Lecture 10

Classification revisited

Topics

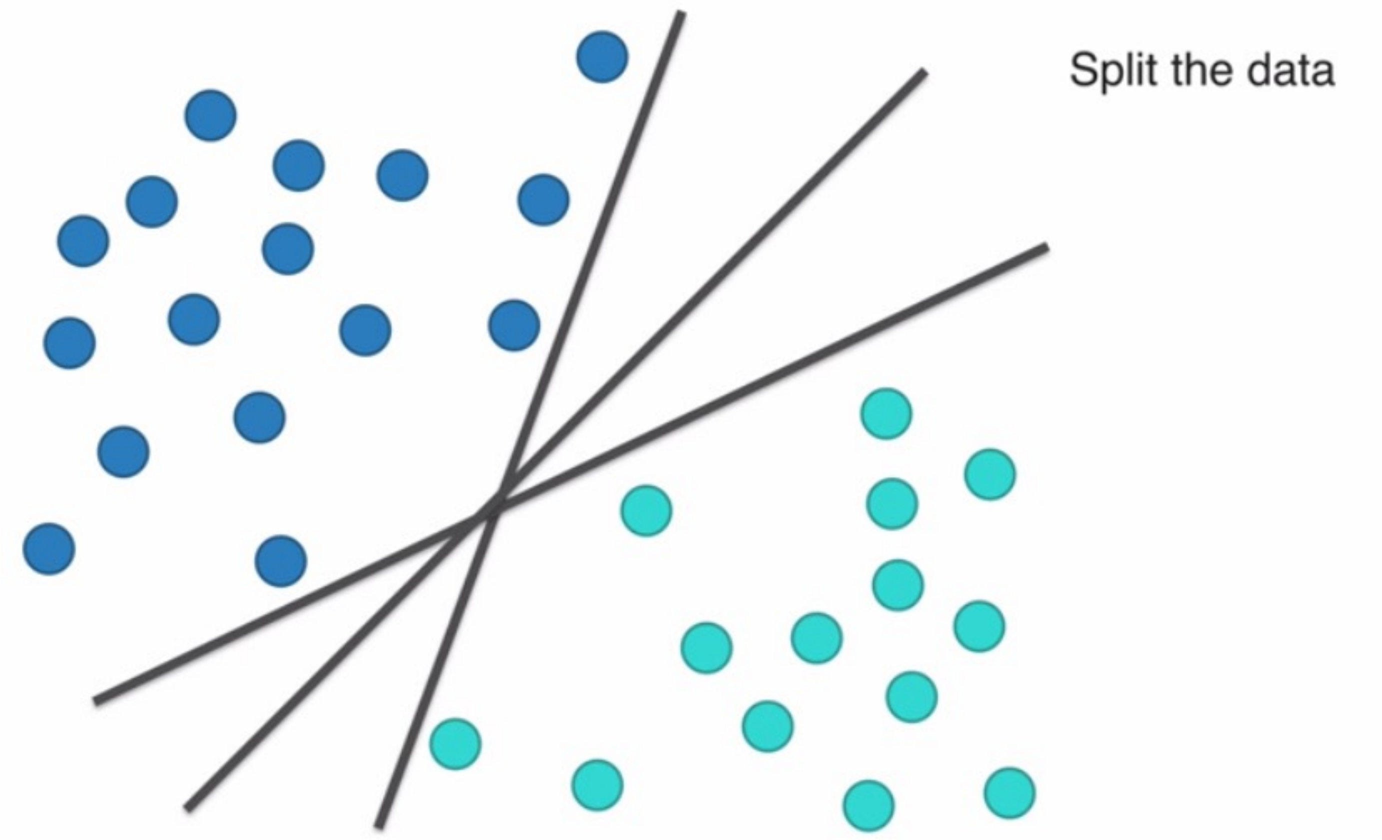
- Support Vector Machines
- Noise
- Kernels
- Decision trees
- Random forests

Support Vector Machines

[Alice Zhao]

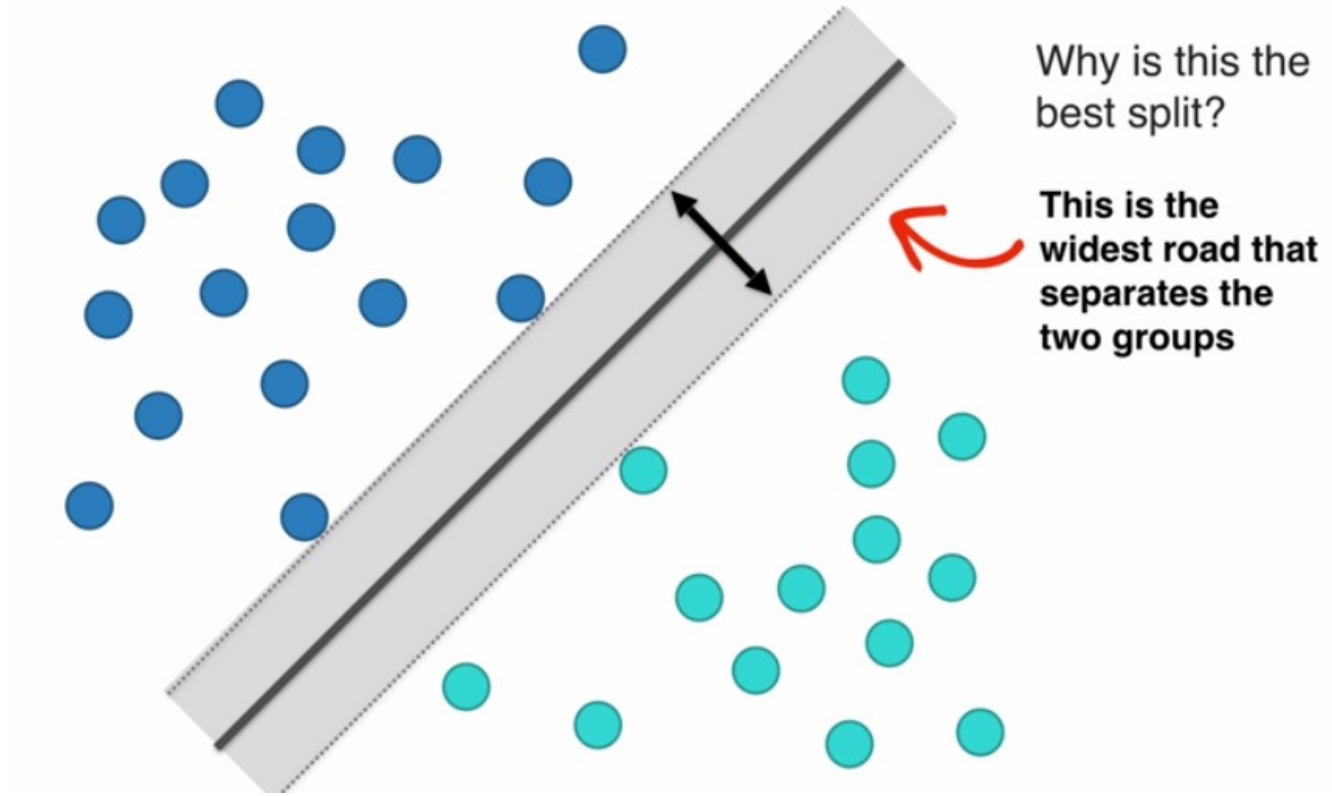
How should we split the data?

In classification we have this problem

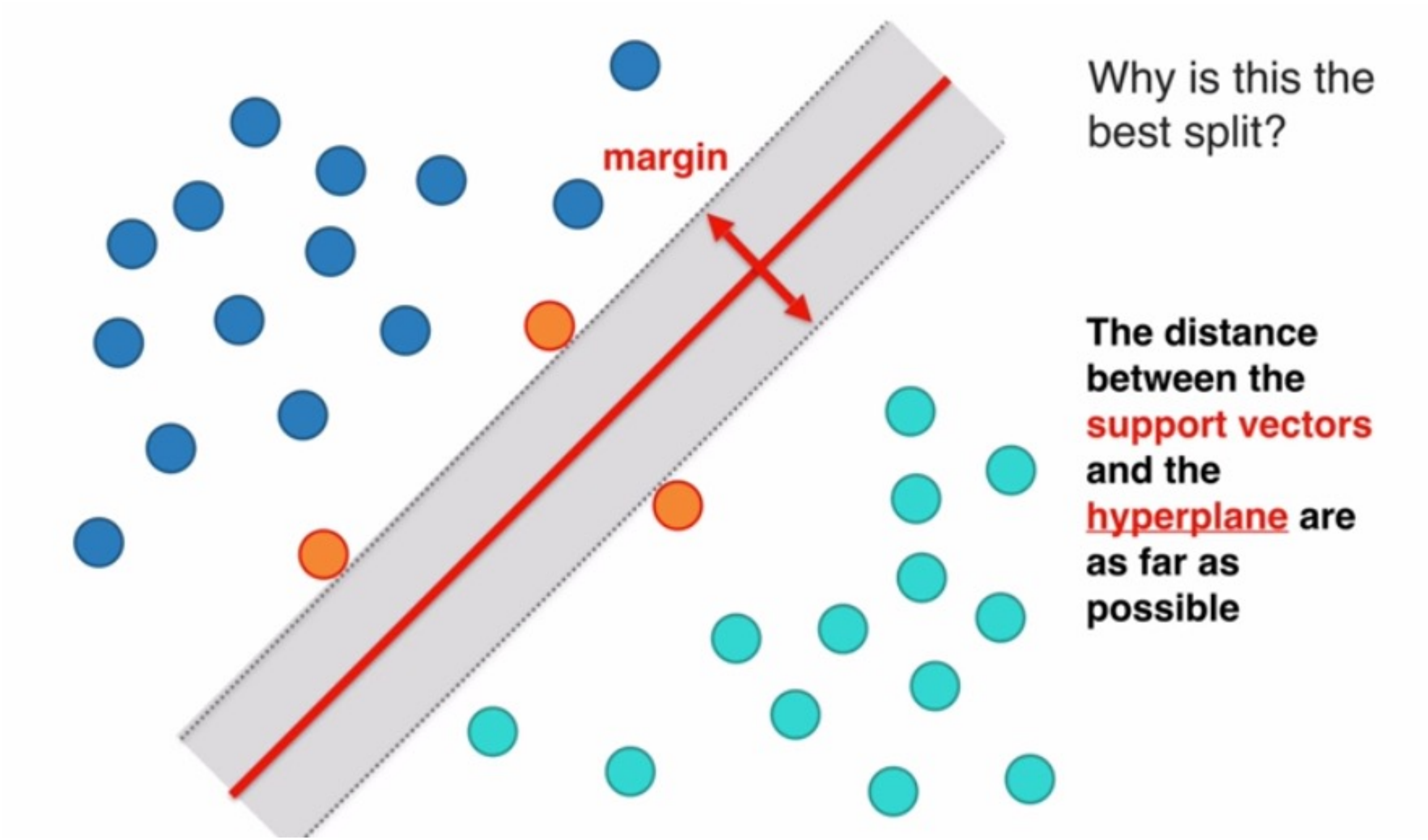


Maybe like this?

The SVM algorithm will do this split for us!



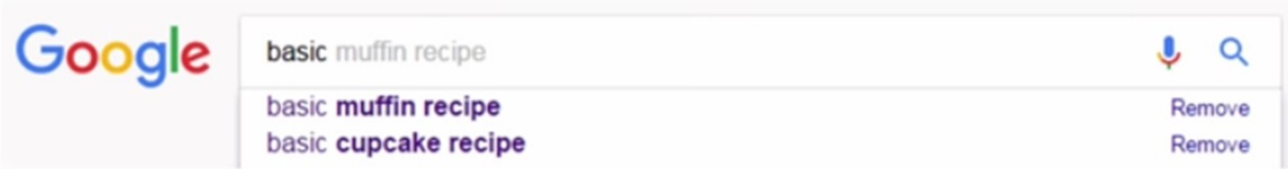
Motivation


margin
Why is this the best split?
The distance between the support vectors and the hyperplane are as far as possible

Cupcakes and muffins: What's the difference?

Let's proceed scientifically!

