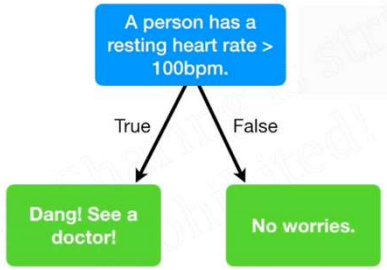
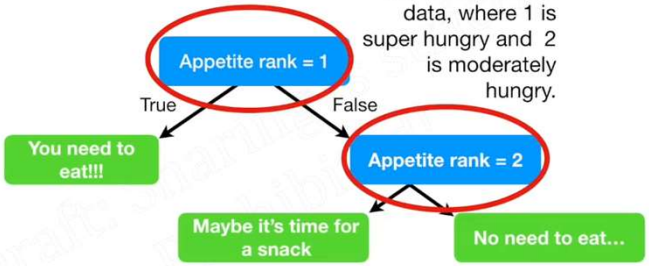
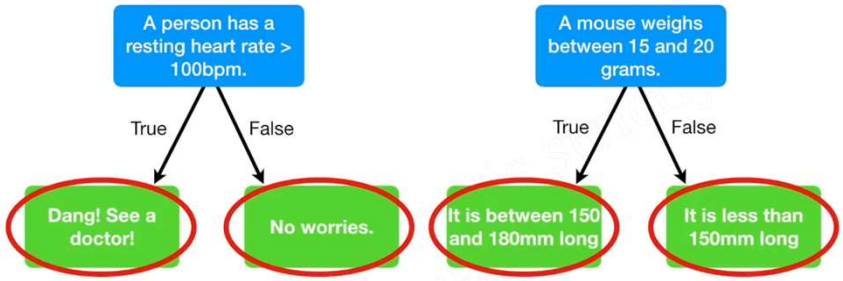
DECISION TREES
&
RANDOM FORESTS



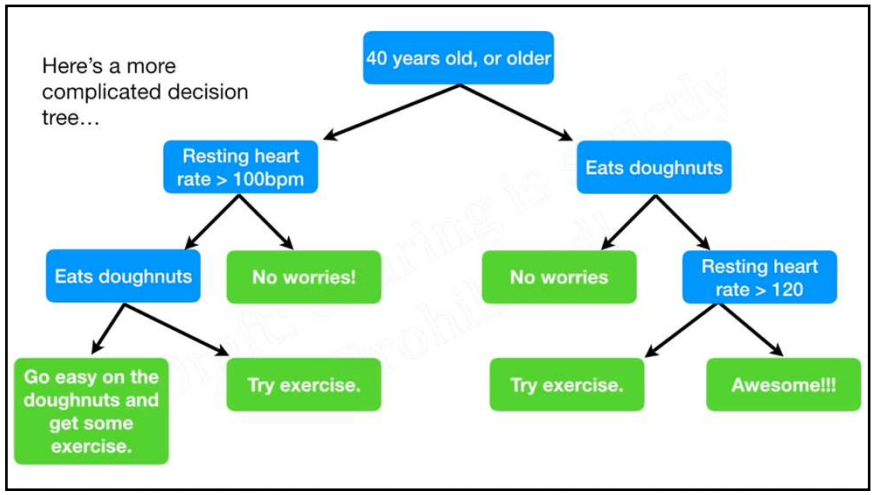
This decision tree is based on **ranked** data, where 1 is super hungry and 2 is moderately hungry.

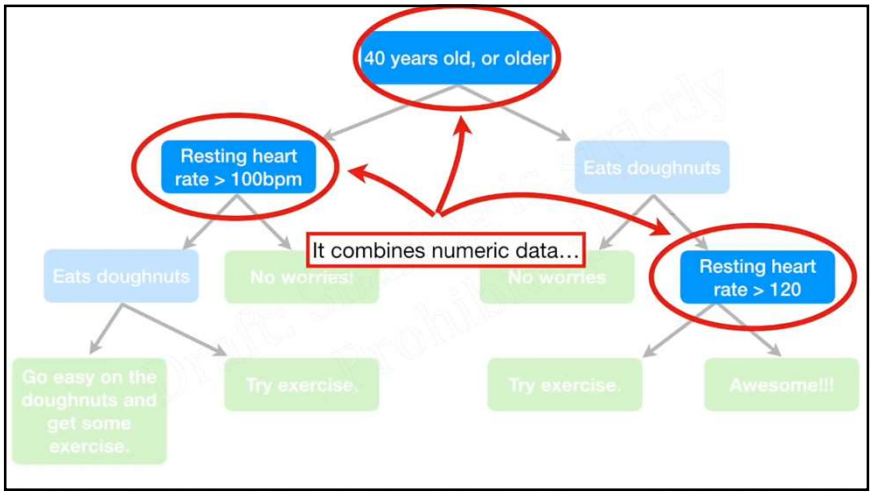


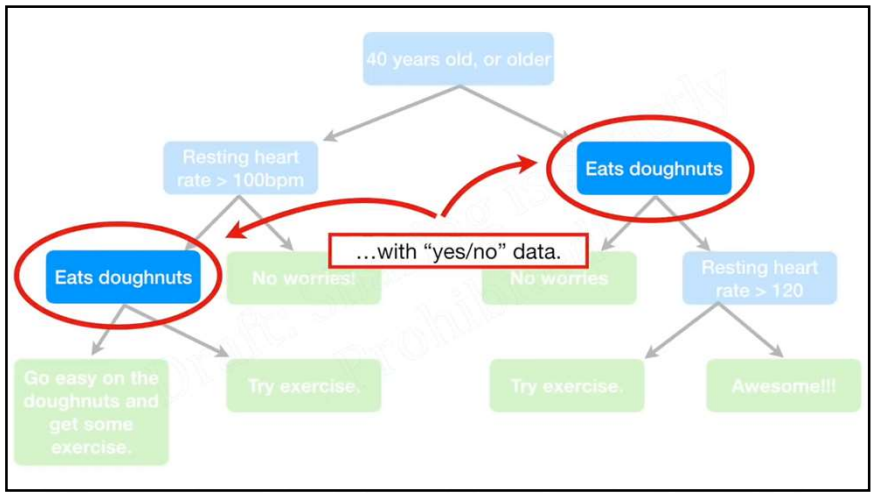


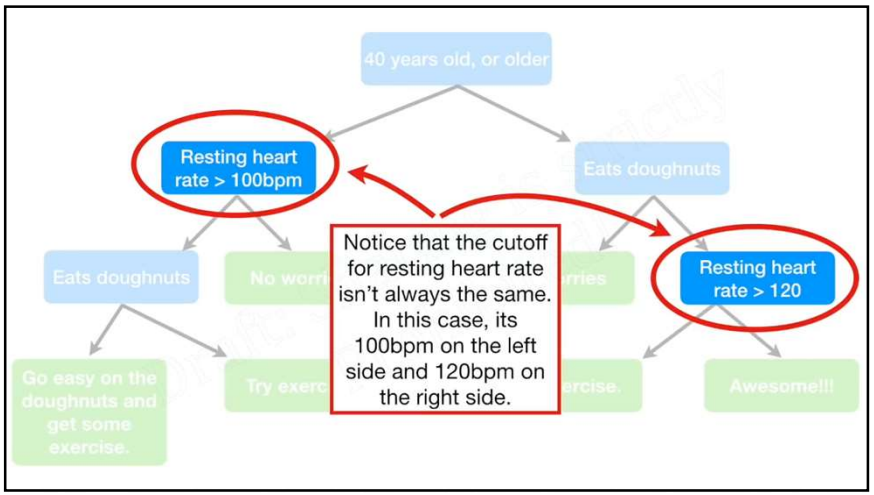
NOTE: The classification can be categories…

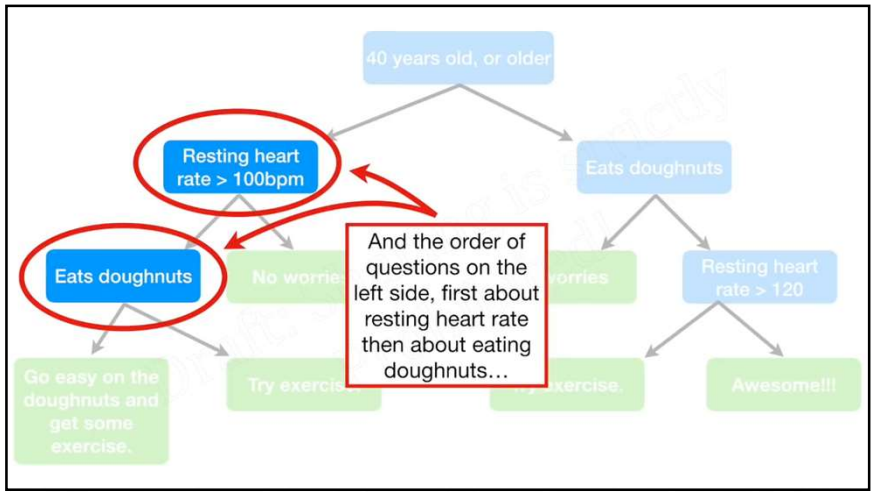
…or numeric.

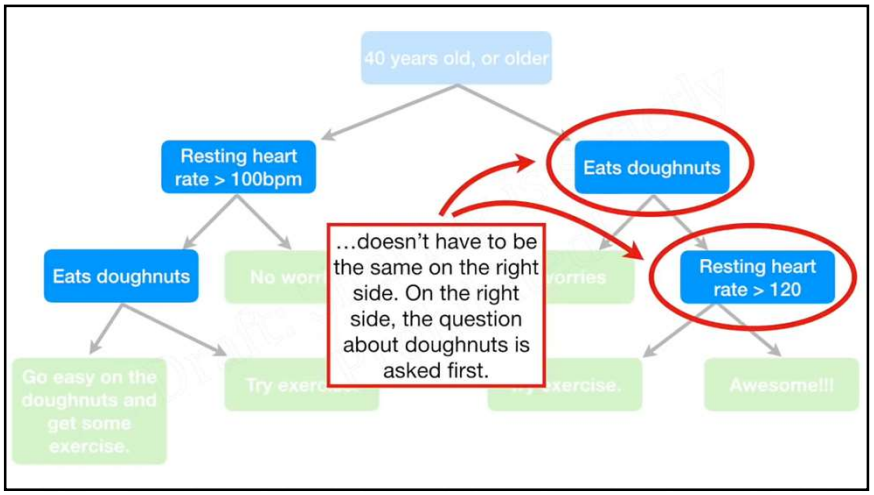

Here's a more complicated decision tree…
40 years old, or older
Resting heart rate > 100bpm
Eats doughnuts
Eats doughnuts
No worries!
No worries
Resting heart rate > 120
Go easy on the doughnuts and get some exercise.
Try exercise.
Try exercise.
Awesome!!!


40 years old, or older
Resting heart rate > 100bpm
Eats doughnuts
It combines numeric data…
Resting heart rate > 120


40 years old, or older
Resting heart rate > 100bpm
Eats doughnuts
…with "yes/no" data.
Eats doughnuts
Resting heart rate > 120


40 years old, or older
Resting heart rate > 100bpm
Eats doughnuts
Notice that the cutoff for resting heart rate isn't always the same. In this case, its 100bpm on the left side and 120bpm on the right side.
Resting heart rate > 120

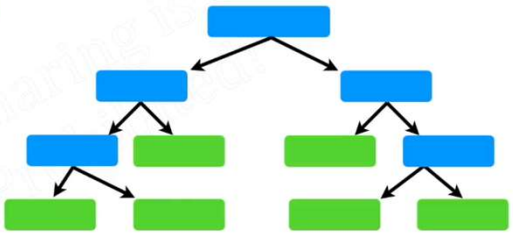




Now we are ready to talk about how to go from a raw table of data…

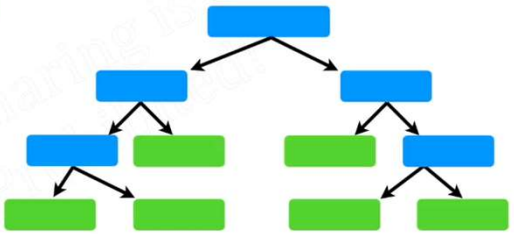
…to a decision tree!!!

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…



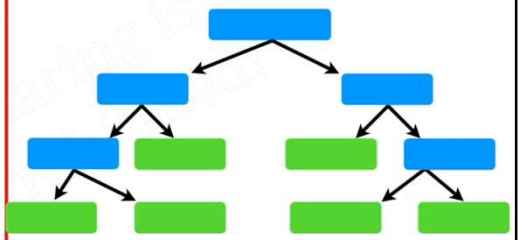
In this example, we want to create a tree that uses **chest pain**, **good blood circulation** and **blocked artery status** to predict…

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…



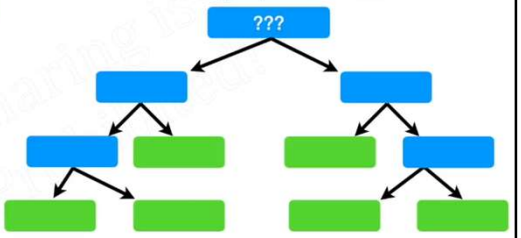
…whether or not a patient has heart disease.

