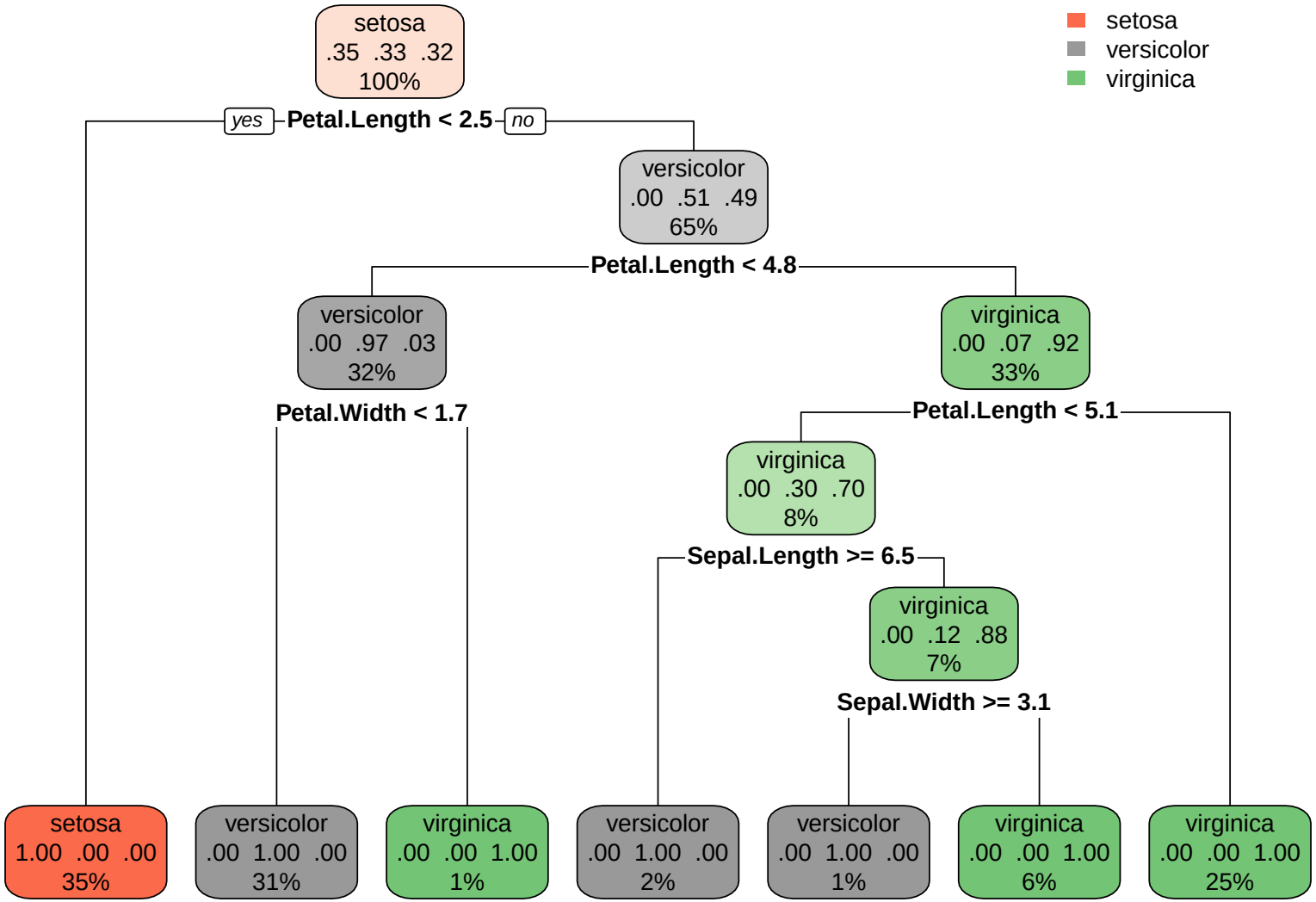
- The response variable is `Species`. It is safe to assume the data is independent.

Step 2:

- 30 observations are randomly selected to be in the test set.
- 120 observations are used to build the tree.