Collecting the data



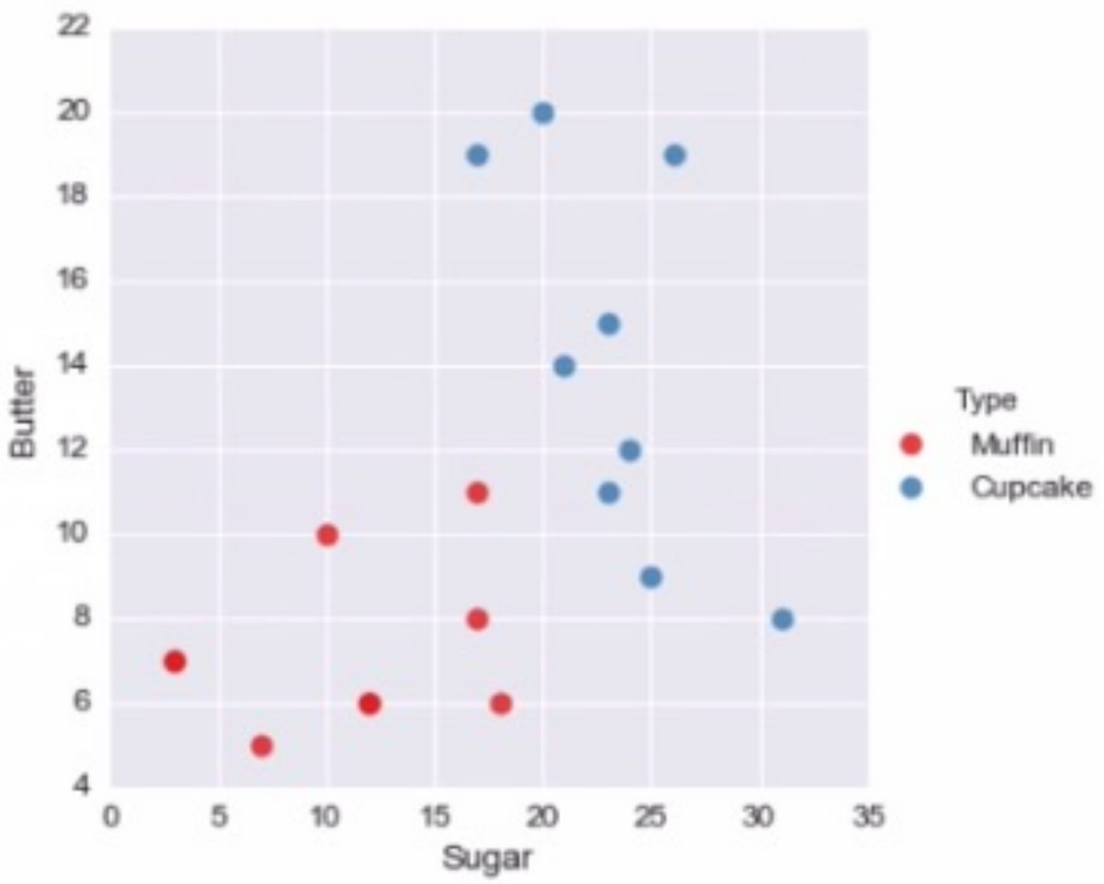
Find 9 recipes of each kind

Type	Flour	Milk	Sugar	Butter	Egg	Baking Powder	Vanilla	Salt
Muffin	55	28	3	7	5	2	0	0
Muffin	47	24	12	6	9	1	0	0
Muffin	47	23	18	6	4	1	0	0
Muffin	50	25	12	6	5	2	1	0
Muffin	55	27	3	7	5	2	1	0
Muffin	54	27	7	5	5	2	0	0
Muffin	47	26	10	10	4	1	0	0
Muffin	50	17	17	8	6	1	0	0
Muffin	50	17	17	11	4	1	0	0
Cupcake	39	0	26	19	14	1	1	0
Cupcake	34	17	20	20	5	2	1	0
Cupcake	39	13	17	19	10	1	1	0
Cupcake	38	15	23	15	8	0	1	0
Cupcake	42	18	25	9	5	1	0	0
Cupcake	36	14	21	14	11	2	1	0
Cupcake	38	15	31	8	6	1	1	0
Cupcake	36	16	24	12	9	1	1	0
Cupcake	34	17	23	11	13	0	1	0

Express the ingredients as weight percentages

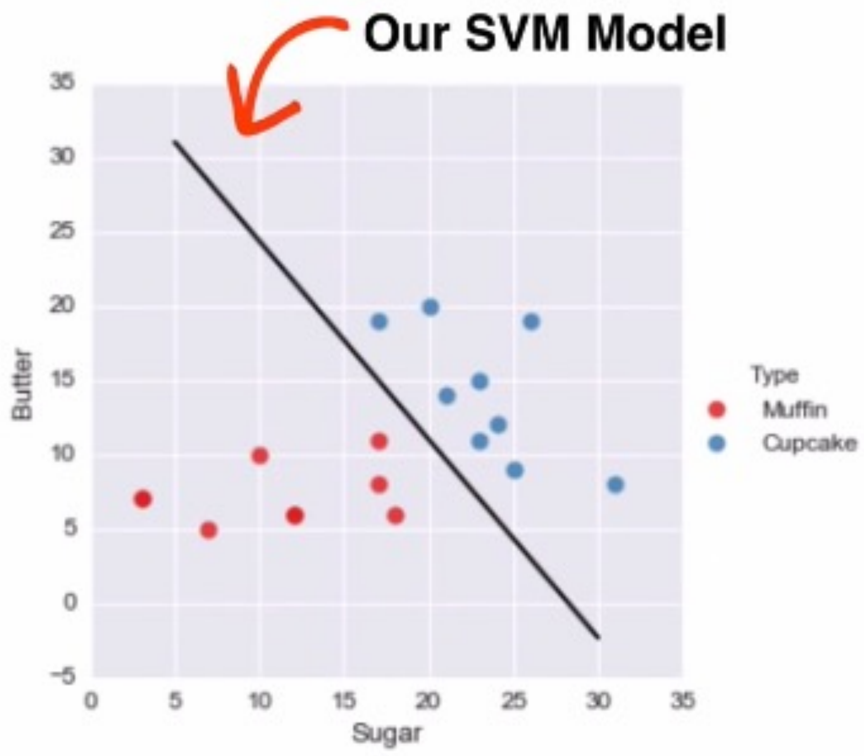
Plotting the data

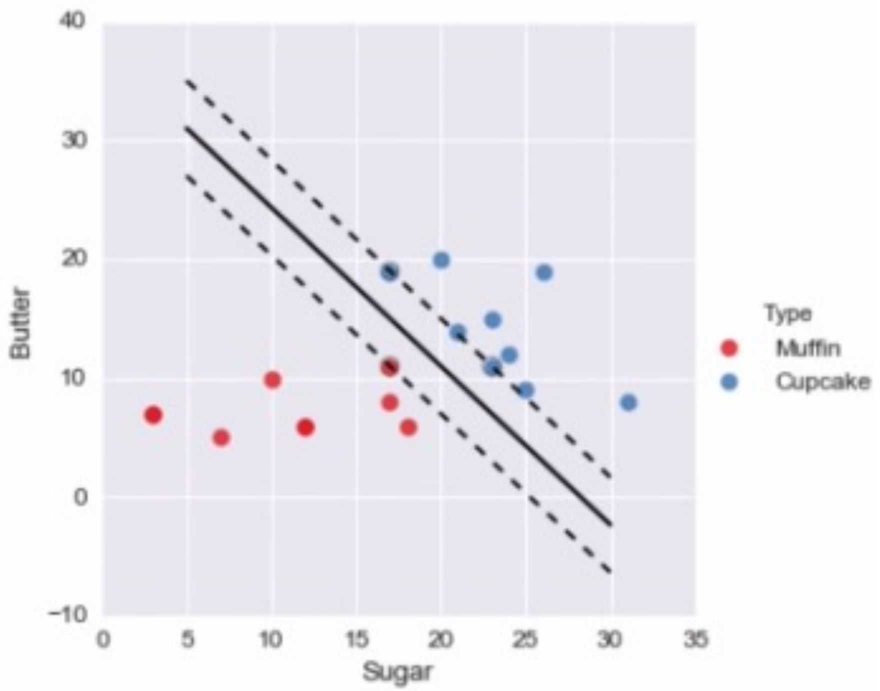
Focusing on butter and sugar only



Applying SVM

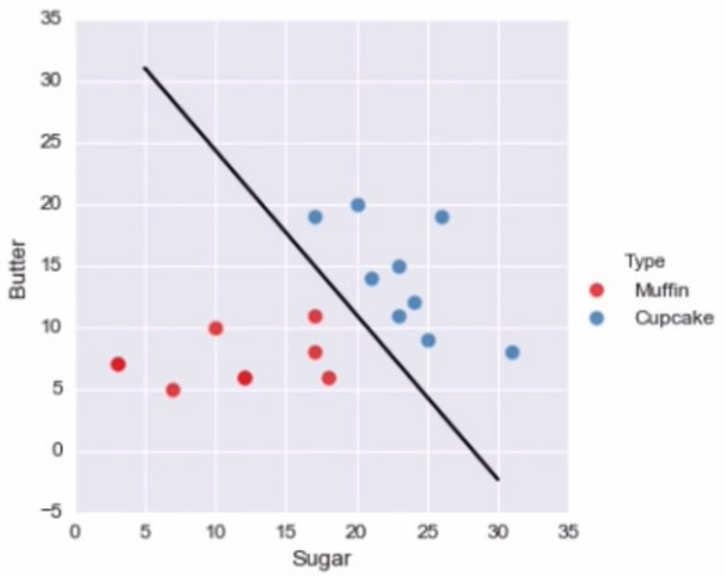
Finding the SVM model



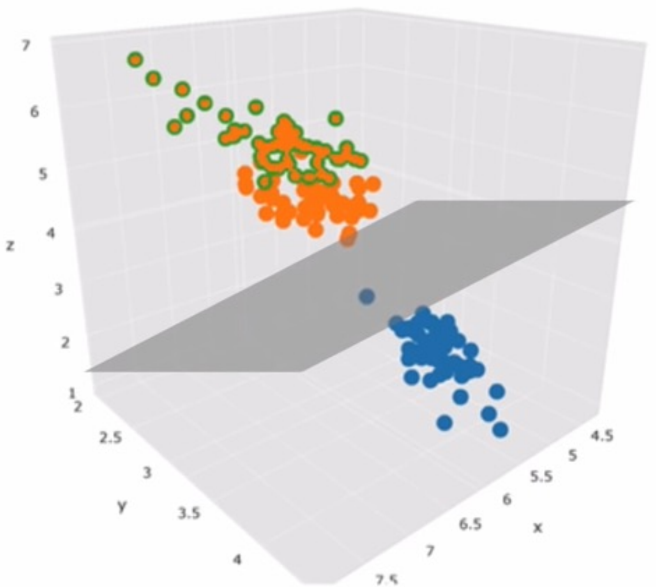


SVMs find separating hyperplanes

2D: Separate with Line



3D: Separate with Plane

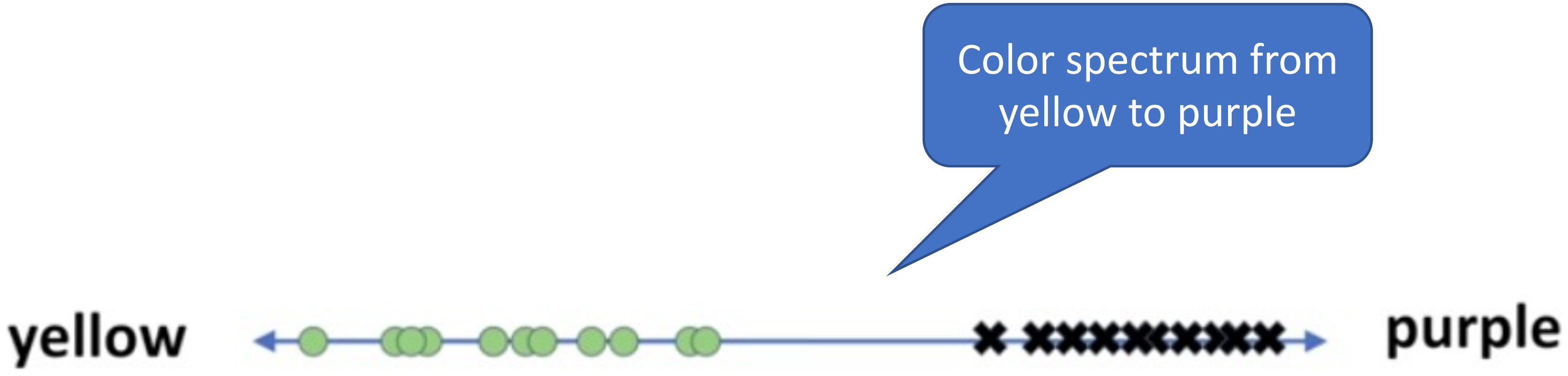


Key point: maximizing the margin can be formulated as a convex optimization problem, and therefore solved efficiently even in high-dimensional spaces.

Noise

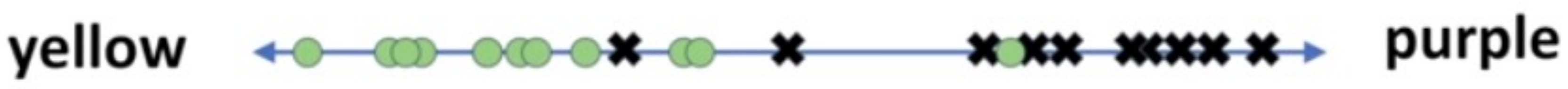
Brandon Rohrer

Peaches



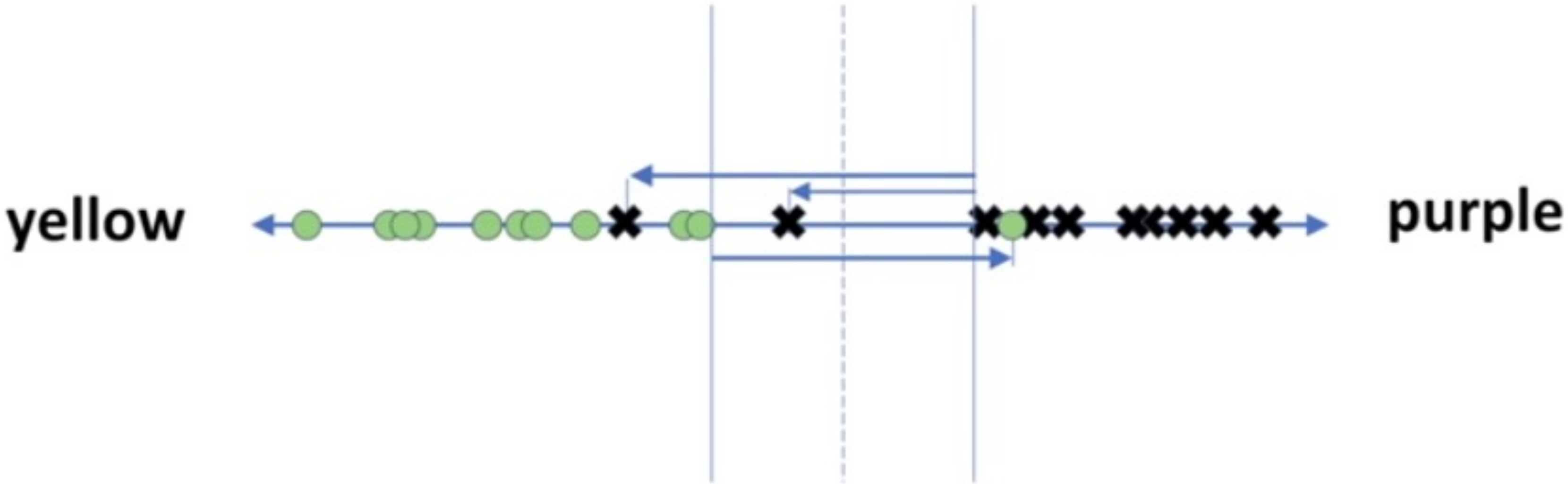
Each green dot represents an edible peach. Each black cross represents a non-edible peach.
Clean situation.