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…



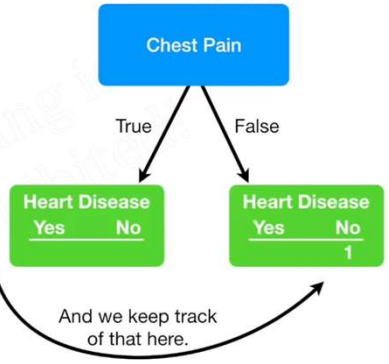
The first thing we want to know is whether **Chest Pain**, **Good Blood Circulation** or **Blocked Arteries** should be at the very top of our tree.

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…



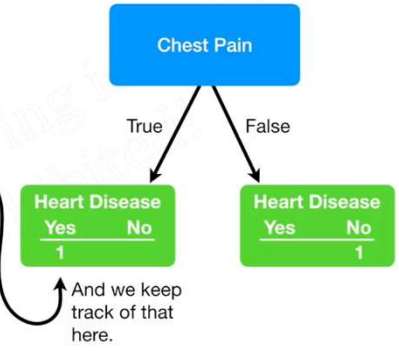
The first patient does not have chest pain and does not have heart disease.

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…



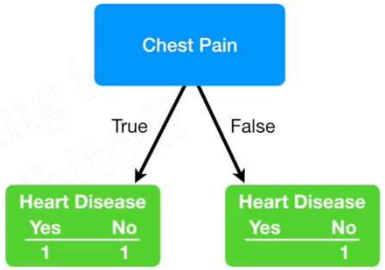
The 2nd patient has chest pain and heart disease.

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…

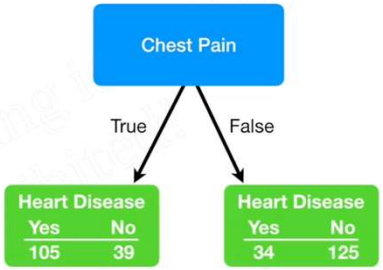


The 4th patient has chest pain and heart disease.

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…



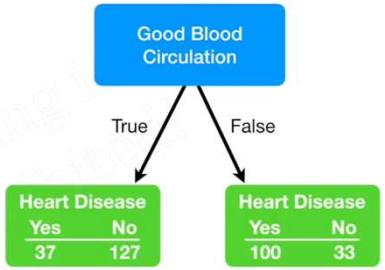
Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…



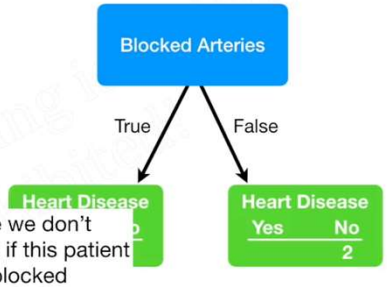
Ultimately, we look at chest pain and heart disease for all 303 patients in this study.

Now we do the exact same thing for **Good Blood Circulation**.

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…

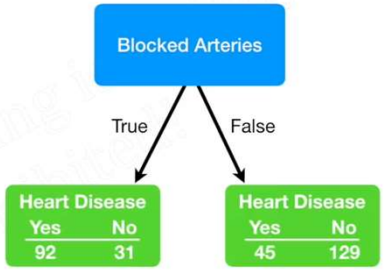


Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…



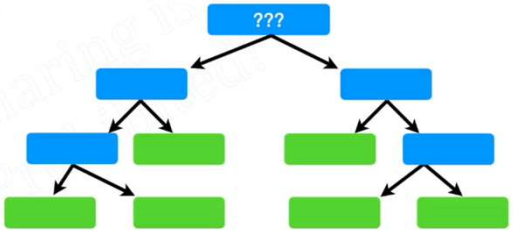
Since we don't know if this patient had blocked arteries or not, we'll skip it.

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…



Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…

Remember the goal is to decide whether **Chest Pain**, **Good Blood Circulation** or **Blocked Arteries** should be the first thing in our decision tree (aka **The Root Node**).



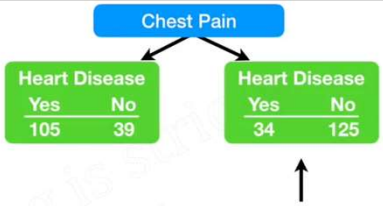
Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…

So we looked at how well **Chest Pain** separated patients with and without heart disease.

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…

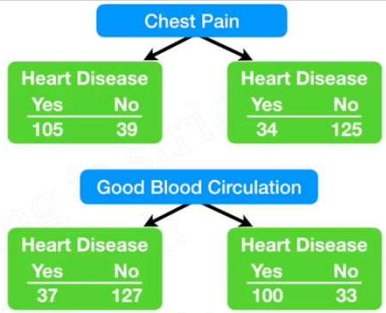
Most of the patients with heart disease ended up in this leaf node…

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…



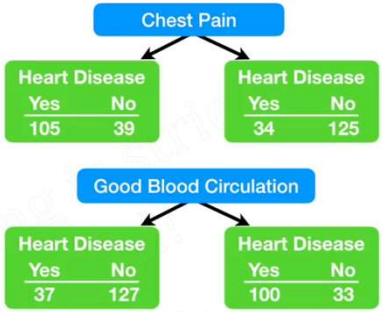
… and most of the patients without heart disease ended up in this leaf node.

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…



Then we looked at how well **Good Blood Circulation** separated patients with and without heart disease.

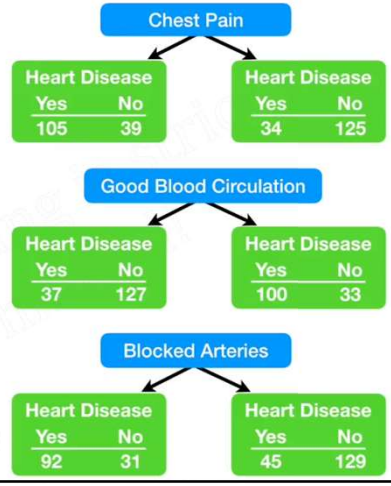
Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc…	etc…	etc…	etc…



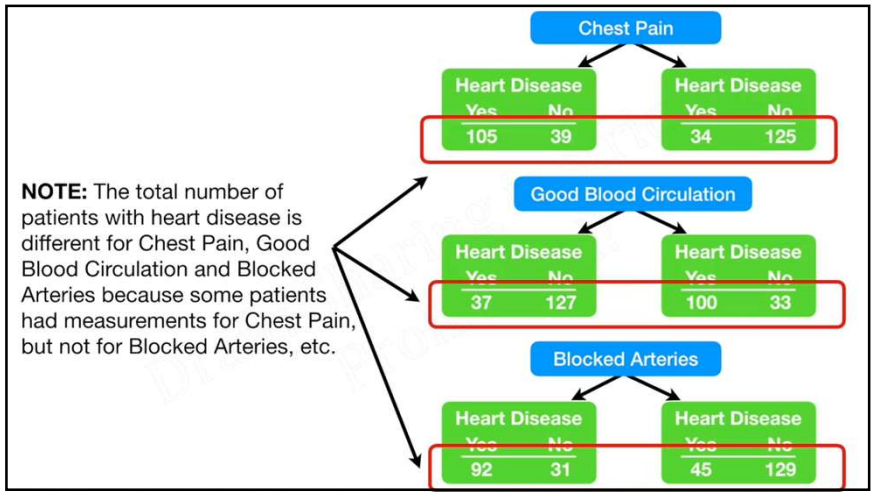
Then we looked at how well **Good Blood Circulation** separated patients with and without heart disease.

It wasn't perfect either.

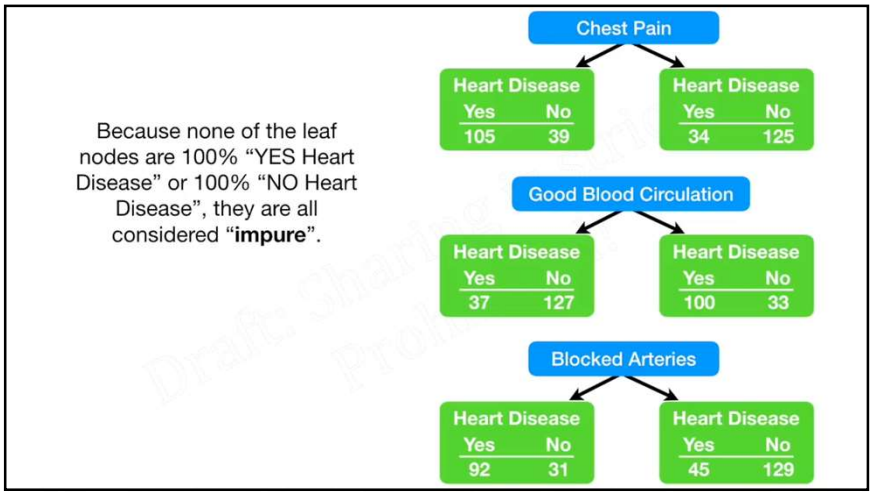
Note:
Yes = 139
No = 164
Total = 303



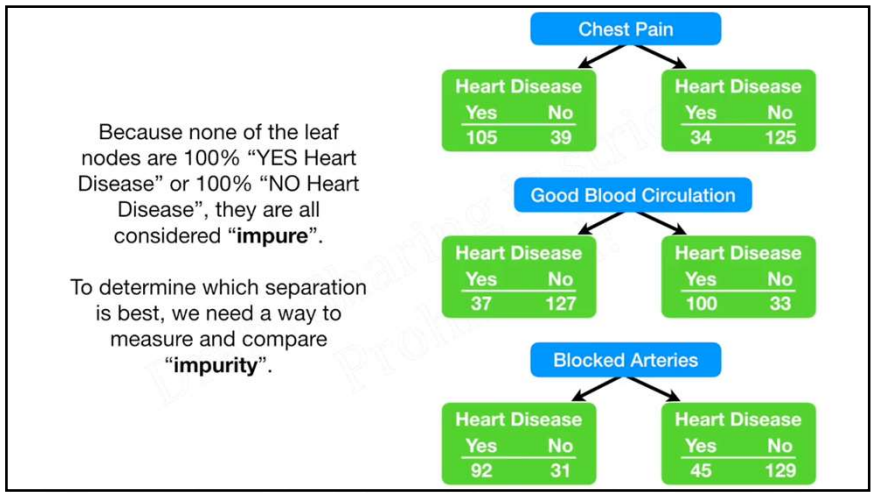
Lastly, we looked at how well **Blocked Arteries** separated patients with and without heart disease.



NOTE: The total number of patients with heart disease is different for Chest Pain, Good Blood Circulation and Blocked Arteries because some patients had measurements for Chest Pain, but not for Blocked Arteries, etc.



Because none of the leaf nodes are 100% "YES Heart Disease" or 100% "NO Heart Disease", they are all considered "**impure**".



Because none of the leaf nodes are 100% "YES Heart Disease" or 100% "NO Heart Disease", they are all considered "**impure**".

To determine which separation is best, we need a way to measure and compare "**impurity**".

IMPURITY MEASUREMENT (GINI and ENTROPY)

Gini

The **gini impurity** is calculated using the following formula:

$$GiniIndex = 1 - \sum_j p_j^2$$

Where p_j is the probability of class j.

$$Gini_{min} = 1 - (1^2) = 0$$

$$Gini_{max} = 1 - (0.5^2 + 0.5^2) = 0.5$$

Gini Index has values inside the interval [0, 0.5] whereas the interval of the Entropy is [0, 1].

Entropy

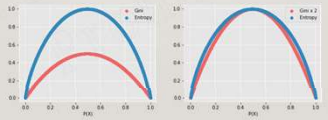
The **entropy** is calculated using the following formula:

$$Entropy = -\sum_j p_j \cdot log_2 \cdot p_j$$

Where, as before, p_j is the probability of class j.

$$Entropy_{min} = -1 \cdot log_2(1) = 0$$

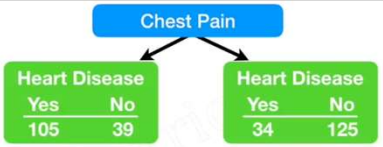
$$Entropy_{max} = -0.5 \cdot log_2(0.5) - 0.5 \cdot log_2(0.5) = 1$$



Computationally, entropy makes use of logarithm, so the calculation of the Gini Index will be faster.