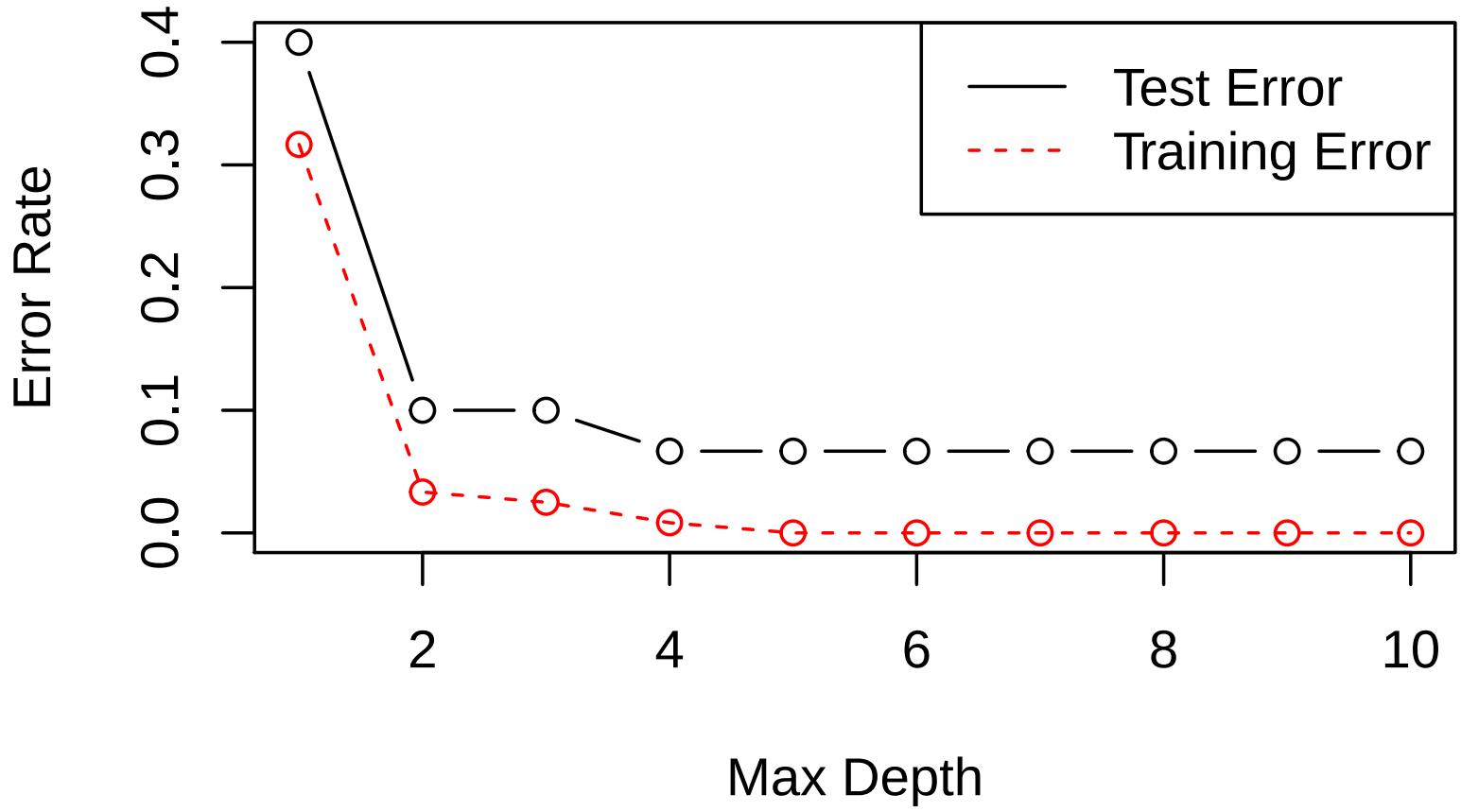
# Step 3: Build tree

Now lets build a decision tree using the training set. Let's see how big it will be if we do not stunt the tree.