Peaches



Less clean situation.

Peaches



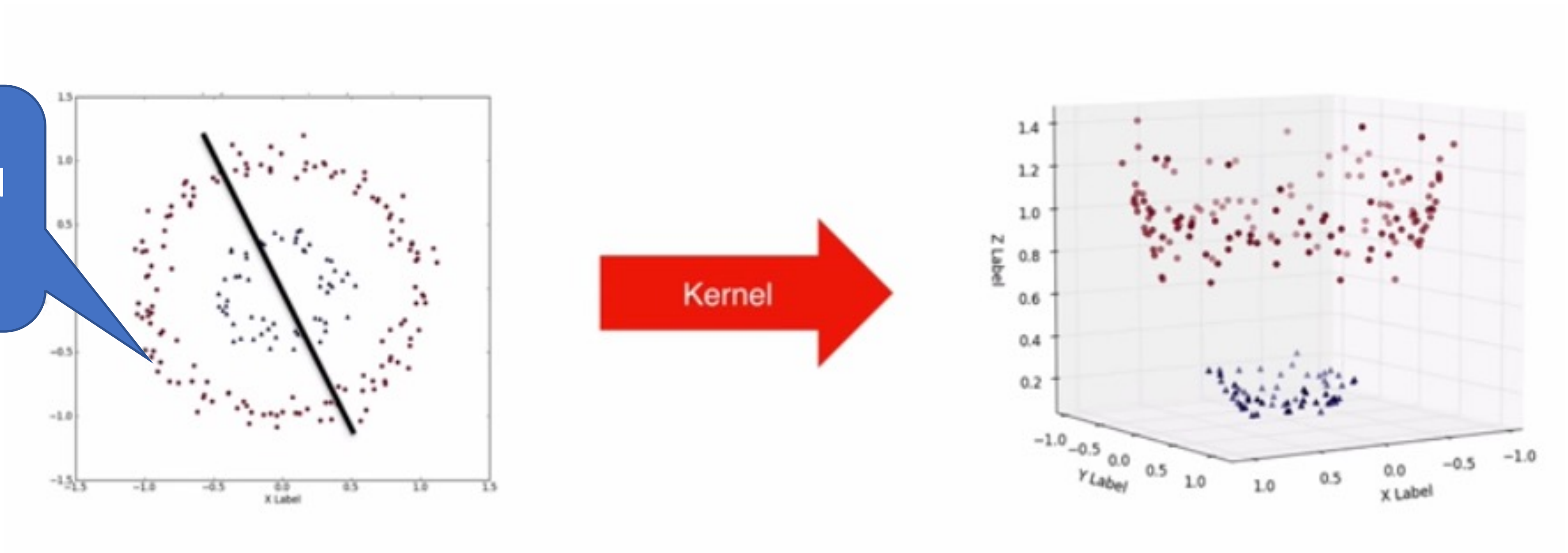
Use penalty for misclassified points. Still possible to solve it.

Kernels

Brandon Rohrer

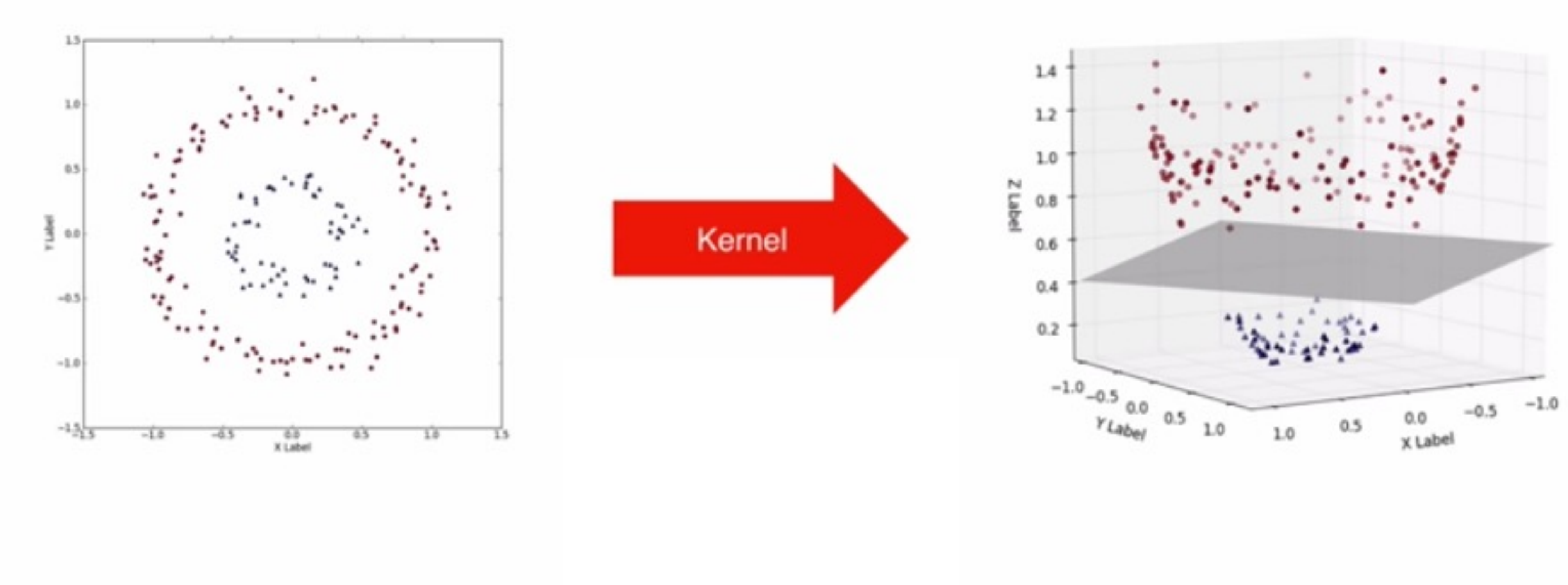
Kernels

Blue points (in the middle) and red points (in the perifery)



Here it is impossible to separate the two classes with a line! Idea: add a dimension and a kernel function that will make them separable! In this case: z = distance of (x,y) from the origin works.

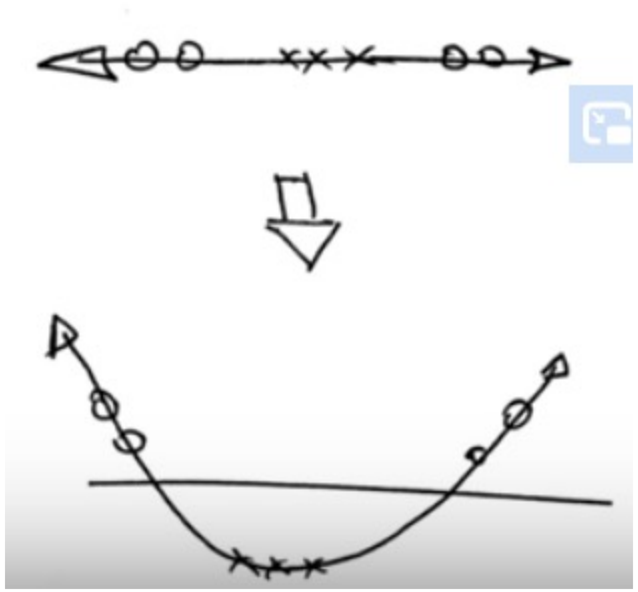
Kernels



In this new space they are separable! So we can use SVM as a classifier for this 2D set too.

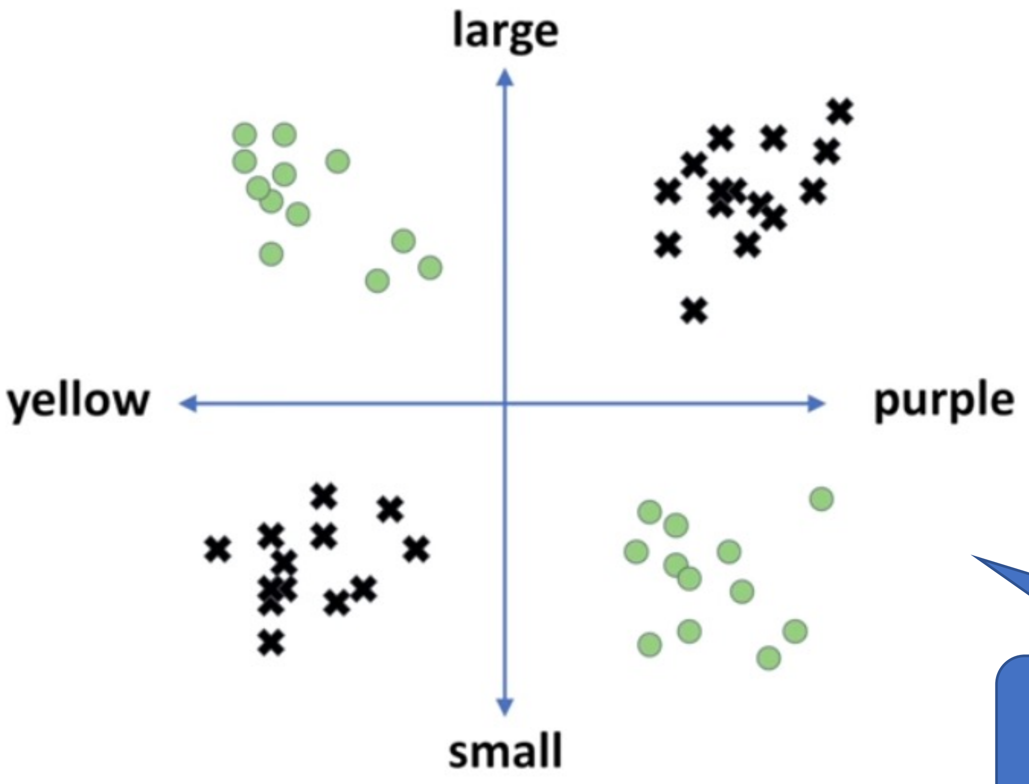
Kernels

Put $x = 0$ here and let $y = x^2$



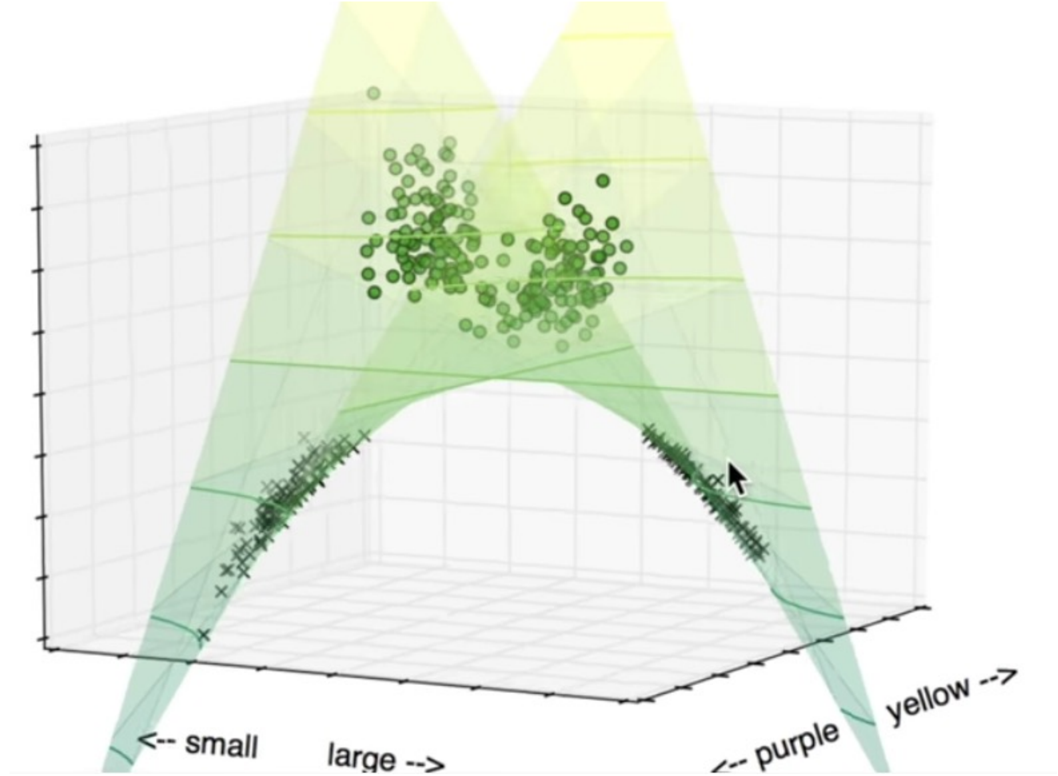
Kernels

Here we have the peaches again but now they have a size as well as a color and we've added plums



The kernel $z = x * y$ will make the two sets linearly separable!

Kernels



Now they are linearly separable. No problem to pay with more dimensions.
Kernel selection is trial and error (an art).