Text and Image Courtesy: https://quantdare.com/decision-trees-gini-vs-entropy/

Let's start by calculating Gini impurity for Chest Pain…



For this leaf, the Gini impurity = 1 - (the probability of "yes")^2 - (the probability of "no")^2

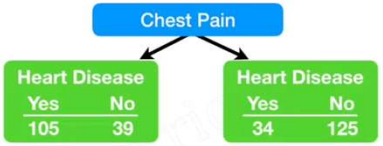
$$= 1 - \left(\frac{105}{105 + 39}\right)^2 - \left(\frac{39}{105 + 39}\right)^2$$

$$= 0.395$$

Draft. Sharing is Strictly Prohibited!

Gini impurity = 0.395

Draft. Sharing is Strictly Prohibited!

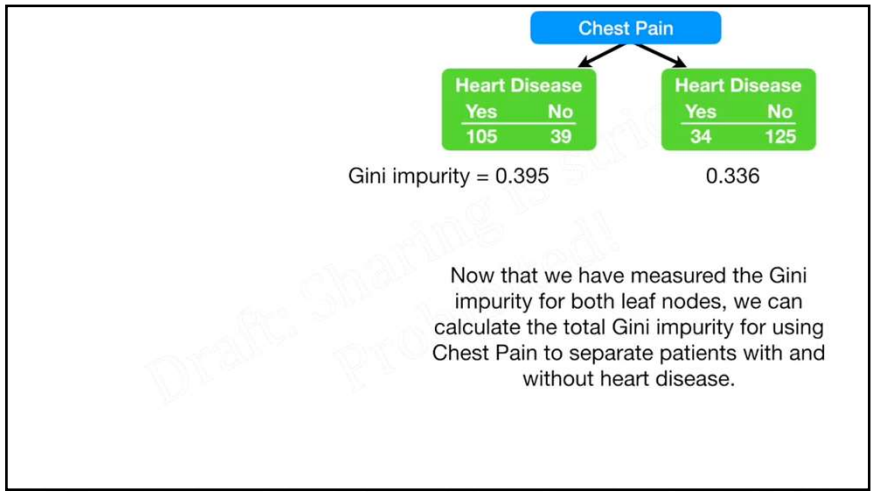


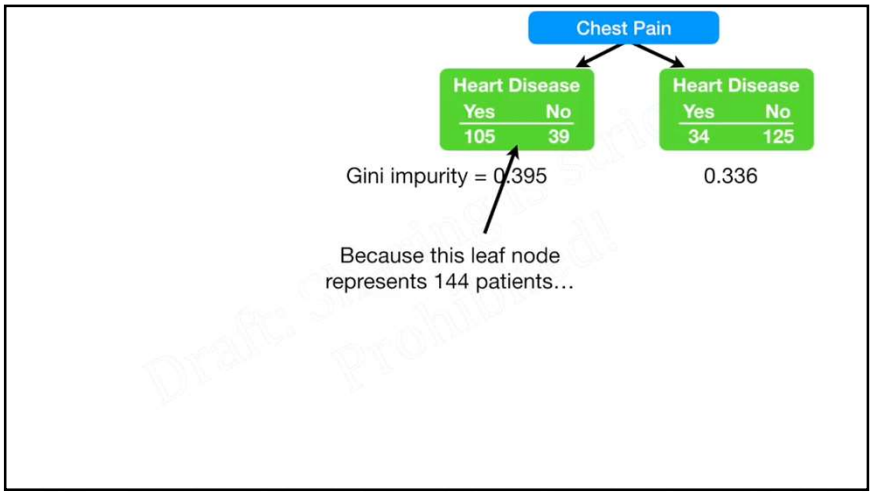
= 1 - (the probability of "yes")^2 - (the probability of "no")^2

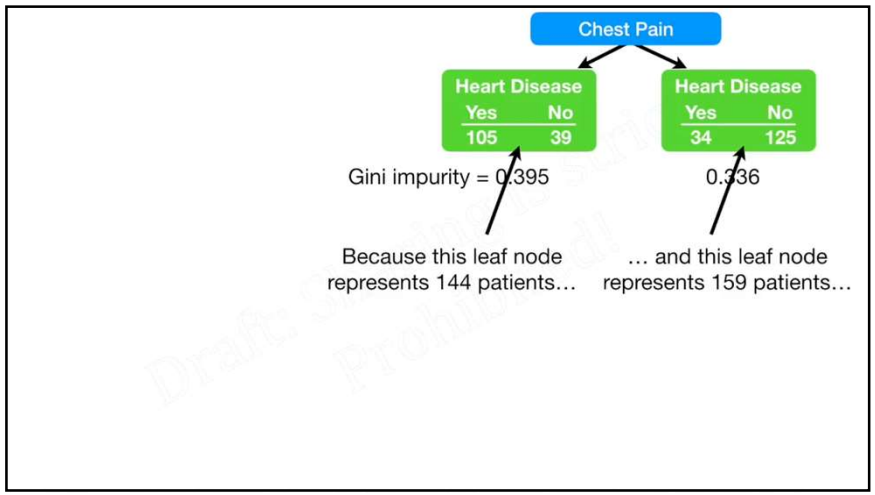
$$= 1 - \left(\frac{34}{34 + 125}\right)^2 - \left(\frac{125}{34 + 125}\right)^2$$

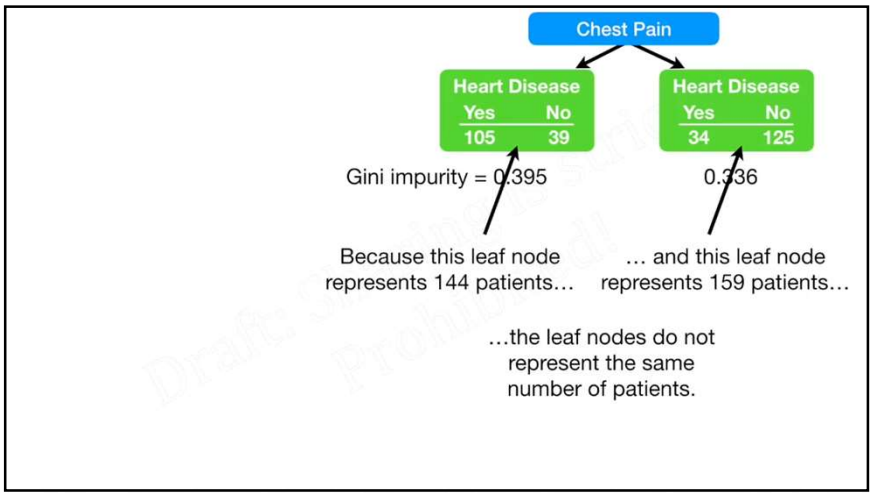
$$= 0.336$$

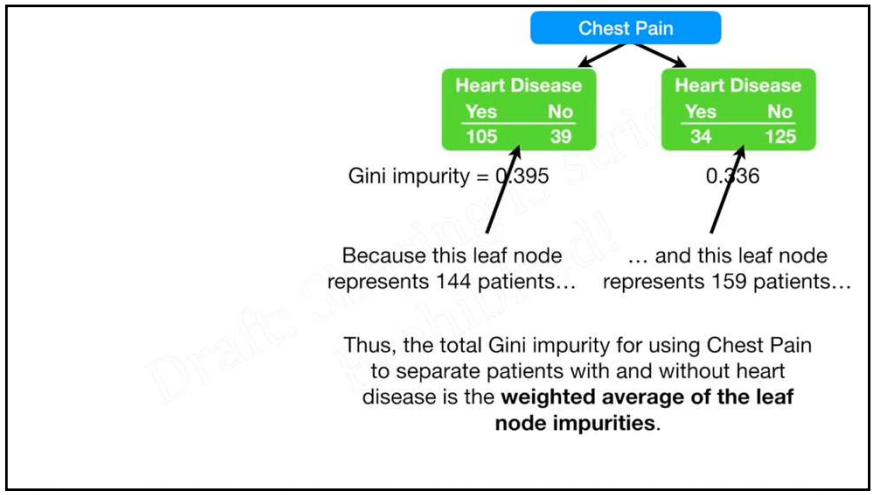
Draft. Sharing is Strictly Prohibited!

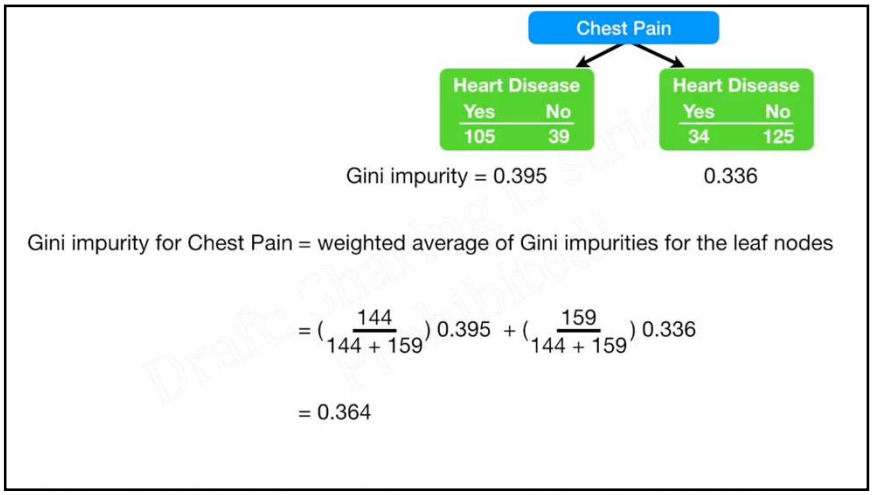

Chest Pain
Heart Disease
Yes No
105 39
Heart Disease
Yes No
34 125
Gini impurity = 0.395
0.336
Now that we have measured the Gini impurity for both leaf nodes, we can calculate the total Gini impurity for using Chest Pain to separate patients with and without heart disease.

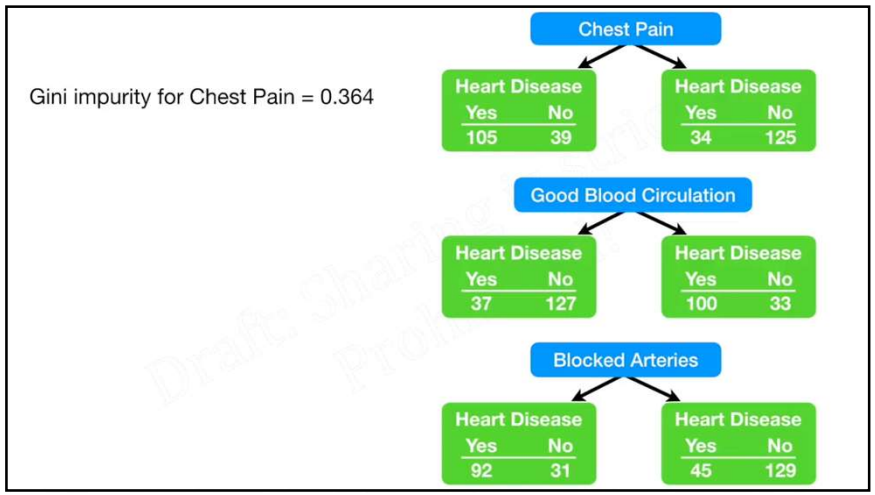

Chest Pain
Heart Disease
Yes No
105 39
Heart Disease
Yes No
34 125
Gini impurity = 0.395
0.336
Because this leaf node represents 144 patients…

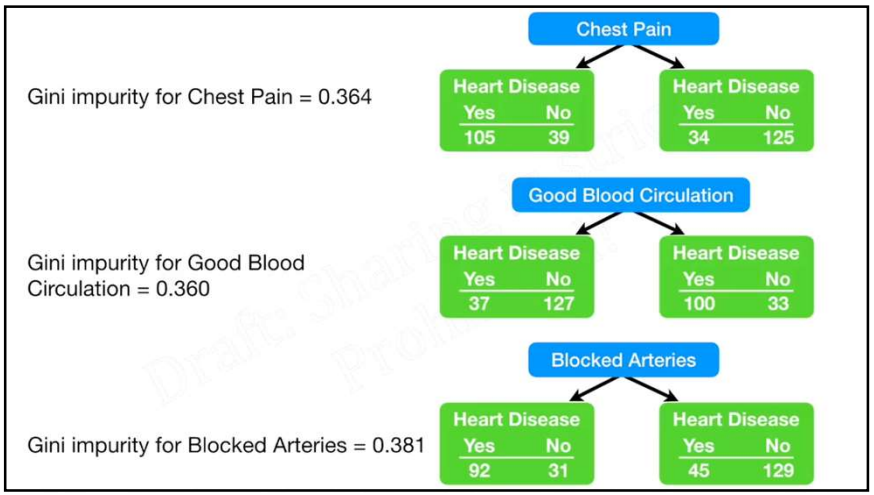

Chest Pain
Heart Disease
Yes No
105 39
Heart Disease
Yes No
34 125
Gini impurity = 0.395
0.336
Because this leaf node represents 144 patients…
… and this leaf node represents 159 patients…


Chest Pain
Heart Disease
Yes No
105 39
Heart Disease
Yes No
34 125
Gini impurity = 0.395
0.336
Because this leaf node represents 144 patients…
… and this leaf node represents 159 patients…
…the leaf nodes do not represent the same number of patients.