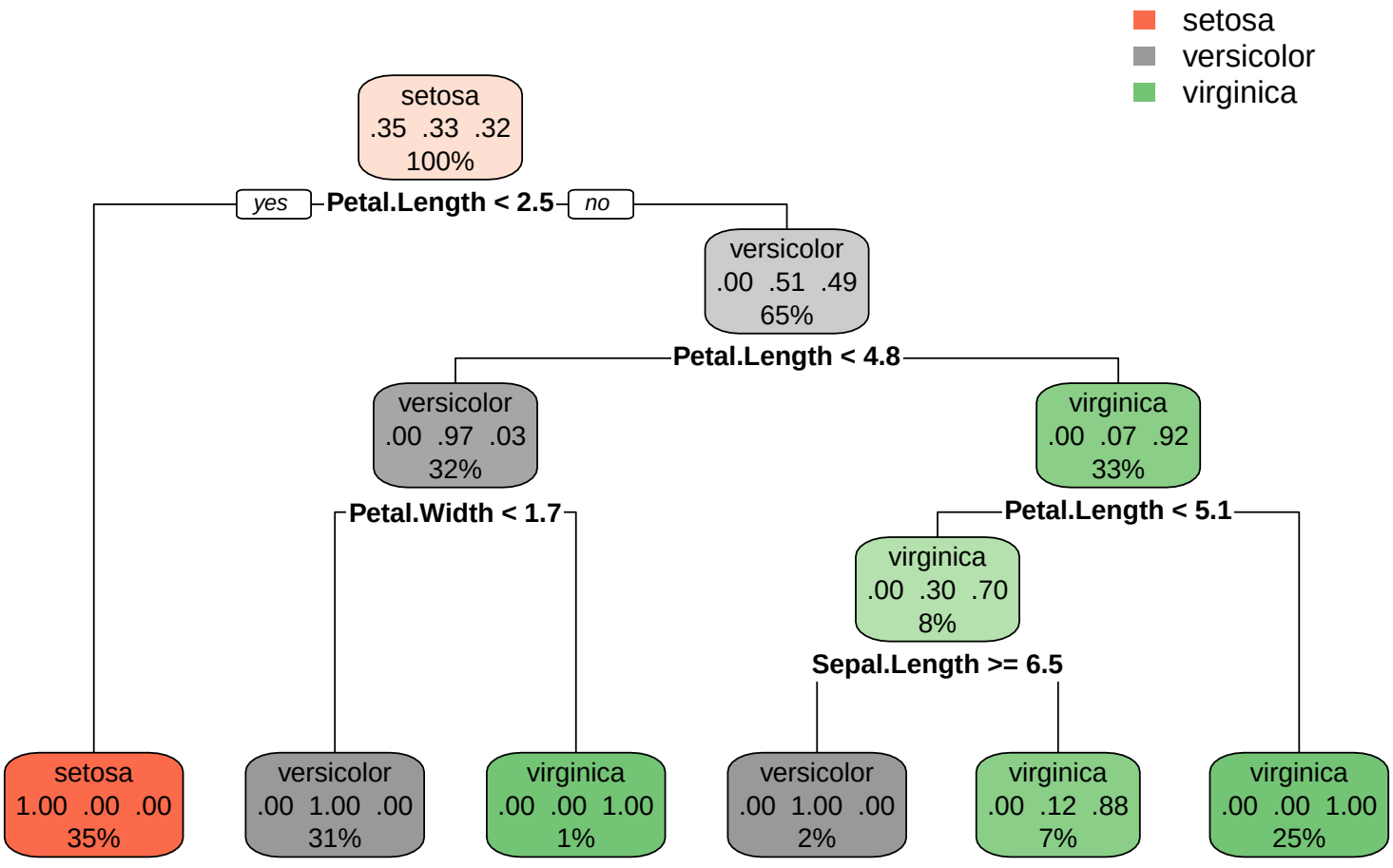
# Step 3: Build Tree

Compare training error and test error for different levels of stunting.



# Step 3: Build Tree

Max depth = 4




The test error rate is 6.67%

# Step 4: Evaluate and Interpret

- What can we do with our tree?
- If we get a new flower with the petal and sepal information we can use this to predict what species the flower is.
- We can also learn about general trends in the data!
  - Setosa tends to have small petal lengths, dramatically so compared to the others.
  - Versicolor and Virginica are very similar, it is harder to tell the difference between them.
  - Petal Length appears to be the most important variable, it appeared the most.
  - Sepal width does not appear to be a distinguishing feature compared to the others.

# Decision Tree PROs

- Can be applied to lot of different types of data. They are very flexible and very few assumptions.
- Easy to read once they are created.
- Intuitive explanation of variable relationships.
- The methods described here can be extended to very complex settings.

# Decision Tree CONs

- Although we can make decision tree for many situations, some are situations are far more difficult to make a decision tree for (time series!).
- Deciding how much to prune a tree or which is the best split is not always clear.
- Not as mathematically rigorous as other methods.