Evaluation of SVM

Advantages

- Works for classification as well as regression
- Works with high-dimensional space
- Works for two or more classes (via one-vs-rest strategy)
- Good accuracy

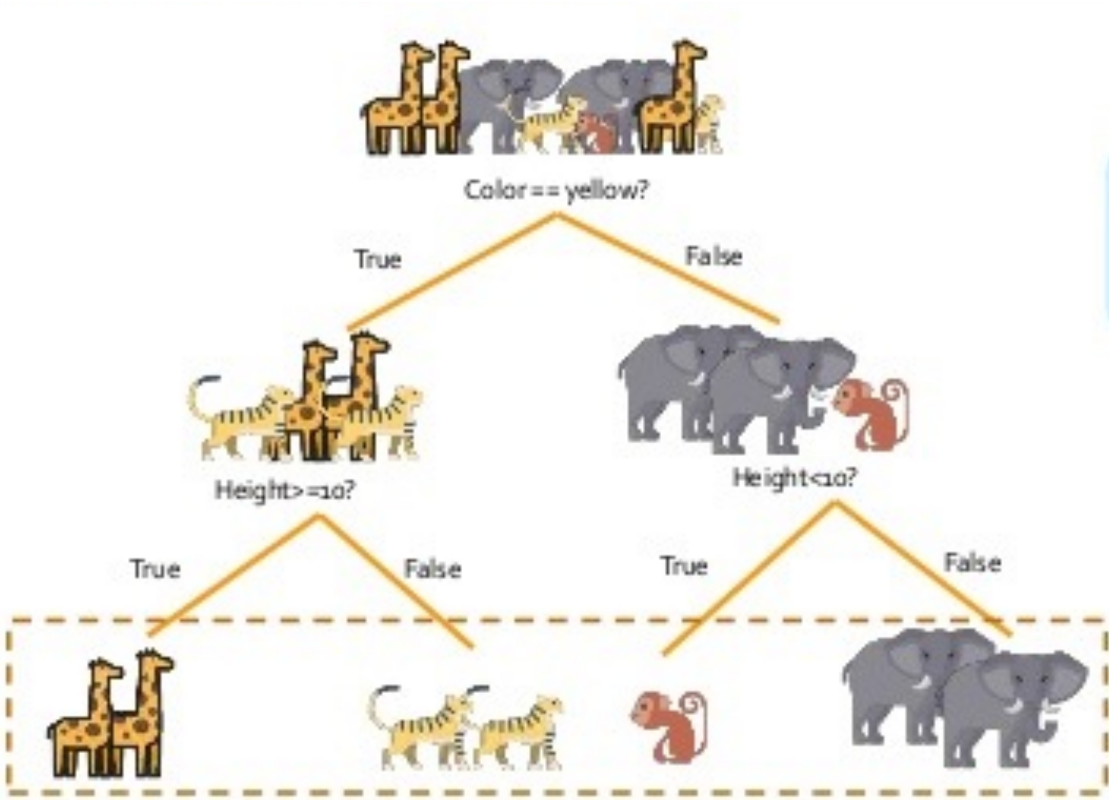
Disadvantages

- Slow on large datasets (compared to Naïve Bayes)
- Works poorly with overlapping classes
- Kernel type must be selected manually.

Decision trees

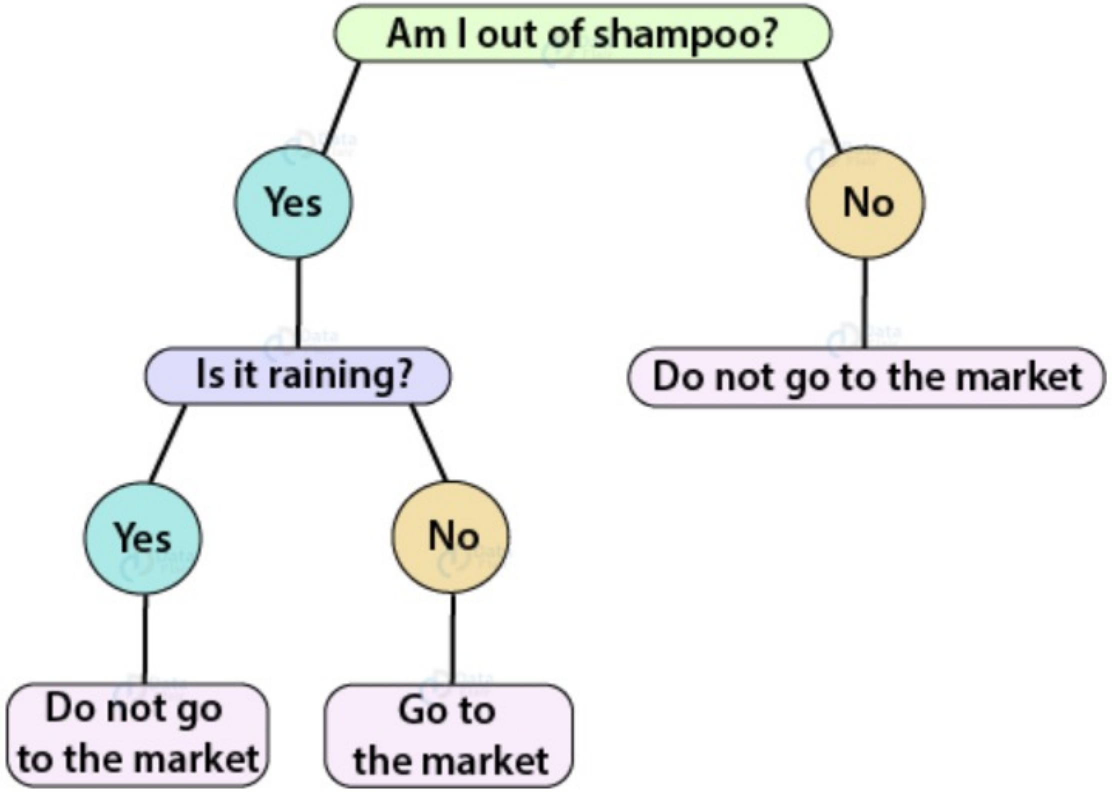
StatQuest

Example



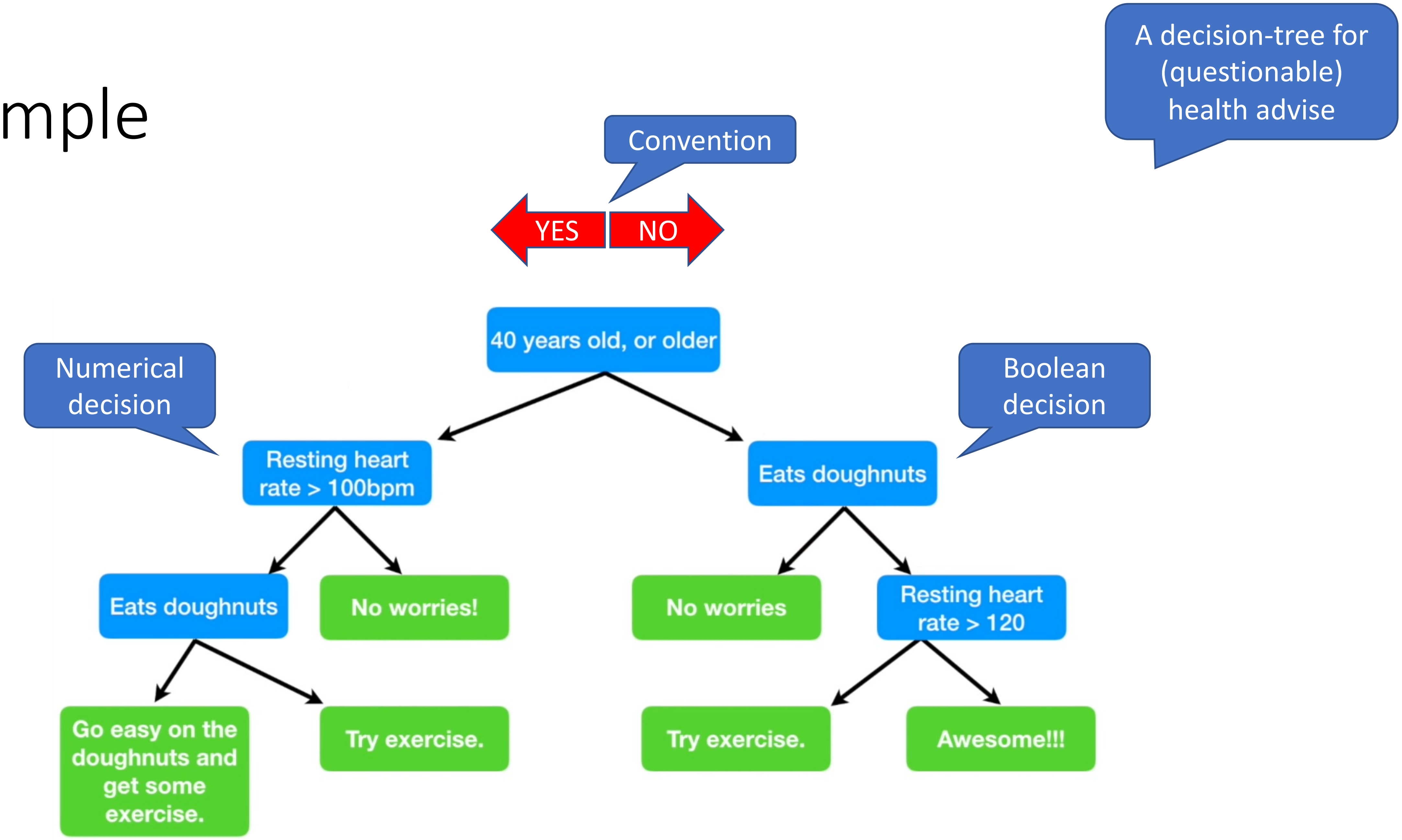
Classification of animals

Example



[Classification of activities]

Example


A decision-tree for (questionable) health advise
Convention
YES
NO
40 years old, or older
Numerical decision
Boolean decision
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!!!

From data to decision-tree

Feature 1

Feature 2

Target

Patient 1

Patient 2

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...

Decision-trees can be made in many ways. One way is to ask Q1 at the root, Q2 everywhere at the next level, and so on, until the labeled leaves. How many leaves will such a tree have?

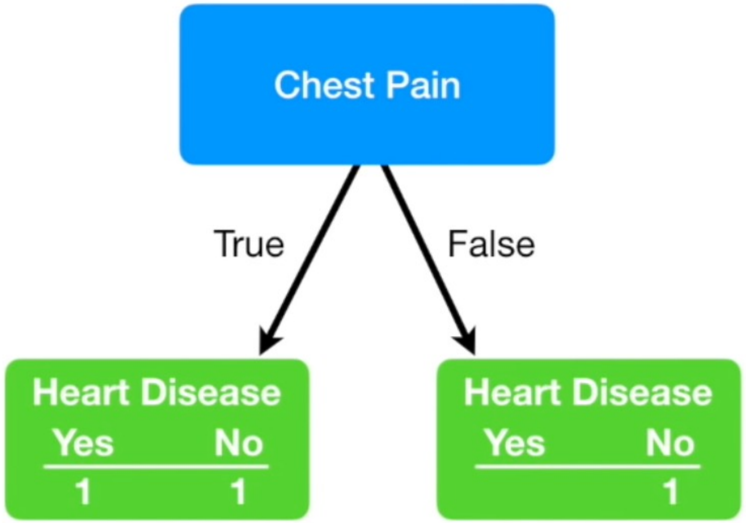
Big trees are bad for generalization + may be expensive to use. (Tests may cause suffering and be expensive.) Making small trees requires a method that finds regularities.

Feature analysis

Which split (or question or feature) should we start with? Let's consider Chest Pain. We calculate the effect of this split in terms of the target variable Heart Disease.

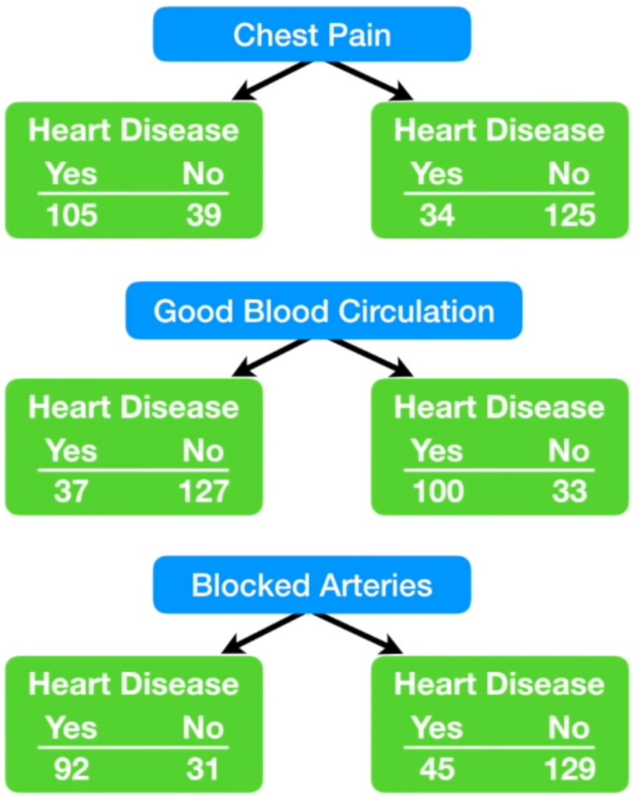
The 4th patient has chest pain and heart disease.

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...



Go through all the patients (rows) and analyze Chest pain vs Heart Disease

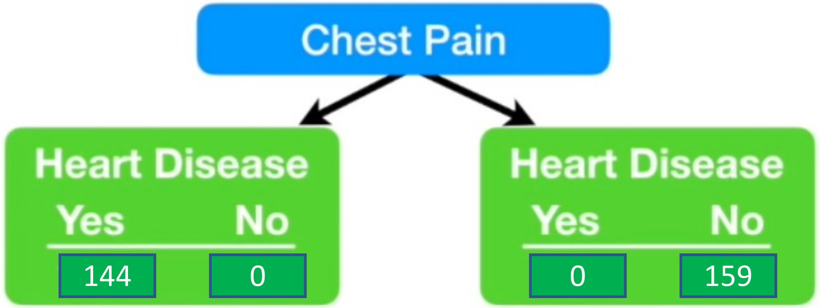
Feature analysis



Result of this analysis for all patients and all three splits (columns). Which split is best to start with?

Wishful thinking…

Gini impurity



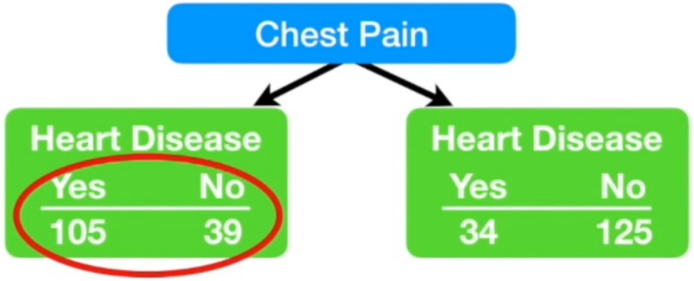
Pure sets like this would be ideal, since then we would not need to ask any more questions. The probability of Yes is 0 or 1.

Maybe we could define a purity measure that measures to what extent a question splits the data into pure Yes or pure No sets?

Equivalently we could define a measure of impurity. We will use a common one called the *Gini impurity*. Another common one is Shannon's notion of *entropy*.