Chest Pain
Heart Disease
Yes No
105 39
Heart Disease
Yes No
34 125
Gini impurity = 0.395
0.336
Because this leaf node represents 144 patients…
… and this leaf node represents 159 patients…
Thus, the total Gini impurity for using Chest Pain to separate patients with and without heart disease is the **weighted average of the leaf node impurities**.


Chest Pain
Heart Disease
Yes No
105 39
Heart Disease
Yes No
34 125
Gini impurity = 0.395
0.336
Gini impurity for Chest Pain = weighted average of Gini impurities for the leaf nodes
$= \left(\frac{144}{144 + 159}\right) 0.395 + \left(\frac{159}{144 + 159}\right) 0.336$
$= 0.364$


Gini impurity for Chest Pain = 0.364
Chest Pain
Heart Disease
Yes No
105 39
Heart Disease
Yes No
34 125
Good Blood Circulation
Heart Disease
Yes No
37 127
Heart Disease
Yes No
100 33
Blocked Arteries
Heart Disease
Yes No
92 31
Heart Disease
Yes No
45 129


Gini impurity for Chest Pain = 0.364
Chest Pain
Heart Disease
Yes No
105 39
Heart Disease
Yes No
34 125
Gini impurity for Good Blood Circulation = 0.360
Good Blood Circulation
Heart Disease
Yes No
37 127
Heart Disease
Yes No
100 33
Gini impurity for Blocked Arteries = 0.381
Blocked Arteries
Heart Disease
Yes No
92 31
Heart Disease
Yes No
45 129

Gini impurity for Chest Pain = 0.364

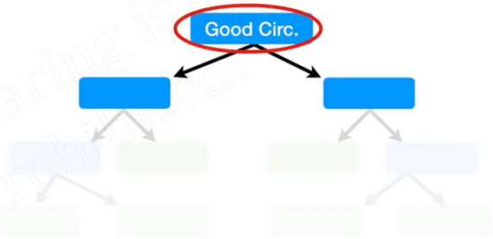
Gini impurity for Good Blood Circulation = 0.360

Good Blood Circulation has the lowest impurity (it separates patients with and without heart disease the best)…

Gini impurity for Blocked Arteries = 0.381

Gini impurity for Chest Pain = 0.364

…so we will use it at the root of the tree.



Gini impurity for Good Blood Circulation = 0.360

Gini impurity for Blocked Arteries = 0.381

Gini impurity for Chest Pain = 0.364

Information gain = impurity reduction

Note:
Yes = 139
No = 164
Total = 303

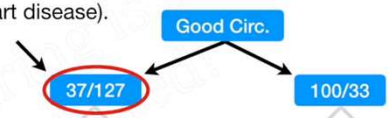
Initial Impurity in the data set was
= 1 - ((139/(139+164))^2 - ((164/(139+164))^2
= 1 - 0.210 - 0.293 = .497

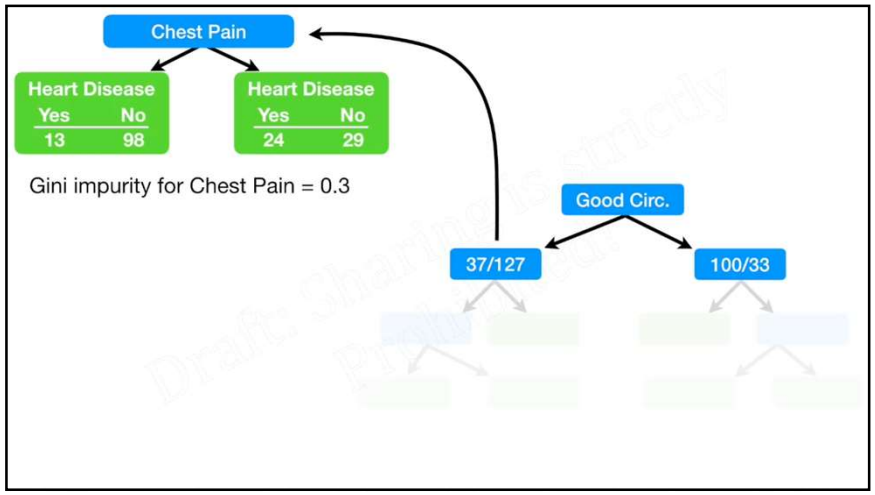
Information gain = .497-.360 = .137

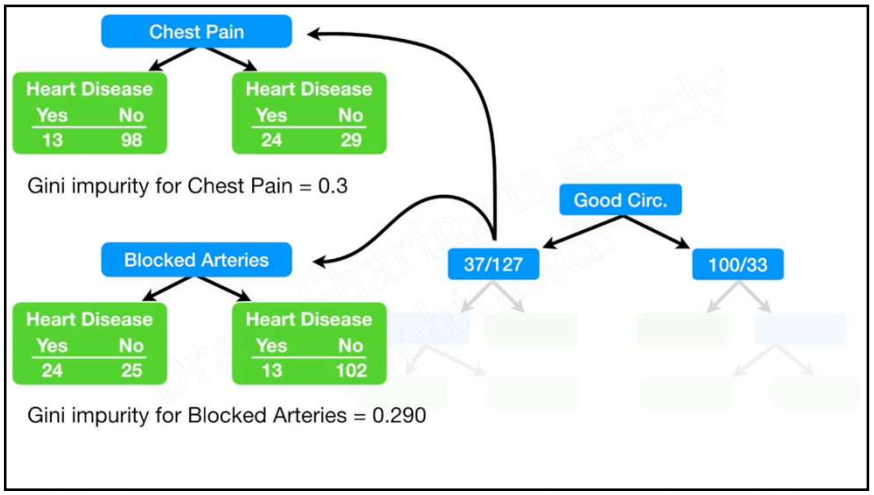
Gini impurity for Good Blood Circulation = 0.360

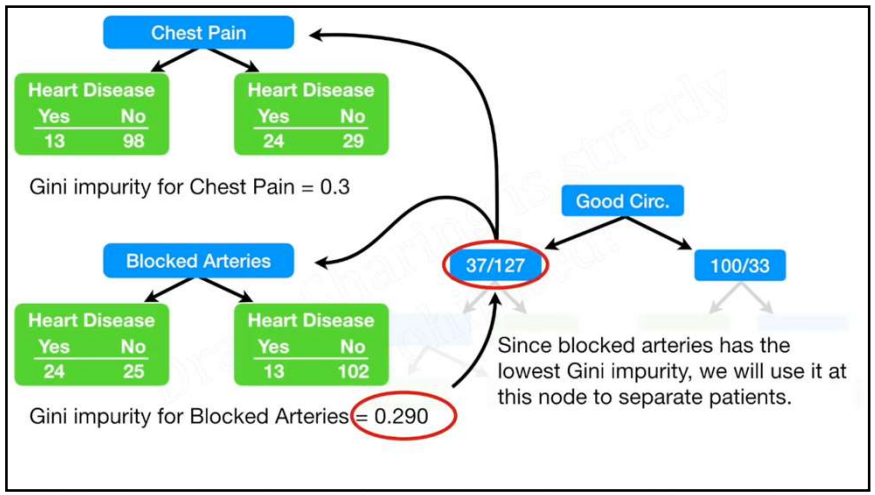
Gini impurity for Blocked Arteries = 0.381

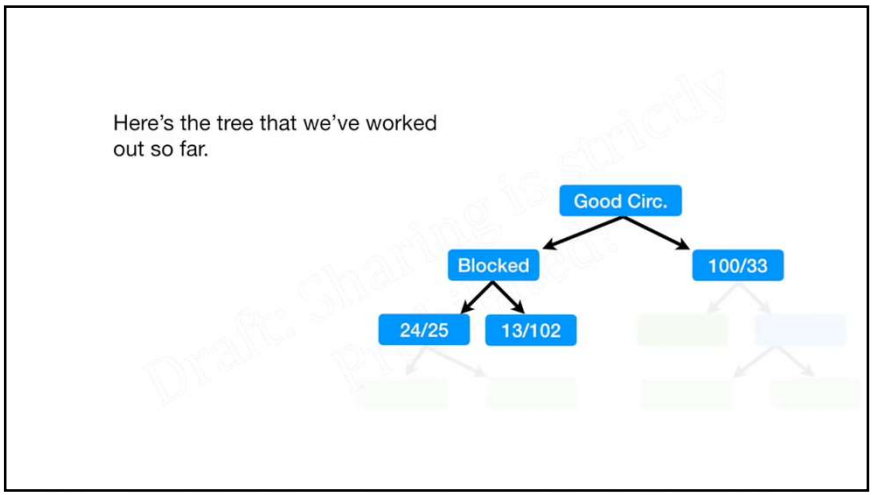
Now we need to figure how well **chest pain** and **blocked arteries** separate these 164 patients (37 with heart disease and 127 without heart disease).



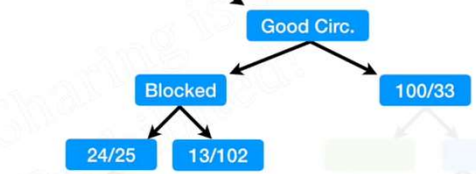

Chest Pain
Heart Disease
Yes No
13 98
Heart Disease
Yes No
24 29
Gini impurity for Chest Pain = 0.3
Good Circ.
37/127
100/33


Chest Pain
Heart Disease
Yes No
13 98
Heart Disease
Yes No
24 29
Gini impurity for Chest Pain = 0.3
Good Circ.
Blocked Arteries
37/127
100/33
Heart Disease
Yes No
24 25
Heart Disease
Yes No
13 102
Gini impurity for Blocked Arteries = 0.290

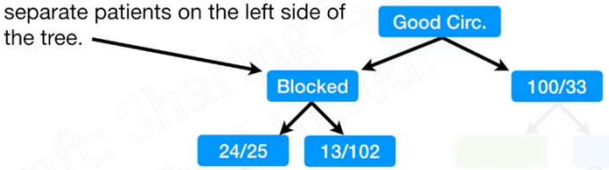

Chest Pain
Heart Disease
Yes No
13 98
Heart Disease
Yes No
24 29
Gini impurity for Chest Pain = 0.3
Good Circ.
Blocked Arteries
37/127
100/33
Heart Disease
Yes No
24 25
Heart Disease
Yes No
13 102
Since blocked arteries has the lowest Gini impurity, we will use it at this node to separate patients.
Gini impurity for Blocked Arteries = 0.290


Here's the tree that we've worked out so far.
Good Circ.
Blocked
100/33
24/25
13/102

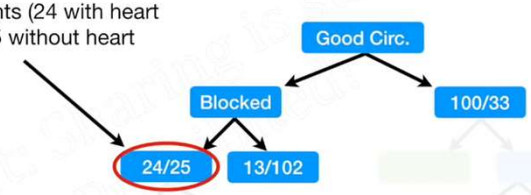
We started at the top by separating patients with Good Circulation…

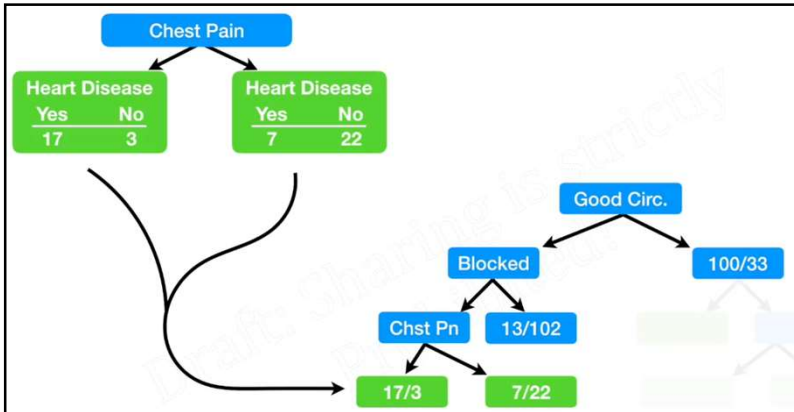


…then we used Blocked Arteries to separate patients on the left side of the tree.



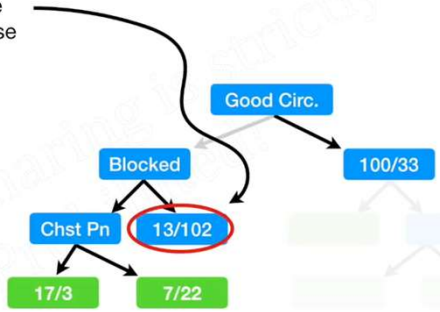
All we have left is Chest Pain, so first we'll see how well it separates these 49 patients (24 with heart disease and 25 without heart disease).





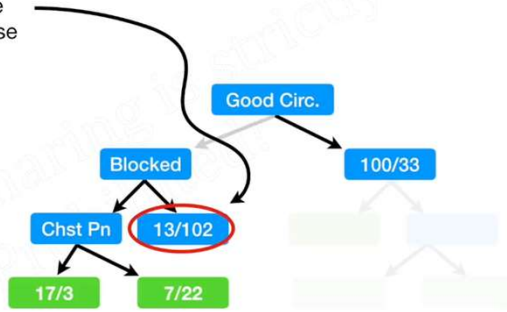
…so these are the final leaf nodes on this branch of the tree.

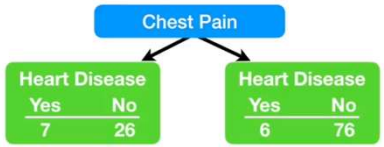
Now let's see what happens when we use chest pain to divide these 115 patients (13 with heart disease and 102 without).



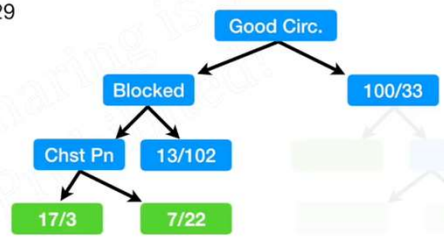
Now let's see what happens when we use chest pain to divide these 115 patients (13 with heart disease and 102 without).

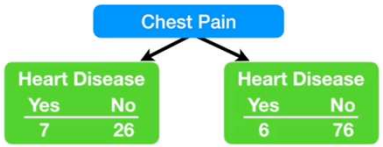
NOTE: The vast majority of the patients in this node (89%) don't have heart disease.





Gini impurity for Chest Pain = 0.29





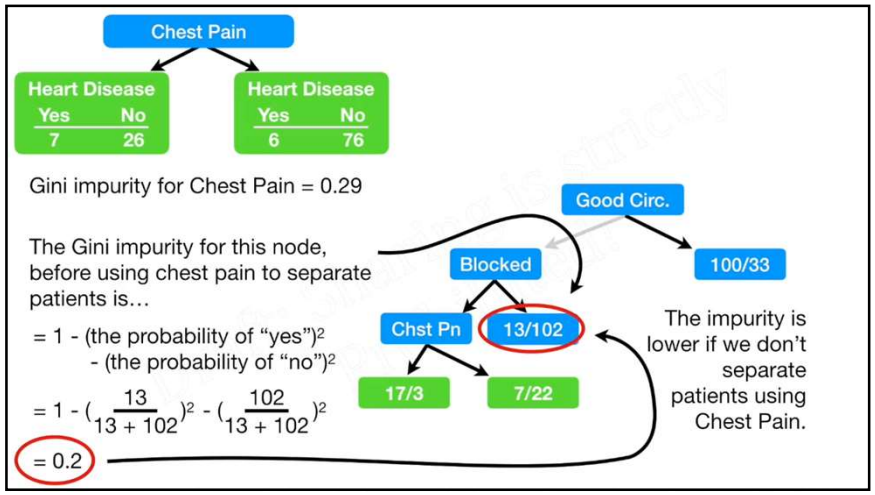
Gini impurity for Chest Pain = 0.29

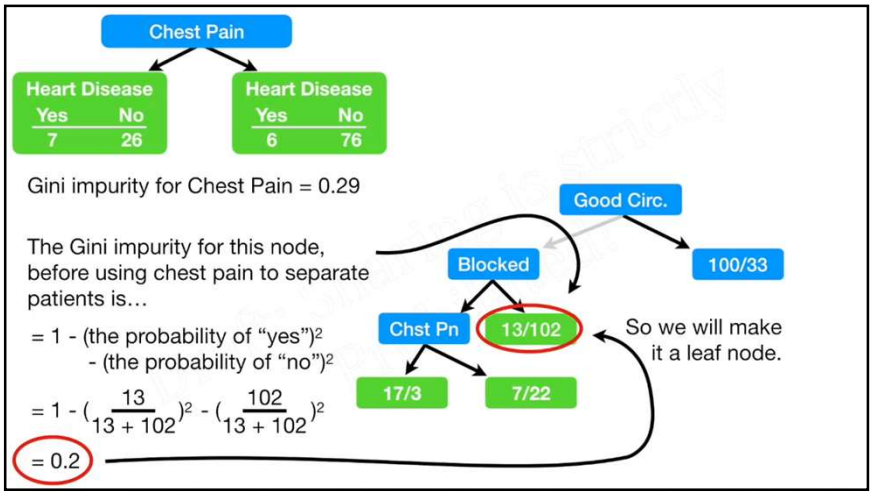
The Gini impurity for this node, before using chest pain to separate patients is…

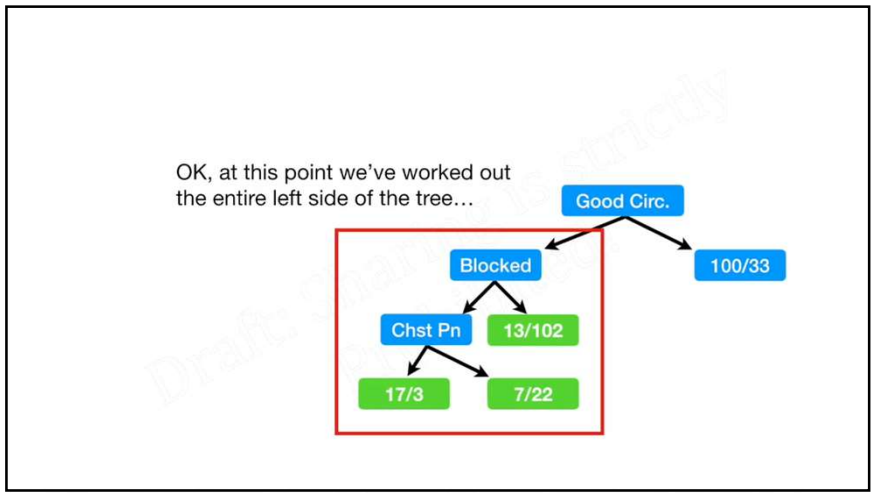
= 1 - (the probability of "yes")^2
- (the probability of "no")^2

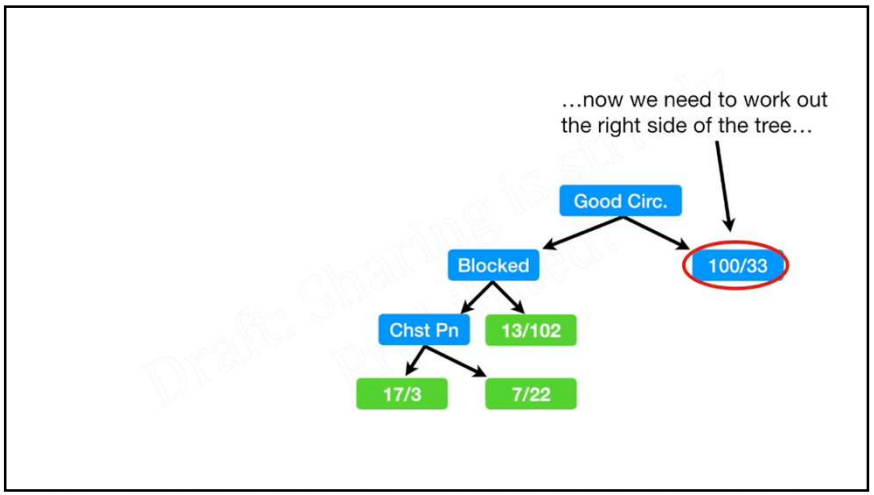
$$= 1 - \left(\frac{13}{13 + 102}\right)^2 - \left(\frac{102}{13 + 102}\right)^2$$

= 0.2


Chest Pain
Heart Disease
Yes No
7 26
Heart Disease
Yes No
6 76
Gini impurity for Chest Pain = 0.29
The Gini impurity for this node, before using chest pain to separate patients is…
= 1 - (the probability of "yes")²
- (the probability of "no")²
= 1 - (13 / 13 + 102)² - (102 / 13 + 102)²
= 0.2
Good Circ.
Blocked
100/33
Chst Pn
13/102
17/3
7/22
The impurity is lower if we don't separate patients using Chest Pain.

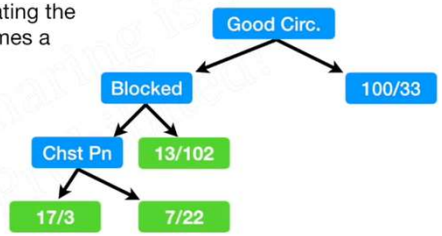

Chest Pain
Heart Disease
Yes No
7 26
Heart Disease
Yes No
6 76
Gini impurity for Chest Pain = 0.29
The Gini impurity for this node, before using chest pain to separate patients is…
= 1 - (the probability of "yes")²
- (the probability of "no")²
= 1 - (13 / 13 + 102)² - (102 / 13 + 102)²
= 0.2
Good Circ.
Blocked
100/33
Chst Pn
13/102
17/3
7/22
So we will make it a leaf node.


OK, at this point we've worked out the entire left side of the tree…
Good Circ.
Blocked
100/33
Chst Pn
13/102
17/3
7/22

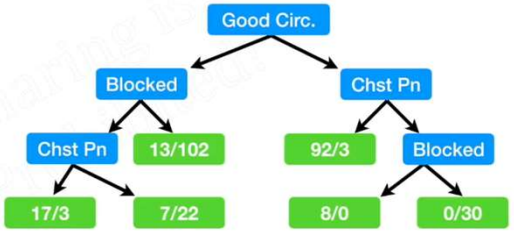

…now we need to work out the right side of the tree…
Good Circ.
Blocked
100/33
Chst Pn
13/102
17/3
7/22

The good news is that we follow the exact same steps as we did on the left side:

1) Calculate all of the Gini impurity scores.
2) If the node itself has the lowest score, than there is no point in separating the patients any more and it becomes a leaf node.
3) If separating the data results in an improvement, than pick the separation with the lowest impurity value.

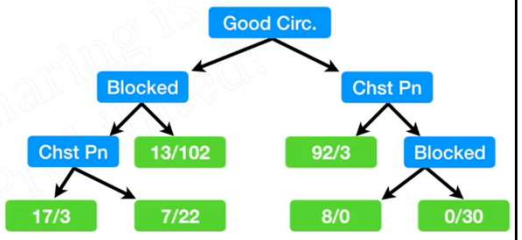


Hooray!!! We made a decision tree!!!



So far we've seen how to build a tree with "yes/no" questions at each step…

…but what if we have numeric data, like patient weight?



Weight	Heart Disease
220	Yes
180	Yes
225	Yes
190	No
155	No

Imagine if this were our data…

	Weight	Heart Disease
Lowest	155	No
	180	Yes
	190	No
	220	Yes
Highest	225	Yes

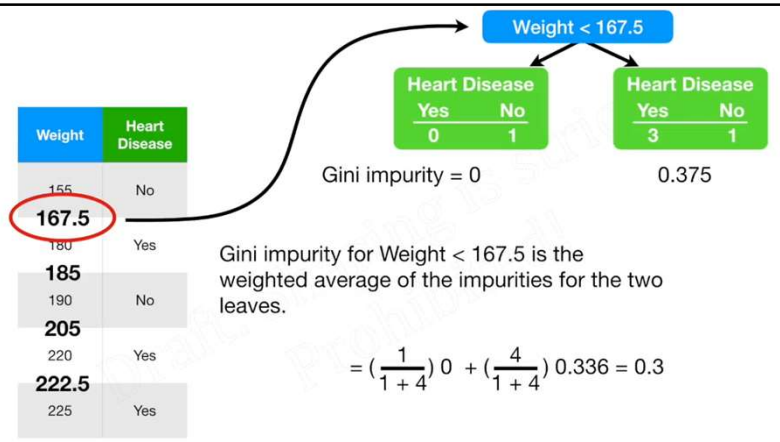
Step 1) Sort the patients by weight, lowest to highest.

Weight	Heart Disease
155	No
167.5	
180	Yes
185	
190	No
205	
220	Yes
222.5	
225	Yes

Step 2) Calculate the average weight for all adjacent patients.

Weight	Heart Disease
155	No
167.5	→ Gini impurity = ?
180	Yes
185	→ Gini impurity = ?
190	No
205	→ Gini impurity = ?
220	Yes
222.5	→ Gini impurity = ?
225	Yes

Step 3) Calculate the impurity values for each average weight.