Gini impurity of sets

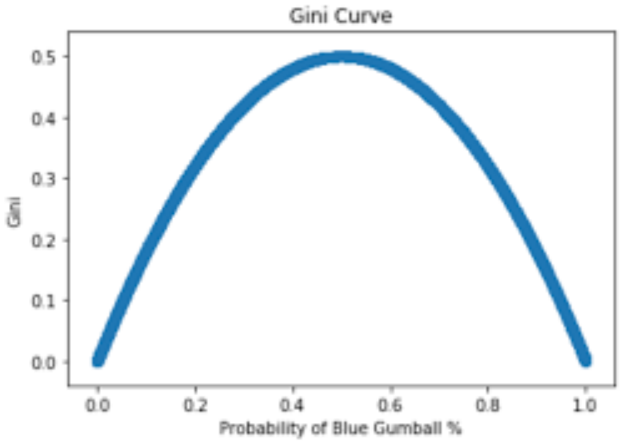
If the probability of Yes is 1, then the probability of No is 0, so Gini = 0



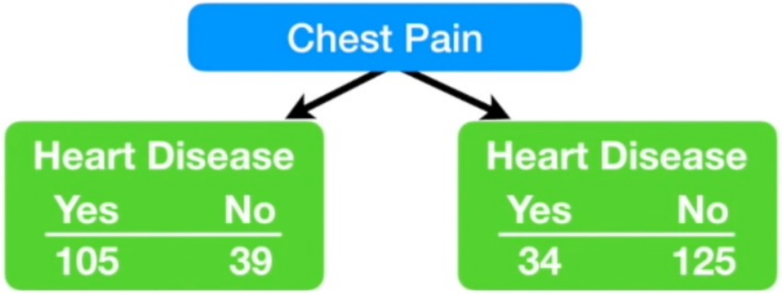
For this leaf, the Gini impurity $= 1 - (\text{the probability of “yes”})^2 - (\text{the probability of “no”})^2$

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

If the probability of Yes is 0.5, then the probability of No is 0.5, so Gini = 0.5



Gini impurity of splits



Gini impurity = 0.395 0.336

Gini impurity for Chest Pain = weighted average of Gini impurities for the child sets

$$= \left(\frac{144}{144+159}\right) 0.395 + \left(\frac{159}{144+159}\right) 0.336$$

$$= 0.364$$

With this probability we have that impurity

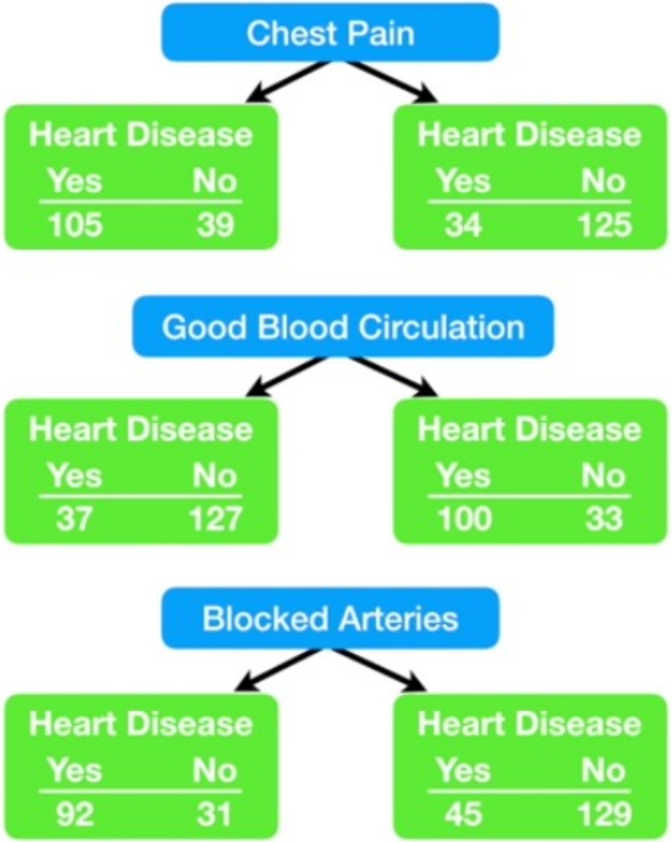
Selecting the best split

Gini impurity for Chest Pain = 0.364

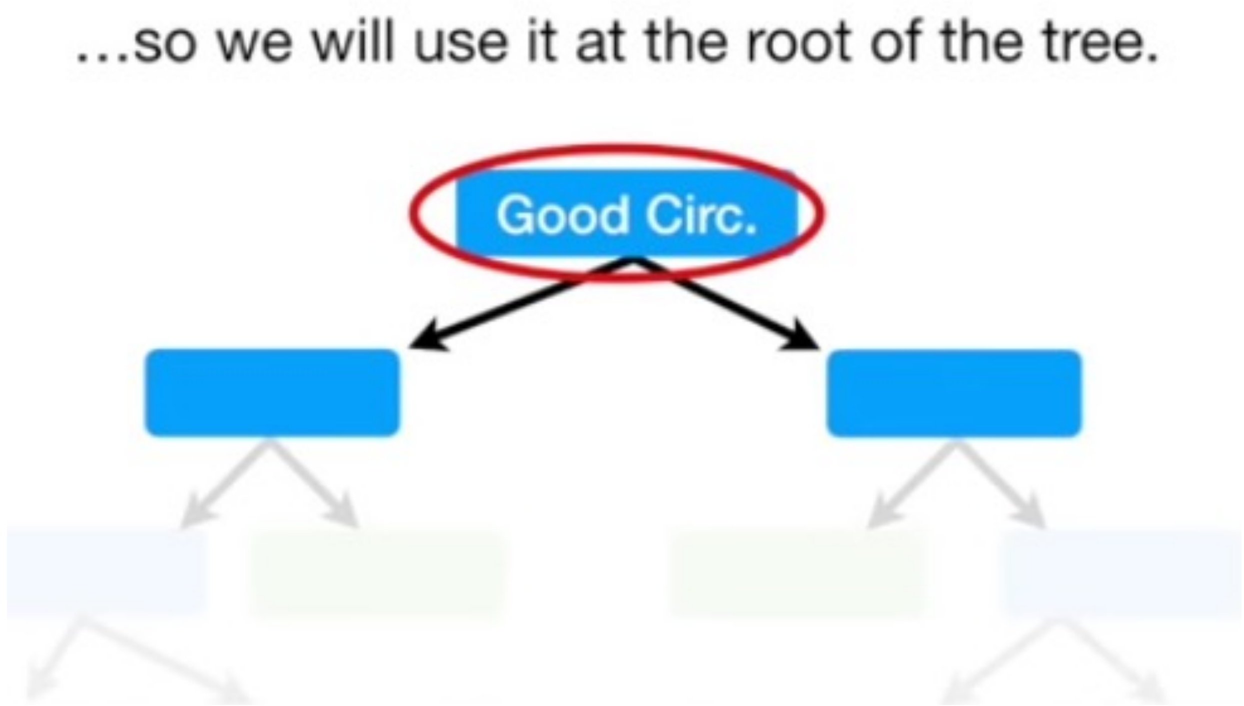
WINNER!

Gini impurity for Good Blood Circulation = 0.360

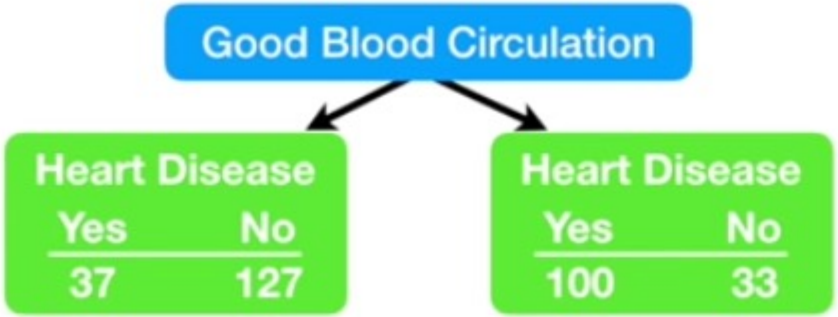
Gini impurity for Blocked Arteries = 0.381

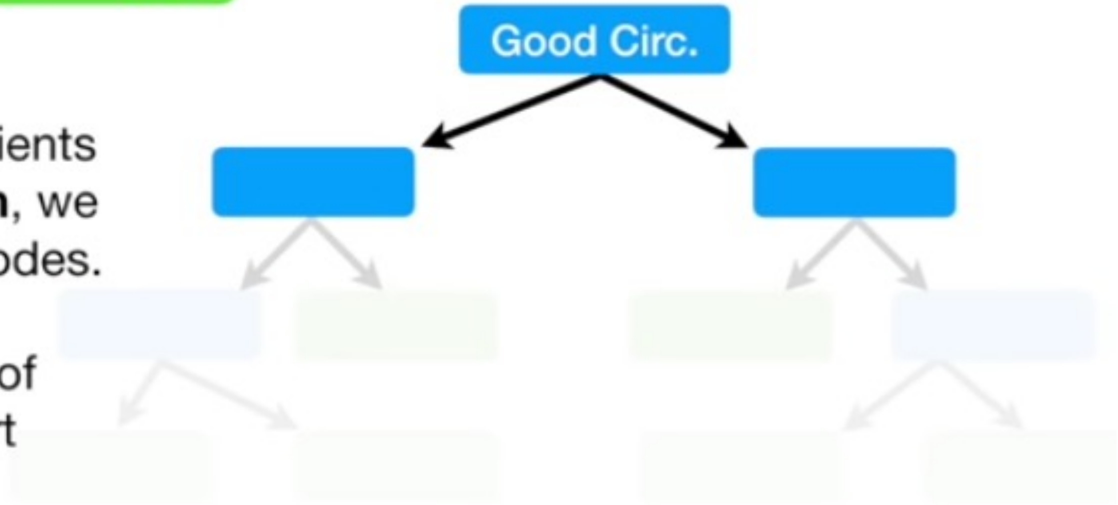


Tree construction



Tree construction





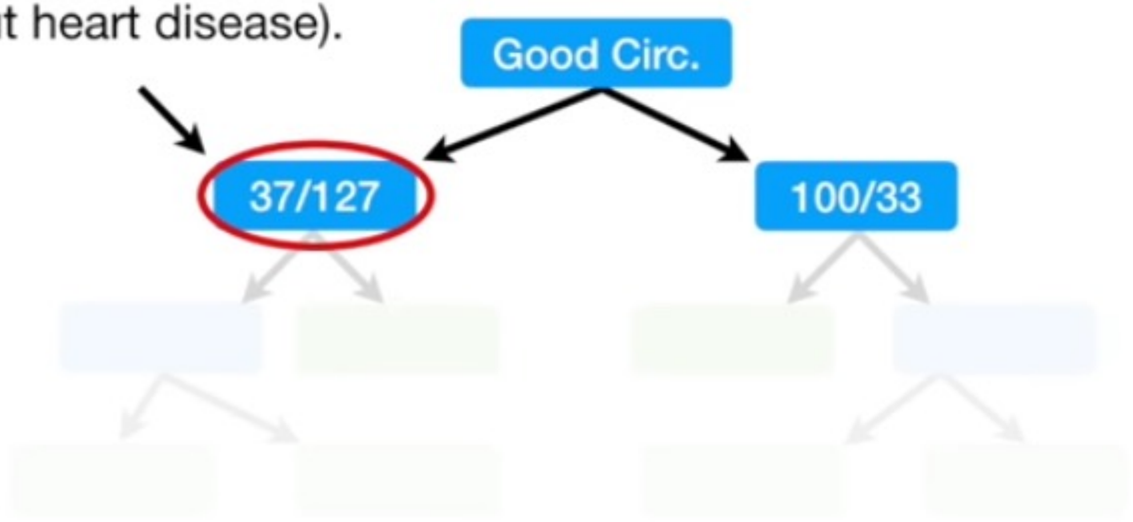
When we divided all of the patients using **Good Blood Circulation**, we ended up with “impure” leaf nodes.

Each leaf contained a mixture of patients with and without Heart Disease.

Let’s continue the construction of the tree!

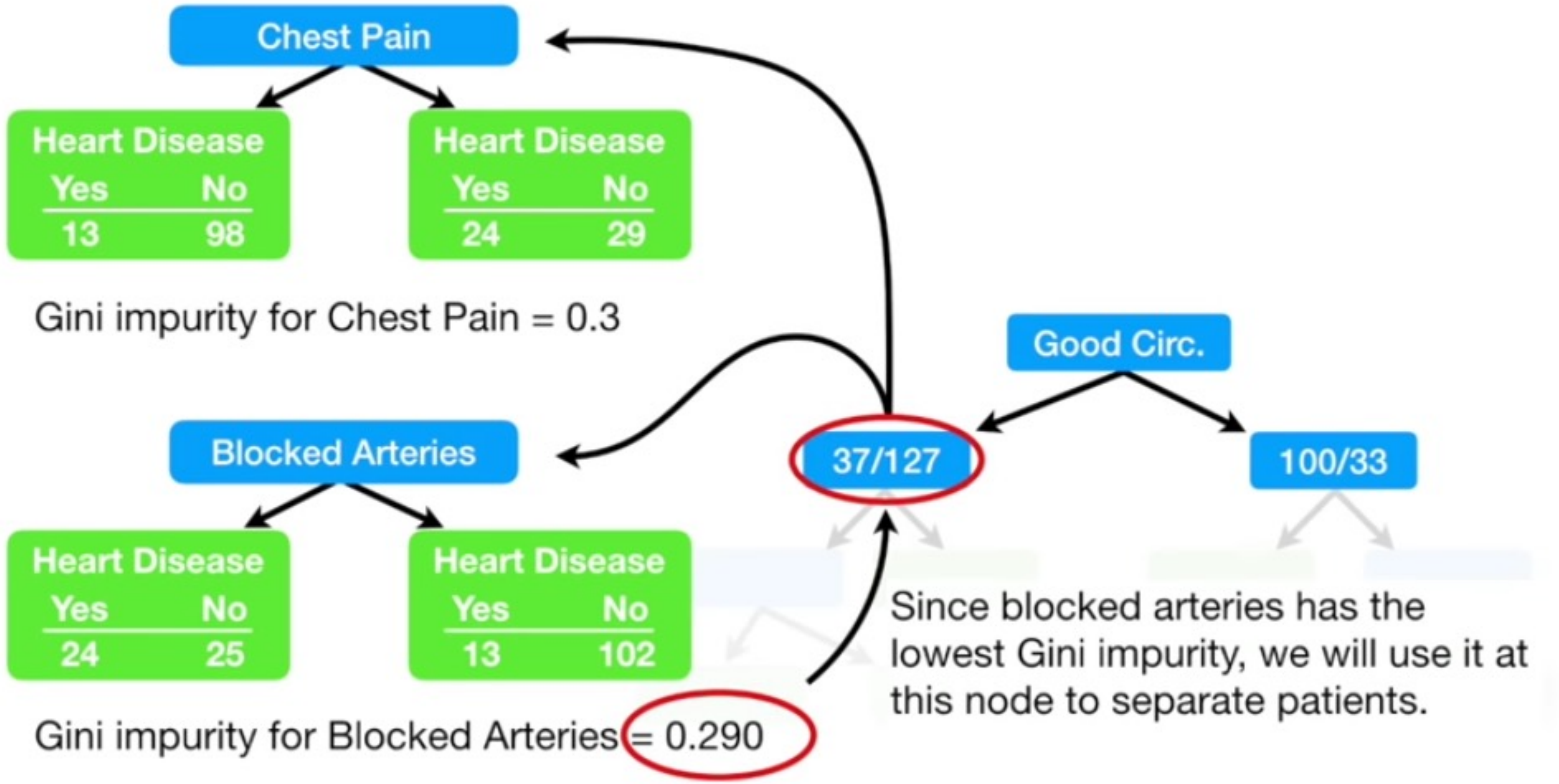
Tree construction

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).