Gini impurity for Weight < 167.5 is the weighted average of the impurities for the two leaves.

$$= \left(\frac{1}{1+4}\right) 0 + \left(\frac{4}{1+4}\right) 0.336 = 0.3$$

Weight	Heart Disease
155	No
167.5	→ Gini impurity = 0.3
180	Yes
185	→ Gini impurity = 0.47
190	No
205	→ Gini impurity = 0.27
220	Yes
222.5	→ Gini impurity = 0.4
225	Yes

Weight	Heart Disease
155	No
167.5	→ Gini impurity = 0.3
180	Yes
185	→ Gini impurity = 0.47
190	No
205	→ Gini impurity = 0.27
220	Yes
222.5	→ Gini impurity = 0.4
225	Yes

The lowest impurity occurs when we separate using **weight < 205**…

Weight	Heart Disease
155	No
167.5	→ Gini impurity = 0.3
180	Yes
185	→ Gini impurity = 0.47
190	No
205	→ Gini impurity = 0.27
220	Yes
222.5	→ Gini impurity = 0.4
225	Yes

The lowest impurity occurs when we separate using **weight < 205**…

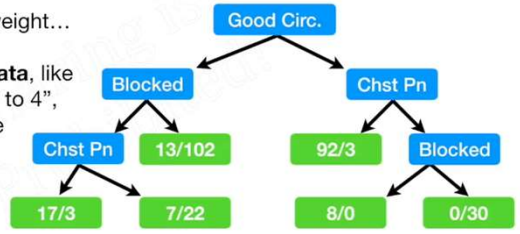
…so this is the cutoff and impurity value we will use when we compare weight to chest pain or blocked arteries.

Now we've seen how to build a tree with…

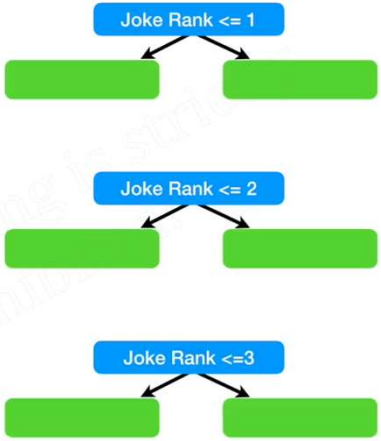
1) "yes/no" questions at each step…

2) Numeric data, like patient weight…

Now let's talk about **ranked data**, like "rank my jokes on a scale of 1 to 4", and **multiple choice data**, like "which color do you like, red, blue or green?"

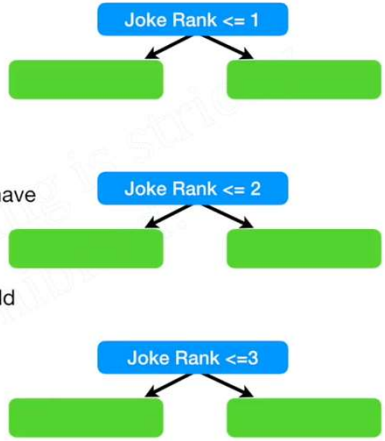


Rank my jokes…	Likes StatQuest
1	Yes
1	No
3	Yes
1	Yes
etc…	etc…



Rank my jokes…	Likes StatQuest
1	Yes
1	No
3	Yes
1	Yes
etc…	etc…

NOTE: We don't have to calculate an impurity score for Joke Rank <= 4 because that would include everyone.



Color Choice	Likes StatQuest
Green	Yes
Blue	No
Red	Yes
Green	Yes
etc…	etc…

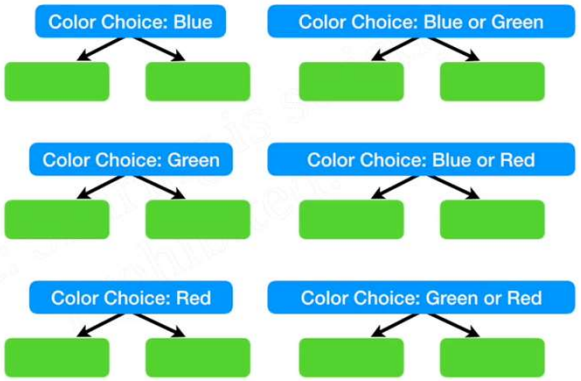
When there are **multiple choices**, like "**color choice can be blue, green or red**", you calculate an impurity score for each one as well as each possible combination.

Color Choice	Likes StatQuest
Green	Yes
Blue	No
Red	Yes
Green	Yes
etc…	etc…

When there are **multiple choices**, like "**color choice can be blue, green or red**", you calculate an impurity score for each one as well as each possible combination.

For this example, with three colors (blue, green and red) we get the following options…

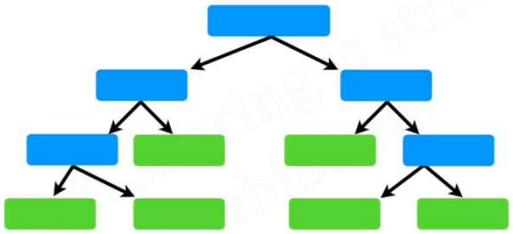
Color Choice	Likes StatQuest
Green	Yes
Blue	No
Red	Yes
Green	Yes
etc…	etc…



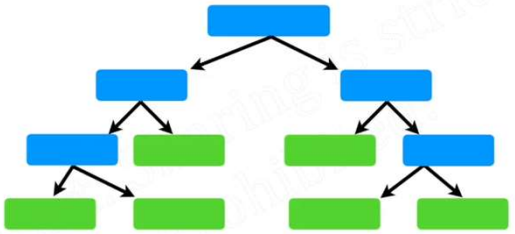
Color Choice	Likes StatQuest
Green	Yes
Blue	No
Red	Yes
Green	Yes
etc…	etc…

NOTE: We don't have to calculate an impurity score for "Color Choice: Blue or Green or Red" since that includes everyone.

Decision Trees are easy to build, easy to use and easy to interpret…

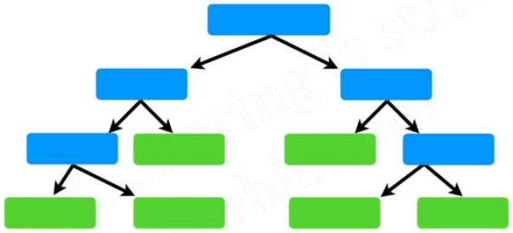


Decision Trees are easy to build, easy to use and easy to interpret…

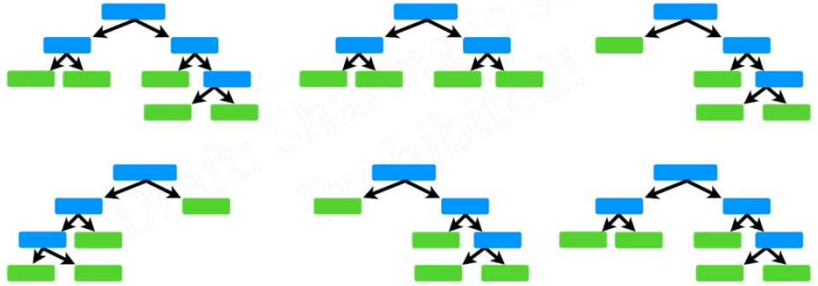


…but in practice they are not that awesome.

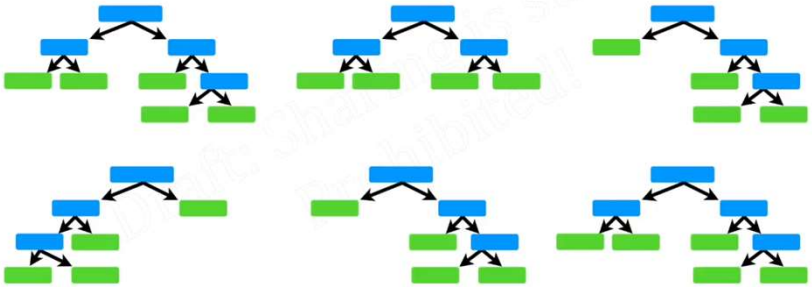
In other words, they work great with the data used to create them, but **they are not flexible when it comes to classifying new samples**.



The good news is that **Random Forests** combine the simplicity of decision trees with flexibility resulting in a vast improvement in accuracy.



So let's make a Random Forest!!!



Step 1: Create a "bootstrapped" dataset.

Original Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
No	No	No	125	No
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes

Imagine that these 4 samples are the entire dataset that we are going to build a tree from…

Original Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
No	No	No	125	No
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes

Bootstrapped Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease

To create a bootstrapped dataset that is the same size as the original, we just randomly select samples from the original dataset.

The important detail is that we're allowed to pick the same sample more than once.

Original Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
No	No	No	125	No
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes

Bootstrapped Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

…and here it is.

Step 2: Create a decision tree using the bootstrapped dataset, but only use a random subset of variables (or columns) at each step.

Bootstrapped Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

Step 2: Create a decision tree using the bootstrapped dataset, but only use a random subset of variables (or columns) at each step.

In this example, we will only consider 2 variables (columns) at each step.

Bootstrapped Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

Thus, instead of considering all 4 variables to figure out how to split the root node…

Bootstrapped Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

Bootstrapped Dataset

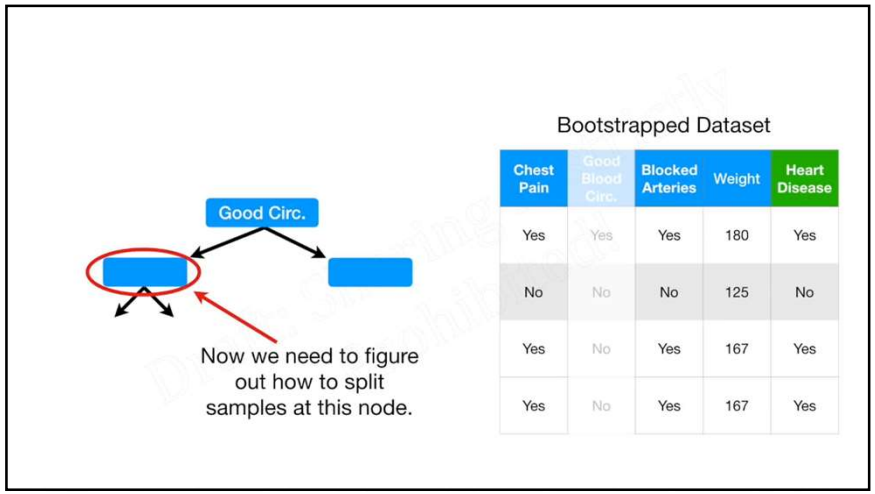
Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
No	Yes	No	167	Yes
No	Yes	No	167	Yes

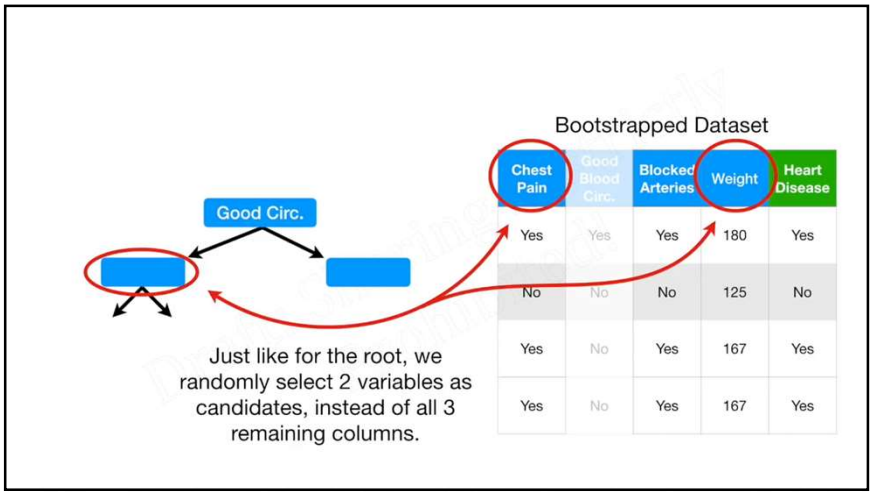
In this case, we randomly selected **Good Blood Circulation** and **Blocked Arteries** as candidates for the root node.

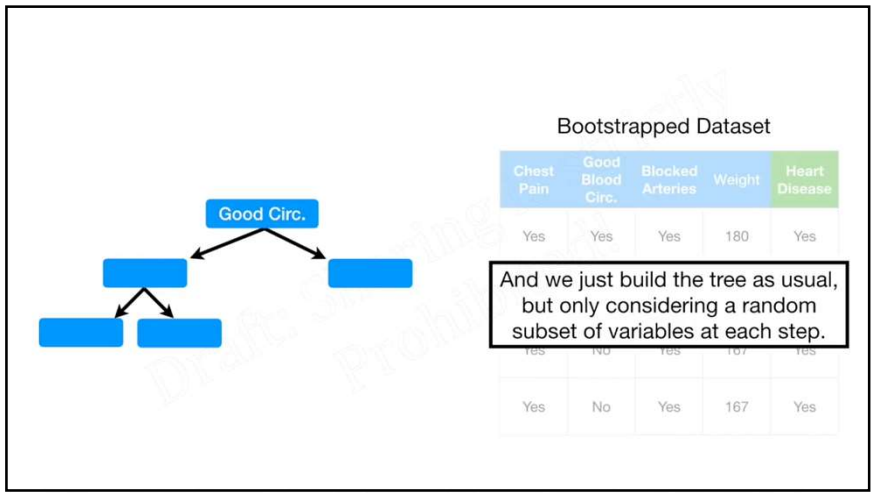
Just for the sake of the example, assume that **Good Blood Circulation** did the best job separating the samples.

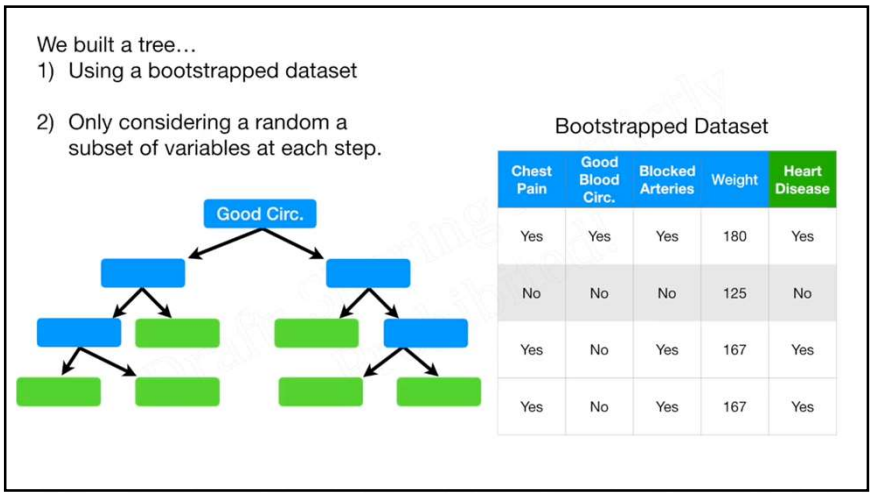
Bootstrapped Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

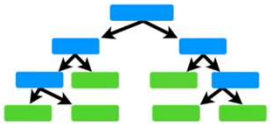

Good Circ.
Now we need to figure out how to split samples at this node.
Bootstrapped Dataset
Chest Pain
Good Blood Circ.
Blocked Arteries
Weight
Heart Disease
Yes Yes Yes 180 Yes
No No No 125 No
Yes No Yes 167 Yes
Yes No Yes 167 Yes


Good Circ.
Just like for the root, we randomly select 2 variables as candidates, instead of all 3 remaining columns.
Bootstrapped Dataset
Chest Pain
Good Blood Circ.
Blocked Arteries
Weight
Heart Disease
Yes Yes Yes 180 Yes
No No No 125 No
Yes No Yes 167 Yes
Yes No Yes 167 Yes

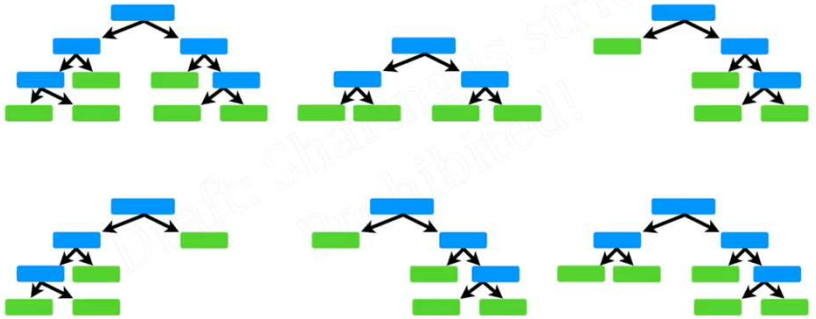

Good Circ.
Bootstrapped Dataset
Chest Pain
Good Blood Circ.
Blocked Arteries
Weight
Heart Disease
Yes Yes Yes 180 Yes
And we just build the tree as usual, but only considering a random subset of variables at each step.
Yes No Yes 167 Yes


We built a tree…
1) Using a bootstrapped dataset
2) Only considering a random a subset of variables at each step.
Good Circ.
Bootstrapped Dataset
Chest Pain
Good Blood Circ.
Blocked Arteries
Weight
Heart Disease
Yes Yes Yes 180 Yes
No No No 125 No
Yes No Yes 167 Yes
Yes No Yes 167 Yes

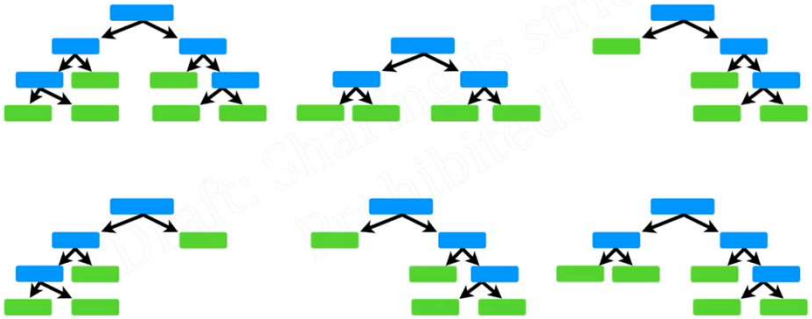
Here's the tree we just made…



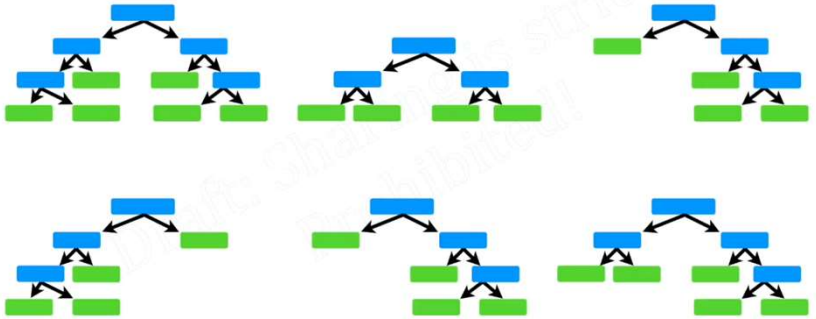
Now go back to Step 1 and repeat: Make a new bootstrapped dataset and build a tree considering a subset of variables at each step.



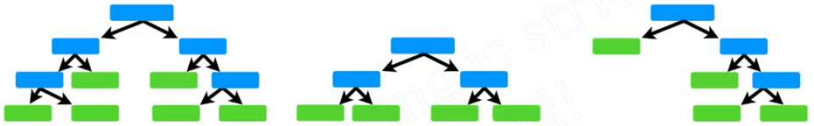
Ideally, you'd do this 100's of times, but we only have space to show 6… but you get the idea.



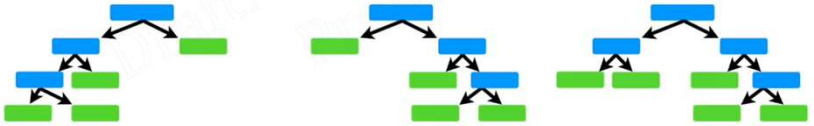
Using a bootstrapped sample and considering only a subset of the variables at each step results in a wide variety of trees.



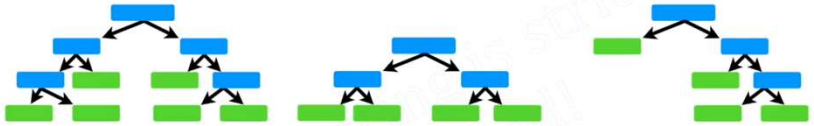
Using a bootstrapped sample and considering only a subset of the variables at each step results in a wide variety of trees.

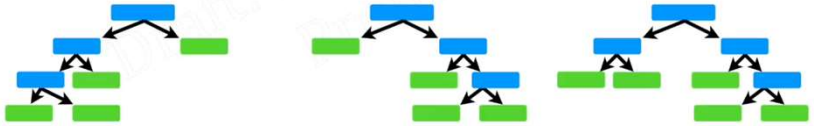


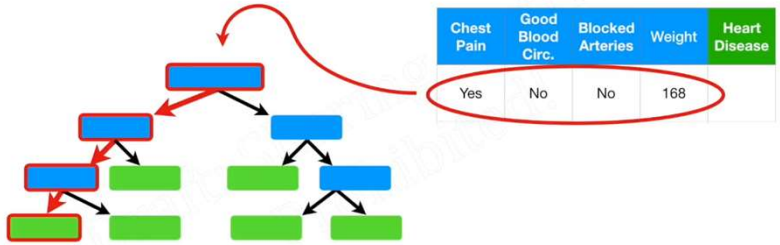
The variety is what makes random forests more effective than individual decision trees.



Sweet!!! Now that we've created a random forest, how do we use it?

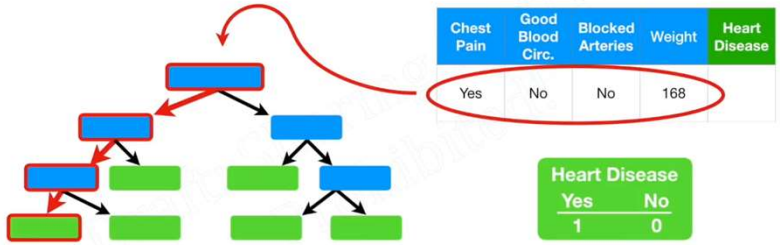






Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	No	No	168	

The first tree says "Yes"…



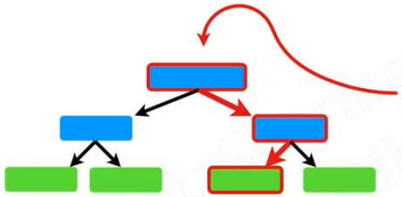
Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	No	No	168	

Heart Disease	
Yes	No
1	0

The first tree says "Yes"…

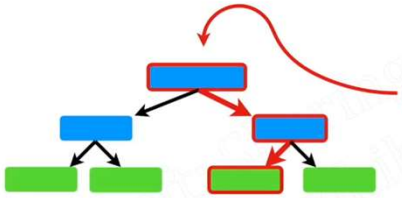
…and we keep track of that here.

Now we run the data down the second tree that we made…



Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	No	No	168	

Heart Disease	
Yes	No
1	0



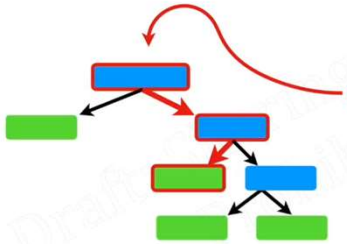
Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	No	No	168	

The second tree also says "Yes"…

Heart Disease	
Yes	No
2	0

…and we keep track of that here.

Then we repeat for all the trees that we made…



Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	No	No	168	

Heart Disease	
Yes	No
4	1

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	No	No	168	

After running the data down all of the trees in the random forest, we see which option received more votes.

Heart Disease	
Yes	No
5	1

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	No	No	168	YES

In this case, "**Yes**" received the most votes, so we will conclude that this patient has heart disease.

Heart Disease	
Yes	No
5	1

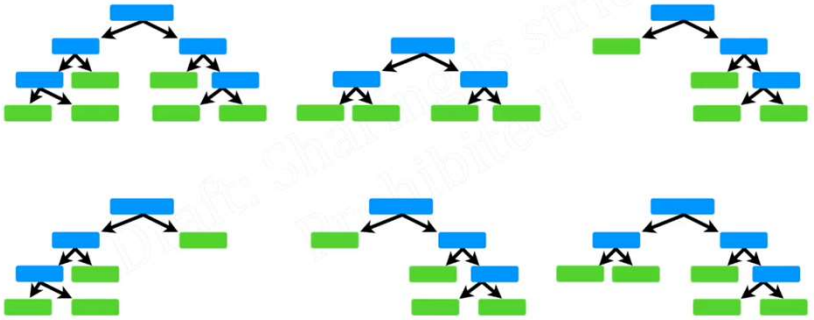
Terminology Alert!!!

Bootstrapping the data plus using the **agg**regate to make a decision is called "**Bagging**"

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	No	No	168	YES

Heart Disease	
Yes	No
5	1

How do we know if it's any good?



Remember when we created the bootstrapped dataset?

Original Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
No	No	No	125	No
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes

Bootstrapped Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes

We allowed duplicate entries in the bootstrapped dataset…

Original Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
No	No	No	125	No
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes

Bootstrapped Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

As a result, this entry was not included in the bootstrapped dataset.

Original Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
No	No	No	125	No
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes

Bootstrapped Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

Typically, about 1/3 of the original data does not end up in the bootstrapped dataset.

Original Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
No	No	No	125	No
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes

Bootstrapped Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

Here is the entry that didn't end up in the bootstrapped dataset..