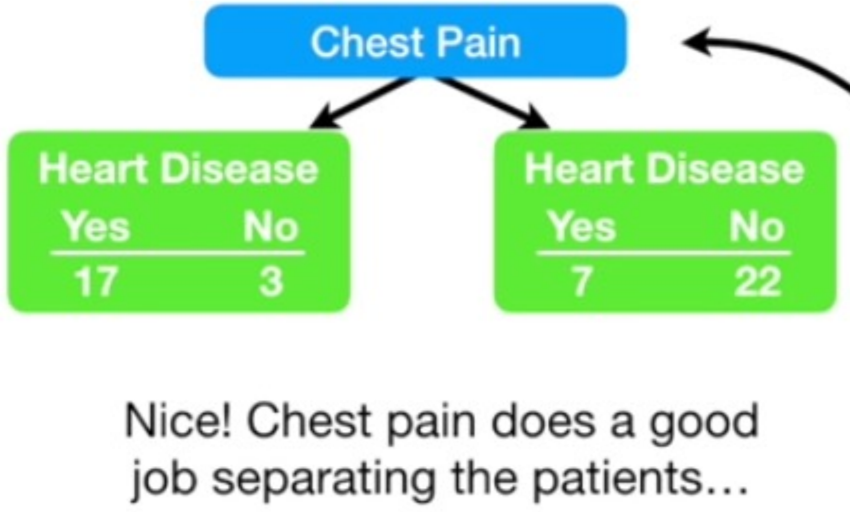


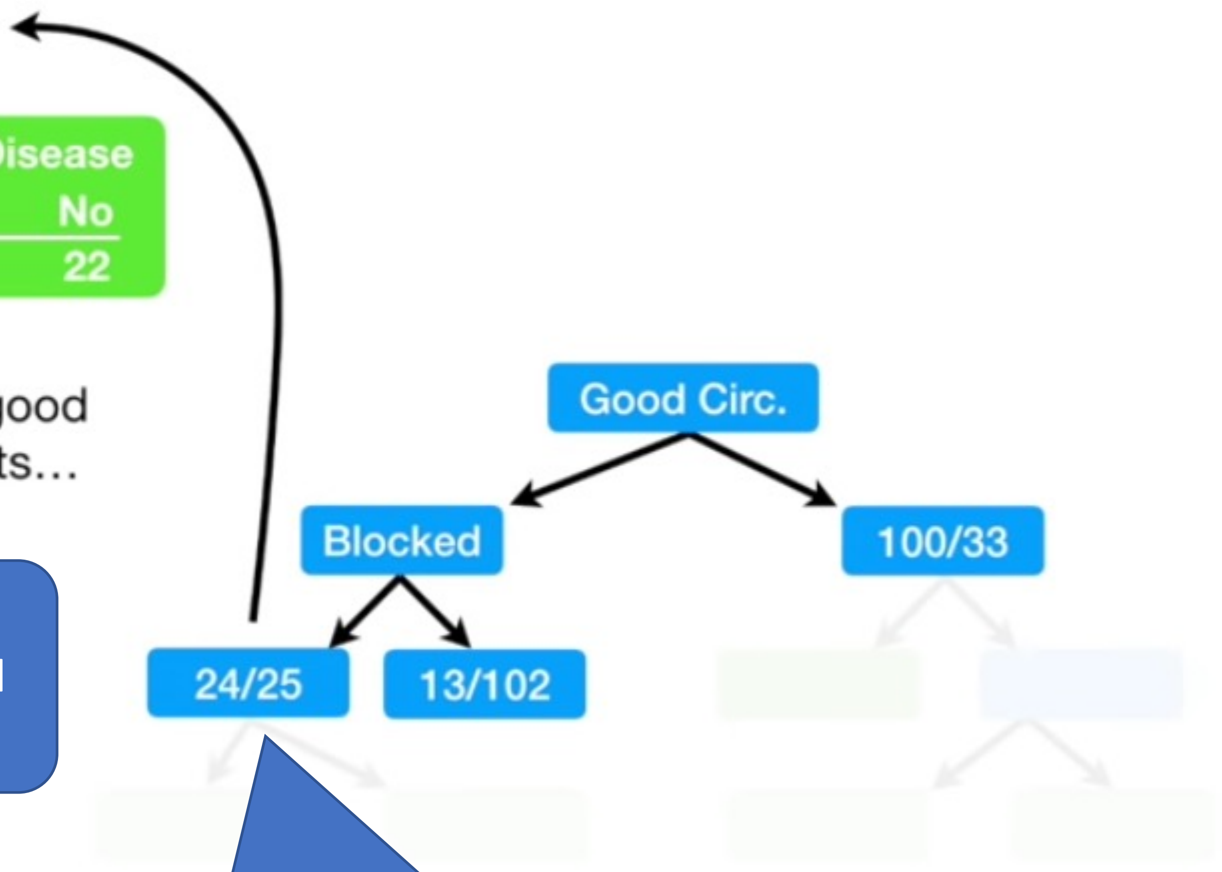
Tree construction

Select the split with the lowest Gini impurity again



Tree construction

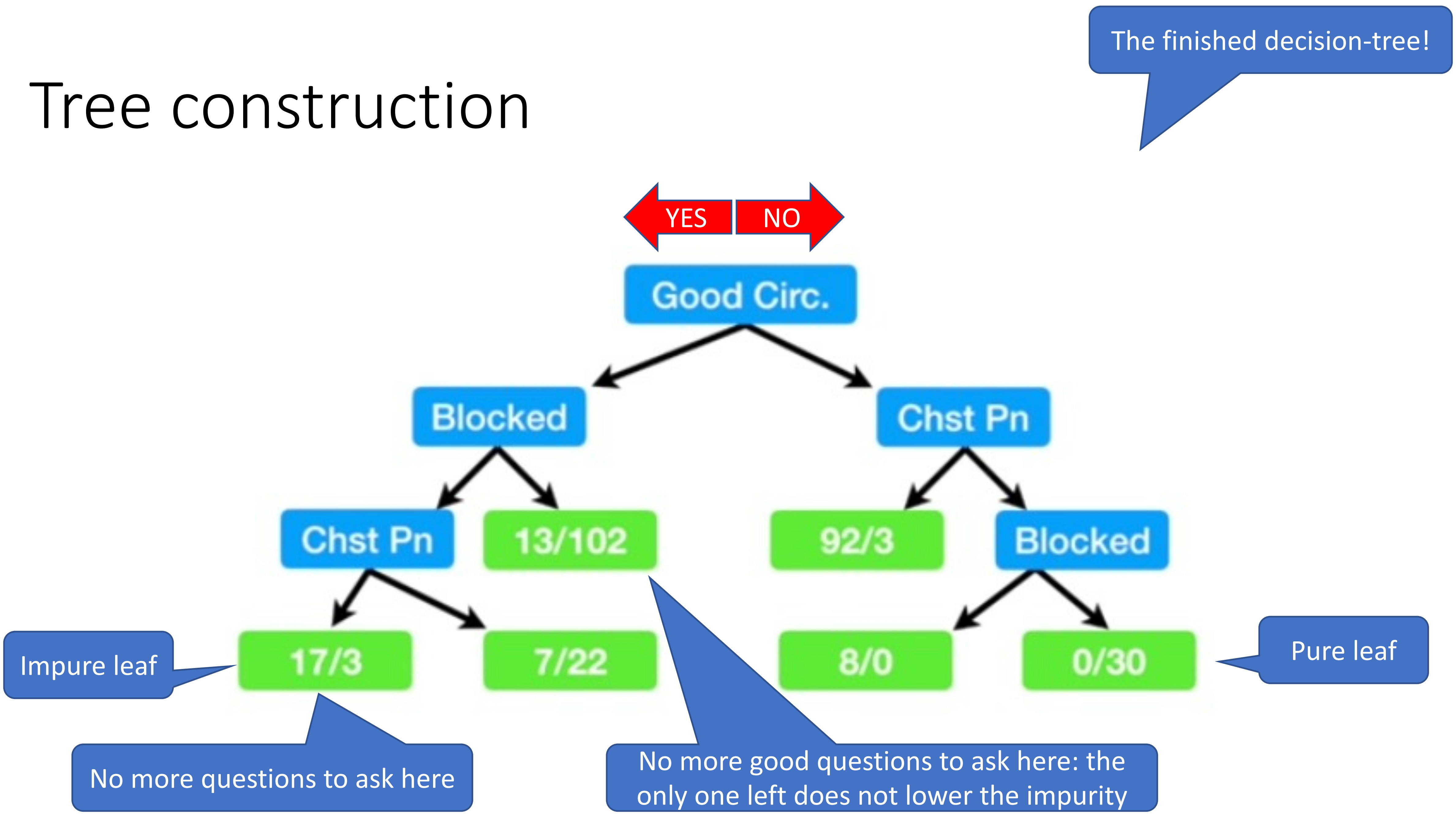




But not a perfect job! The three features did not lead to a perfect separation

Only one question left here: Chest Pain

Tree construction



Numerical decisions

Now suppose we also have column with numerical data, e.g. Weight.

Then we may add decisions of the form Weight ≥ 180 or the like

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

Numerical decisions

Start by sorting the data by Weight

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

Then compute the Gini impurities of Weight ≥ x for different x (that appear in the column).

Finally select the best split among all splits like before!

Decision tree applications

- **Marketing and Sales –** Decision Trees play an important role in a decision-oriented sector like marketing. In order to understand the consequences of marketing activities, organizations make use of Decision Trees to initiate careful measures. This helps in making efficient decisions that help the company to reap profits and minimize losses.
- **Reducing Churn Rate –** Banks make use of Decision Trees to retain their customers. It is always cheaper to keep customers than to gain new ones. Banks are able to analyze which customers are more vulnerable to leaving their business. Based on the output, they are able to *make decisions by providing better services, discounts as well as several other features.*
- **Anomaly & Fraud Detection –** Industries like finance and banking suffer from various cases of fraud. In order to filter out anomalous or fraud loan applications, information and insurance fraud, these companies deploy decision trees to provide them with the necessary information to identify fraudulent customers.
- **Medical Diagnosis –** Classification trees identifies patients who are at risk of suffering from serious diseases such as cancer and diabetes.