Original Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
No	No	No	125	No
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No

Original Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
No	No	No	125	No
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes

(psst… if the original dataset were larger, we'd have more than just 1 entry over here…)

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No

Original Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
No	No	No	125	No
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes

This is called the "**Out-Of-Bag Dataset**"

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No

Original Dataset

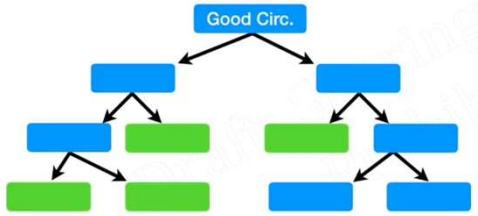
Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
No	No	No	125	No
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes

This is called the "**Out-Of-Bag Dataset**"

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No

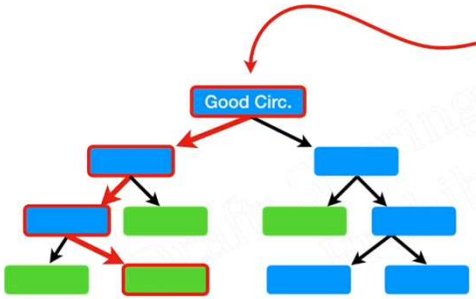
(If it were up to me, I would have named it the "**Out-Of-Boot Dataset**", since it's the entries that didn't make it into the bootstrap dataset.)

Since the Out-Of-Bag data was not used to create this tree…

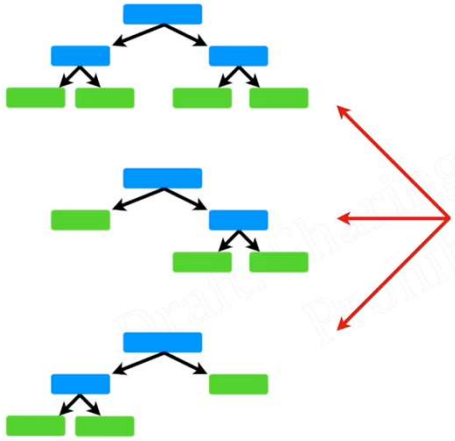


Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No

…we can run it through and see if it correctly classifies the sample as "No Heart Disease"

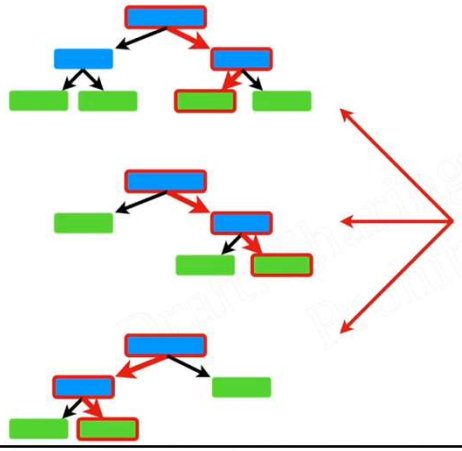


Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No



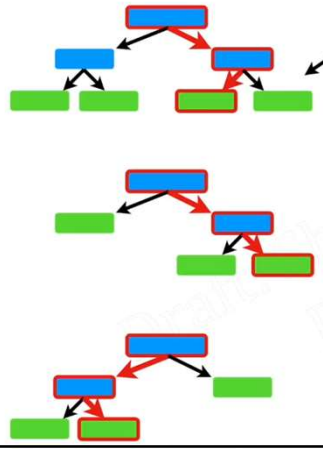
Then we run this Out-Of-Bag sample through all of the other trees that were built without it…

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No



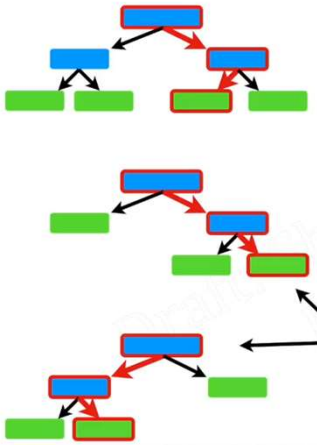
Then we run this Out-Of-Bag sample through all of the other trees that were built without it…

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No



This tree incorrectly labeled the Out-of-Bag sample "**Yes**".

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No



Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No

These trees correctly labeled the Out-of-Bag sample "**No**".

Classification of the Out-Of-Bag sample	
Yes	No
1	3

Since the label with the most votes wins, it is the label that we assign this Out-of-Bag sample.

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No

Classification of the Out-Of-Bag sample	
Yes	No
1	3

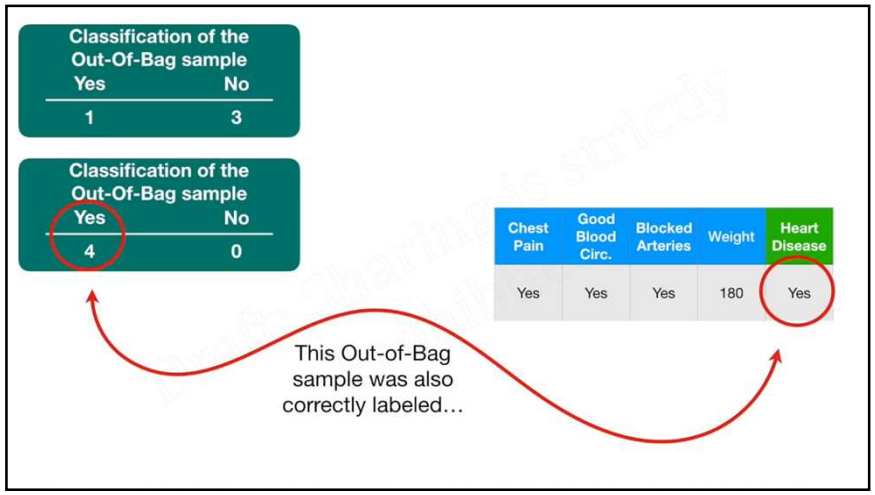
Since the label with the most votes wins, it is the label that we assign this Out-of-Bag sample.

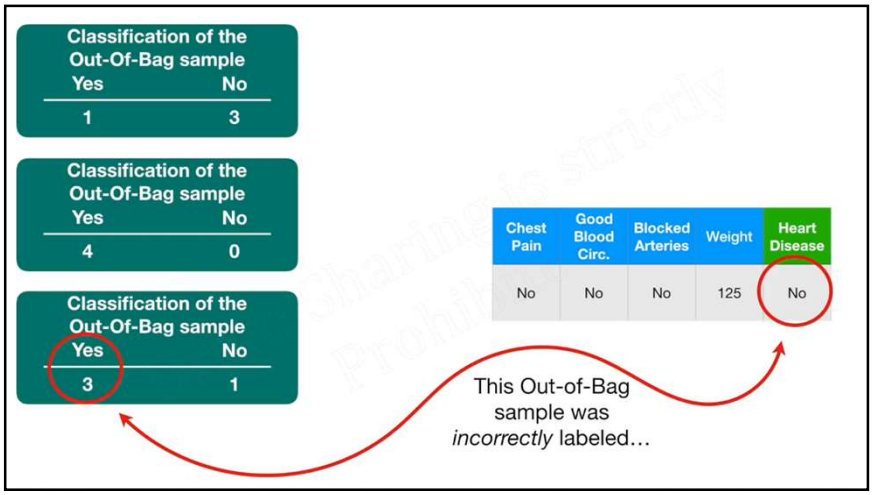
In this case, the Out-of-Bag sample is correctly labeled by the Random Forest.

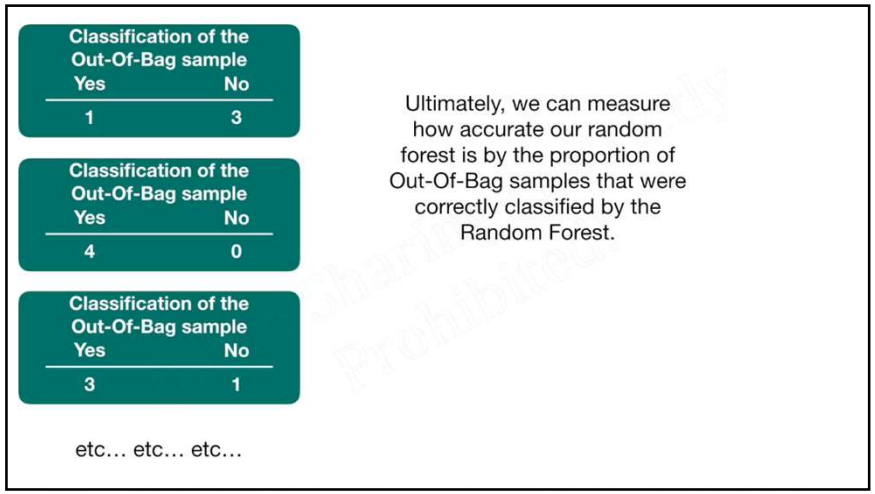
Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No

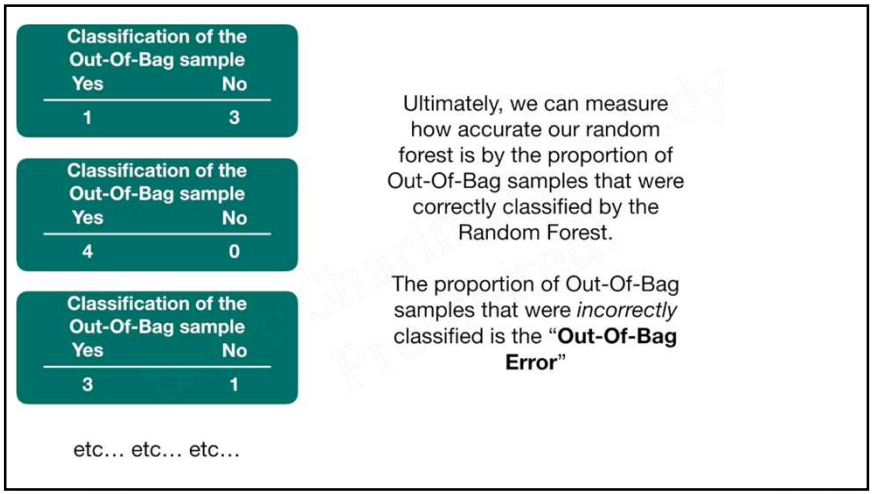
Classification of the Out-Of-Bag sample	
Yes	No
1	3

We then do the same thing for all of the other Out-Of-Bag samples for all of the trees.


Classification of the Out-Of-Bag sample
Yes No
1 3
Classification of the Out-Of-Bag sample
Yes No
4 0
Chest Pain
Good Blood Circ.
Blocked Arteries
Weight
Heart Disease
Yes Yes Yes 180 Yes
This Out-of-Bag sample was also correctly labeled…


Classification of the Out-Of-Bag sample
Yes No
1 3
Classification of the Out-Of-Bag sample
Yes No
4 0
Classification of the Out-Of-Bag sample
Yes No
3 1
Chest Pain
Good Blood Circ.
Blocked Arteries
Weight
Heart Disease
No No No 125 No
This Out-of-Bag sample was incorrectly labeled…


Classification of the Out-Of-Bag sample
Yes No
1 3
Classification of the Out-Of-Bag sample
Yes No
4 0
Classification of the Out-Of-Bag sample
Yes No
3 1
etc… etc… etc…
Ultimately, we can measure how accurate our random forest is by the proportion of Out-Of-Bag samples that were correctly classified by the Random Forest.


Classification of the Out-Of-Bag sample
Yes No
1 3
Classification of the Out-Of-Bag sample
Yes No
4 0
Classification of the Out-Of-Bag sample
Yes No
3 1
etc… etc… etc…
Ultimately, we can measure how accurate our random forest is by the proportion of Out-Of-Bag samples that were correctly classified by the Random Forest.
The proportion of Out-Of-Bag samples that were incorrectly classified is the "Out-Of-Bag Error"

OK, we now know how to:

1) Build a Random Forest
2) Use a Random Forest
3) Estimate the accuracy of a Random Forest.

Remember when we built our first tree and we only used 2 variables (columns of data) to make a decision at each step?

Bootstrapped Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

Now we can compare the Out-Of-Bag error for a random forest built using only 2 variables per step…

Bootstrapped Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

…to random forest built using 3 variables per step…

Bootstrapped Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

…to random forest built using 3 variables per step…

Bootstrapped Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

…and we test a bunch of different settings and choose the most accurate random forest.

In other words…

…change the number of variables used per step…

1) Build a Random Forest
2) Estimate the accuracy of a Random Forest.

In other words…

…change the number of variables used per step…

1) Build a Random Forest
2) Estimate the accuracy of a Random Forest.

Do this for a bunch of times and then choose the one that is most accurate.

In other words…

…change the number of variables used per step…

1) Build a Random Forest
2) Estimate the accuracy of a Random Forest.

SQRT Typically, we start by using the square of the number of variables and then try a a few settings above and below that value.


THANK YOU
Courtesy and Acknowledgement:
StatQuest: Random Forests: https://www.youtube.com/watch?v=J4Wdy0Wc_xQ
StatQuest: Decision Trees: https://www.youtube.com/watch?v=7VeUPuFGJHk