https://data-flair.training/blogs/r-decision-trees/

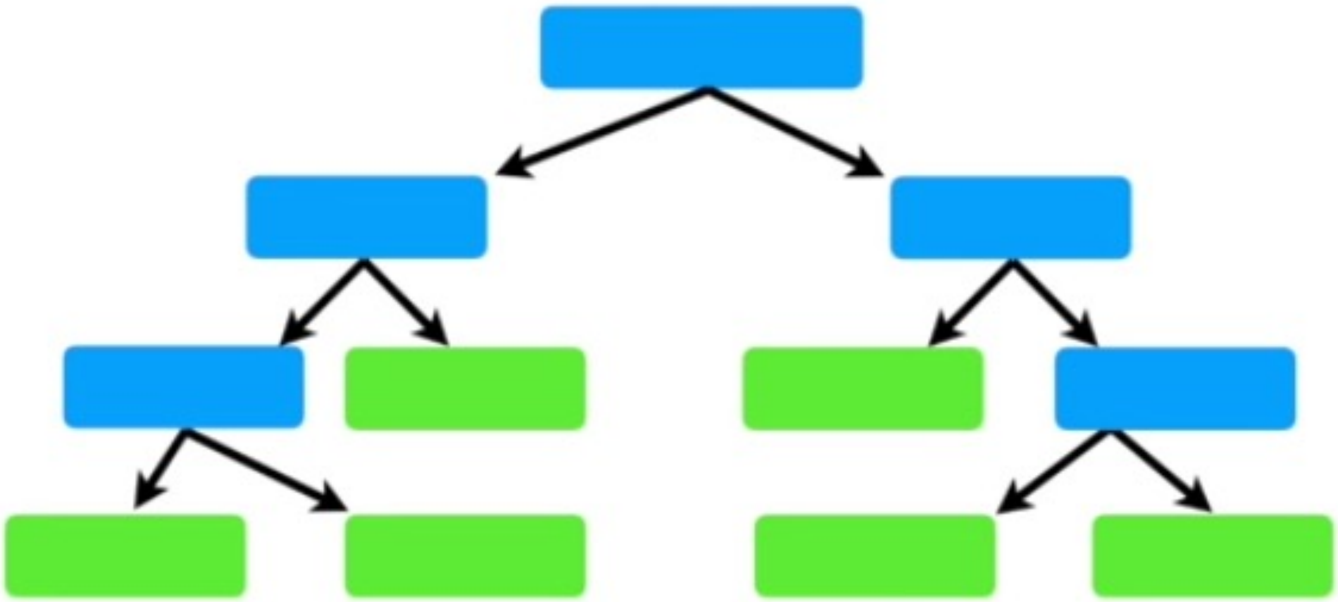
Random Forests

StatQuest

Advantages

Decision trees

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



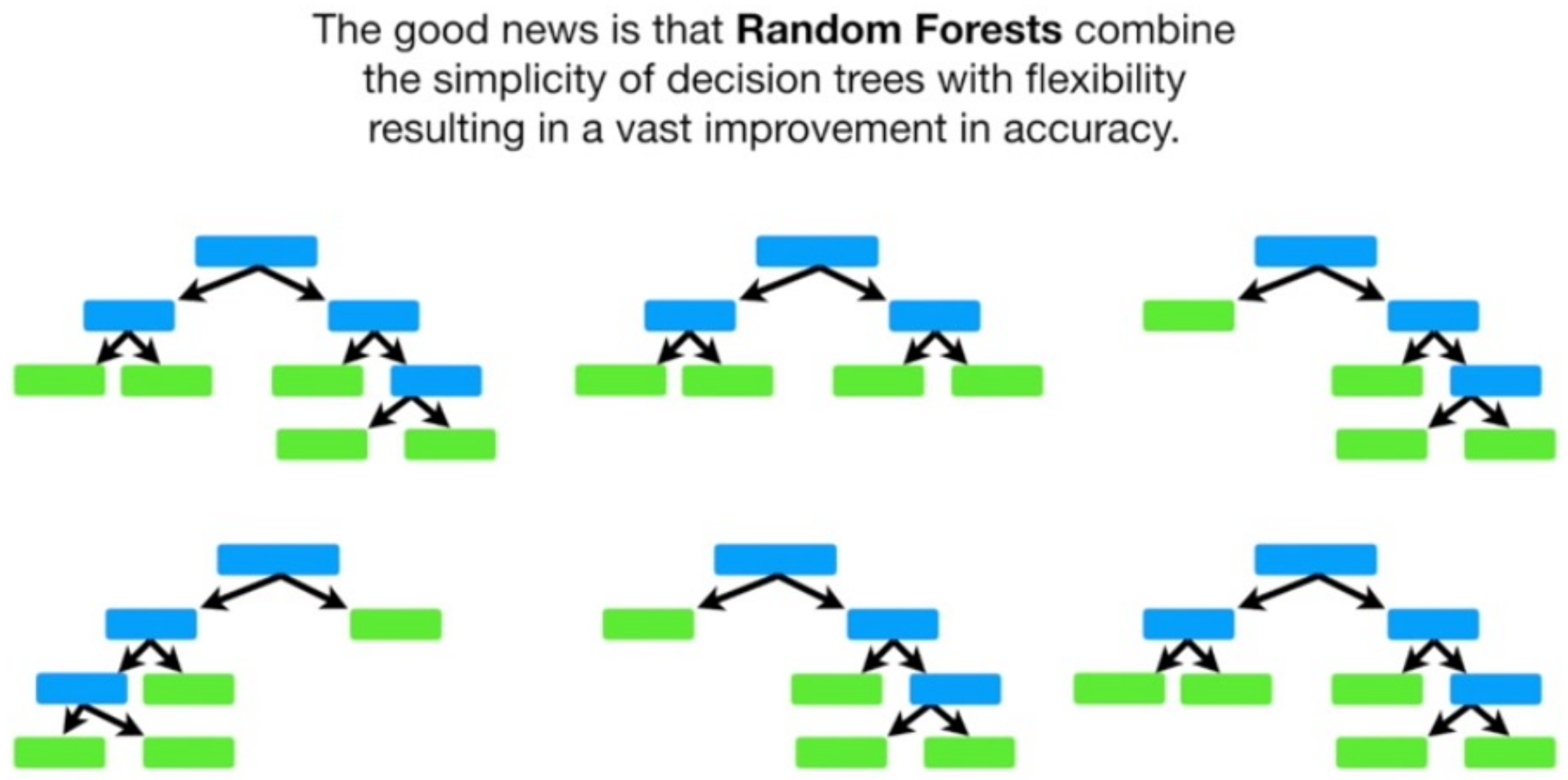
Decision trees

Drawbacks

To quote from ***The Elements of Statistical Learning*** (aka The Bible of Machine Learning), “Trees have one aspect that prevents them from being the ideal tool for predictive learning, namely **inaccuracy**.”

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**.

Random forests



Building random forests

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

Let us use this dataset again and build a random forest!

Building random forests

Step 1: create a bootstrapped dataset by randomly picking lines from the original 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

Same line twice!

Building random forests

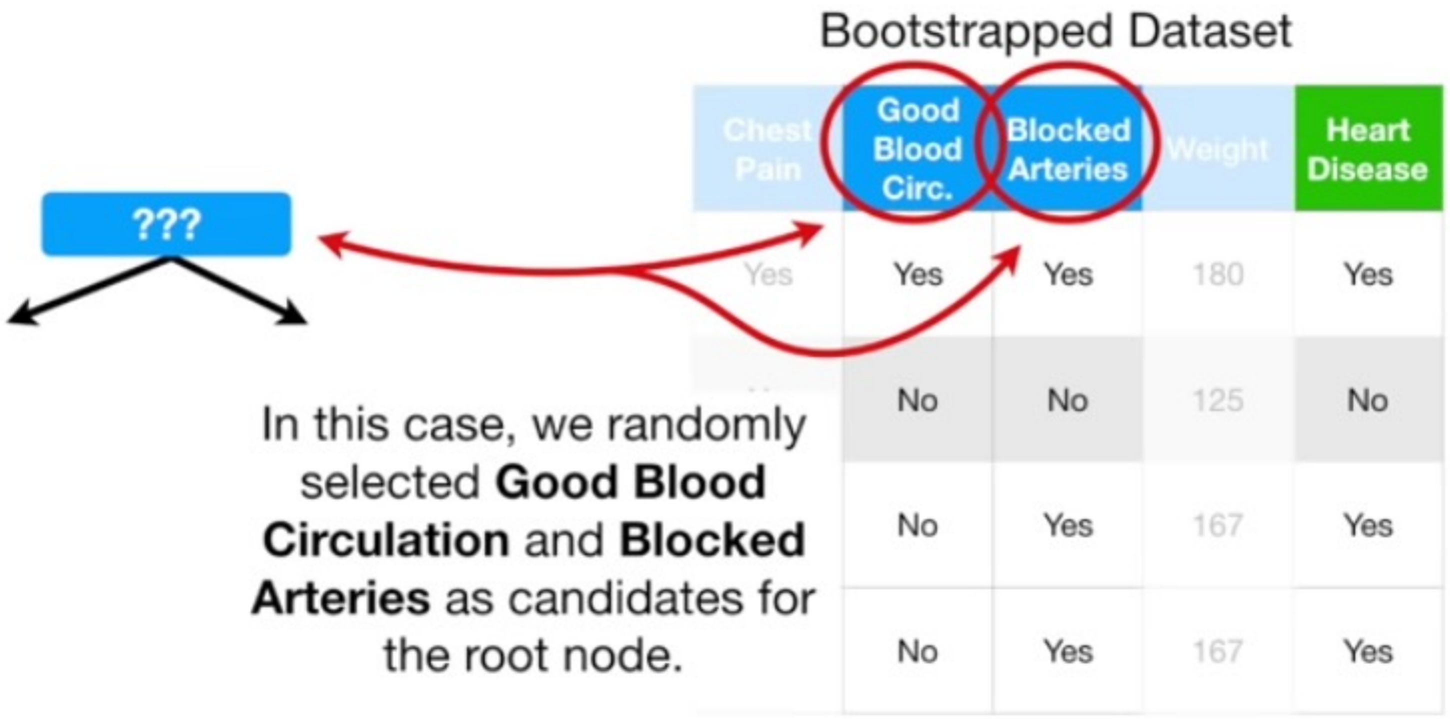
Step 2: Create a decision tree using the bootstrapped dataset, but only use a random subset of variables (or columns) at each step.

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

Building random forests



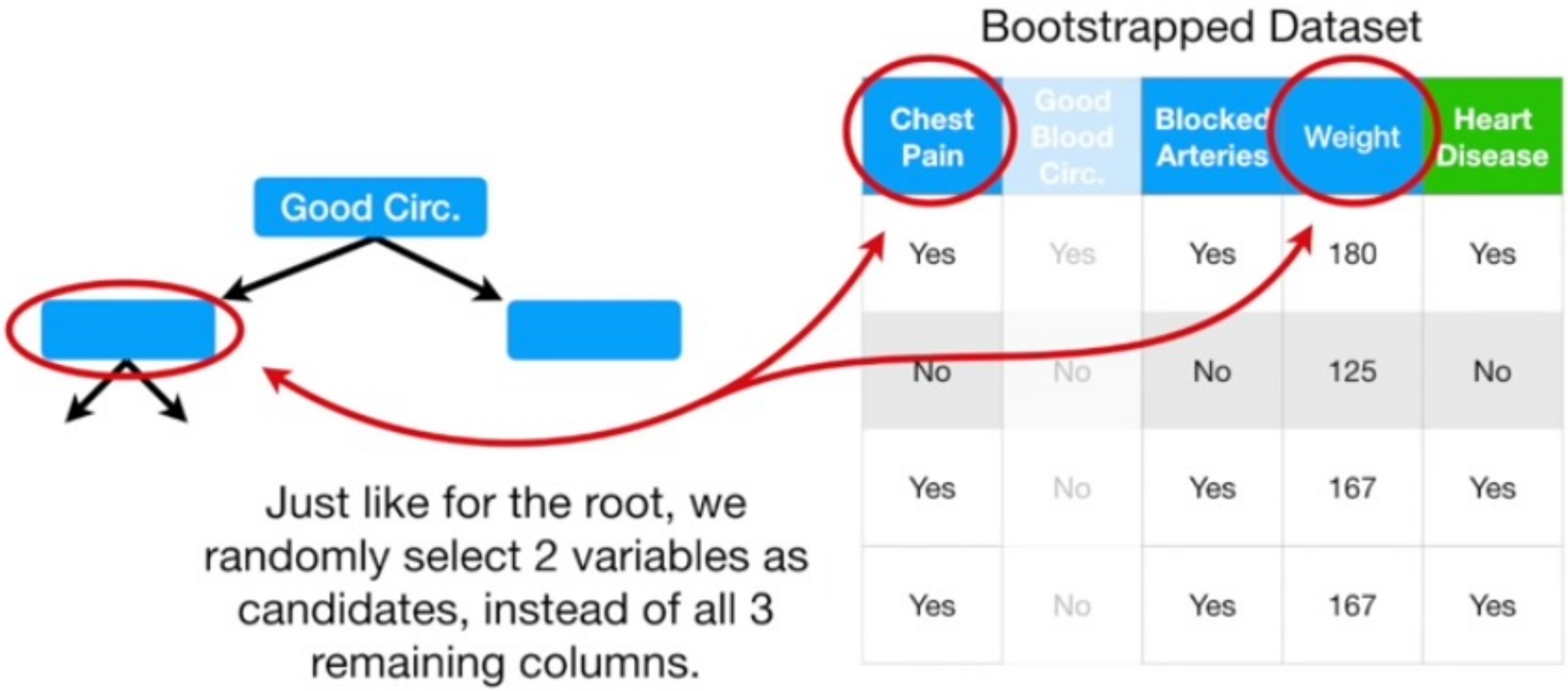
Building random forests

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

Building random forests



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

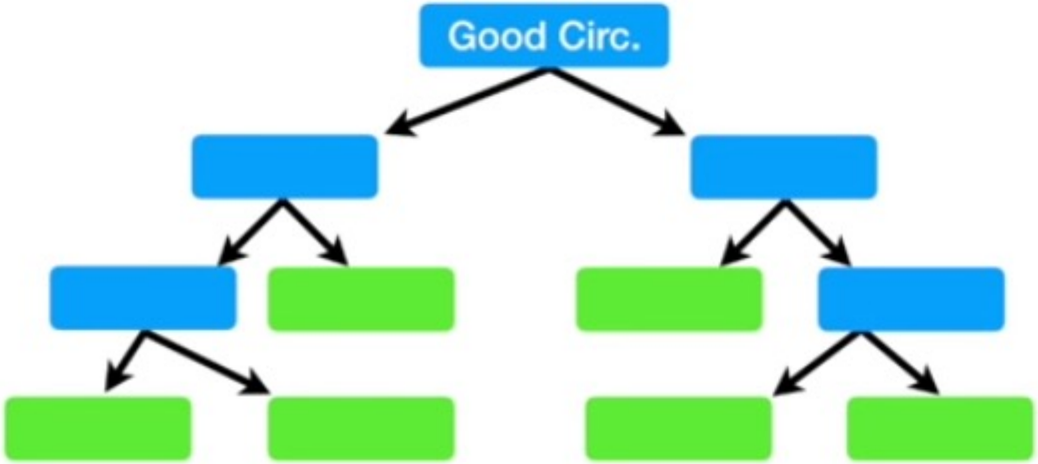
Continue as usual, but select features from a random subset!

Random forests

We built a tree…

1) Using a bootstrapped dataset

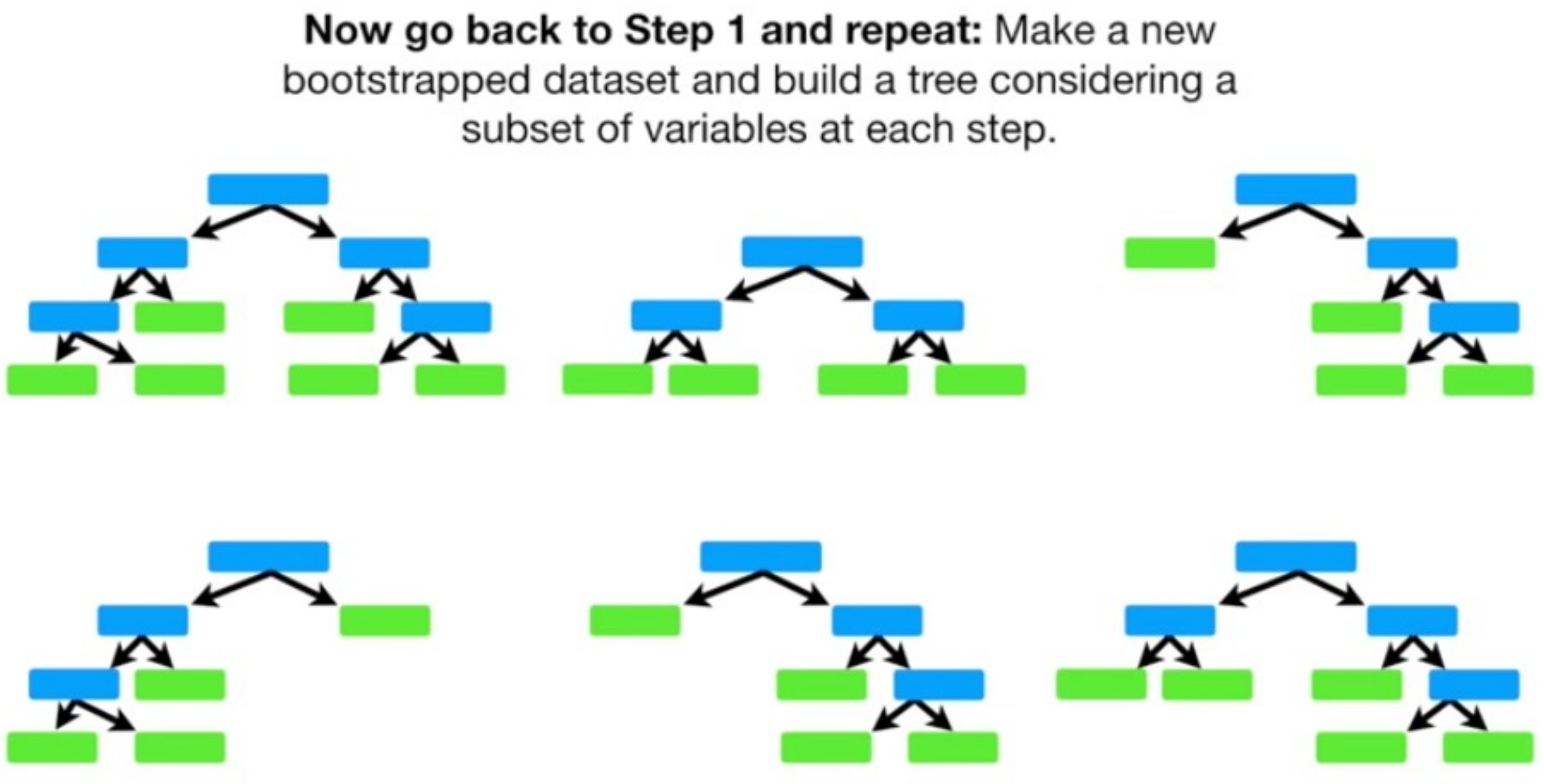
2) Only considering a random subset of variables at each step.



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

Building random forests



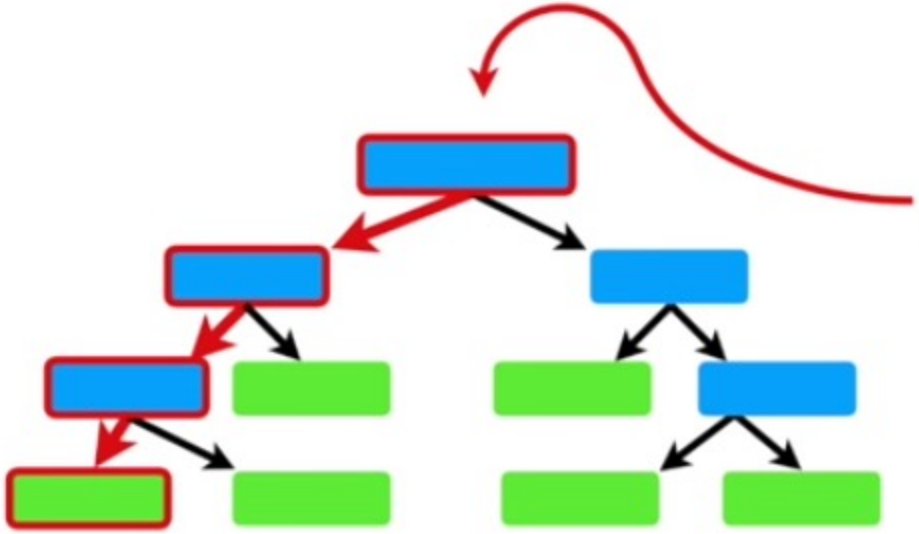
Using random forests

Suppose we have this new patient…

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

…and now we want to know if they have heart disease or not.

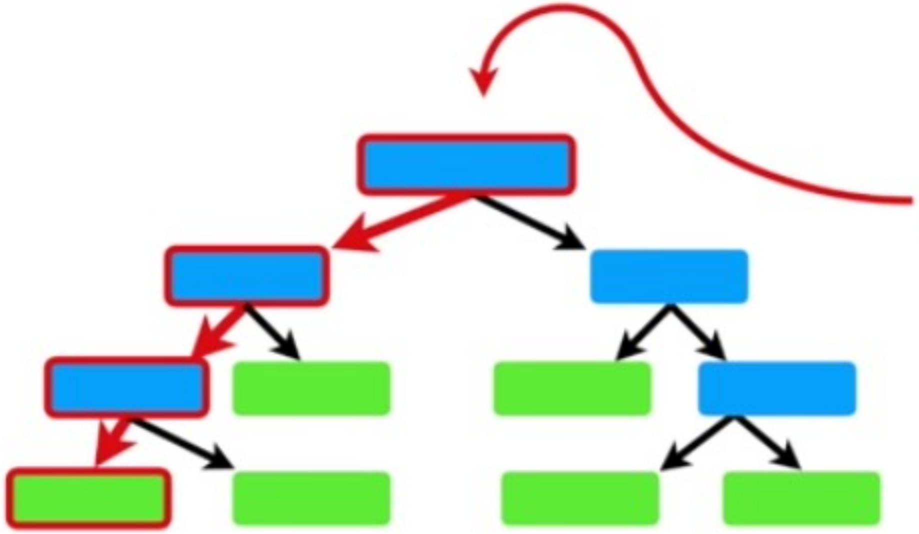
Using random forests



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

The first tree says "Yes"...

Using random forests



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

Keeping track of the votes

Using random forests

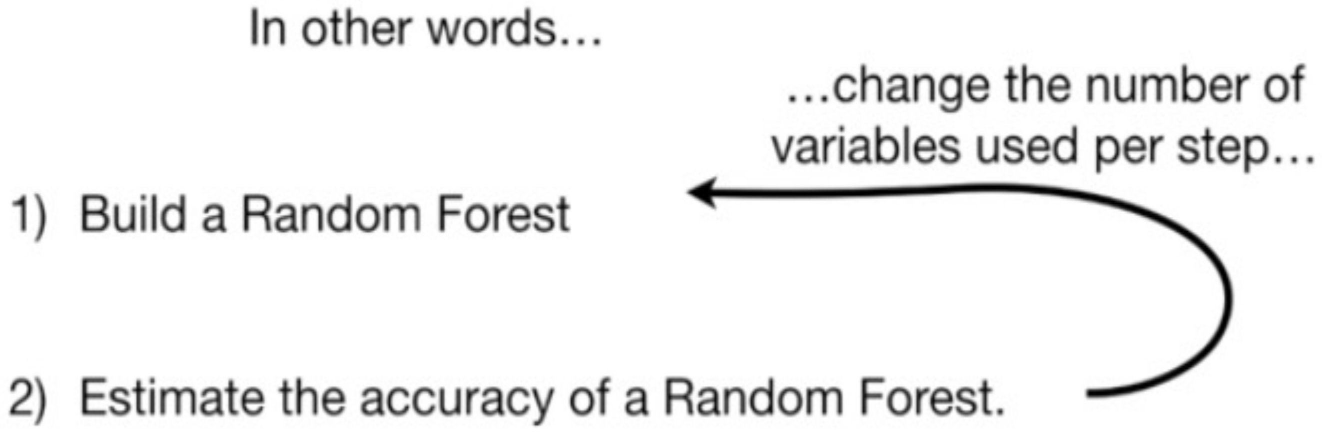
Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	No	No	168	**YES**

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

Heart Disease	
Yes	No
5	1

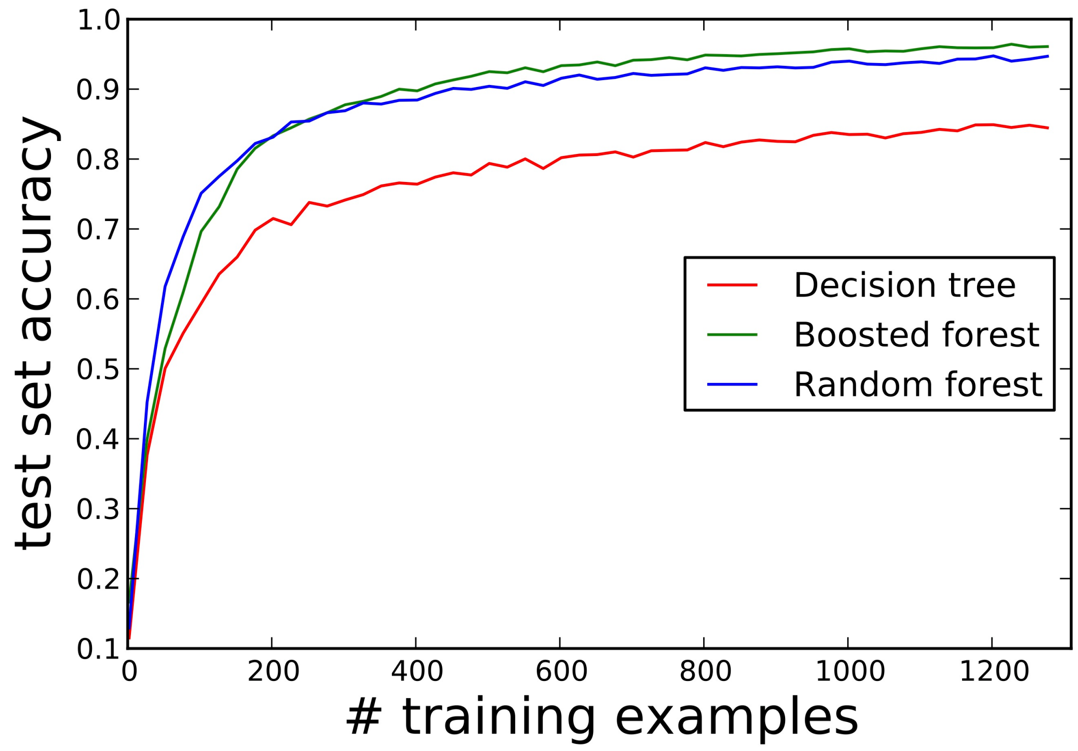
All 6 trees have voted! Then we make an aggregated decision.

Using random forests



Typically, start using the square root of the number of variables and then try a few more nearby values.

Forests vs. trees



Always evaluate accuracy on datapoints that were not used in the model construction (unseen datapoints). For example separated from the beginning or not included in the Bootstrap sets.

Random forests

Advantages:

- Random forests is considered as a highly accurate and robust method because of the number of decision trees participating in the process.
- It does not suffer from the overfitting problem. The main reason is that it takes the average of all the predictions, which cancels out the biases.

https://www.datacamp.com/community/tutorials/random-forests-classifier-python

Random forests

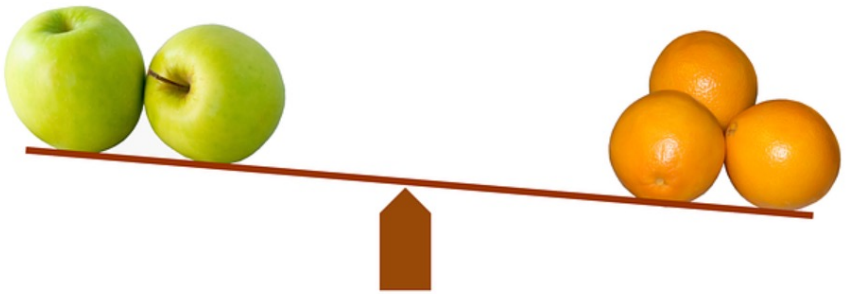
Disadvantages:

- Random forests is slow in generating predictions because it has multiple decision trees. Whenever it makes a prediction, all the trees in the forest have to make a prediction for the same given input and then perform voting on it. This whole process is time-consuming.
- The model is difficult to interpret compared to a decision tree, where you can easily make a decision by following the path in the tree.

https://www.datacamp.com/community/tutorials/random-forests-classifier-python

Comparing classification methods

- Accuracy
- Scalability
- Manual steps
- Multiclass or two-class
- Interpretability/Explainability



Best accuracy: random forest and SVM. Deep learning not tested.

[Comparison of 14 different families of classification algorithms on 115 binary datasets]

Comparing classification methods

Search this file...

	Gender	Age	Salary	Purchased Iphone
1	Gender	Age	Salary	Purchased Iphone
2	Male	19	19000	0
3	Male	35	20000	0
4	Female	26	43000	0
5	Female	27	57000	0
6	Male	19	76000	0
7	Male	27	58000	0
8	Female	27	84000	0
9	Female	32	150000	1

```
Logistic Regression: Mean Accuracy = 82.75% — SD Accuracy = 11.37%
K Nearest Neighbor: Mean Accuracy = 90.50% — SD Accuracy = 7.73%
Kernel SVM: Mean Accuracy = 90.75% — SD Accuracy = 9.15%
Naive Bayes: Mean Accuracy = 85.25% — SD Accuracy = 10.34%
Decision Tree: Mean Accuracy = 84.50% — SD Accuracy = 8.50%
Random Forest: Mean Accuracy = 88.75% — SD Accuracy = 8.46%
```

Classification Algorithms

Neural networks are not included in this comparison!