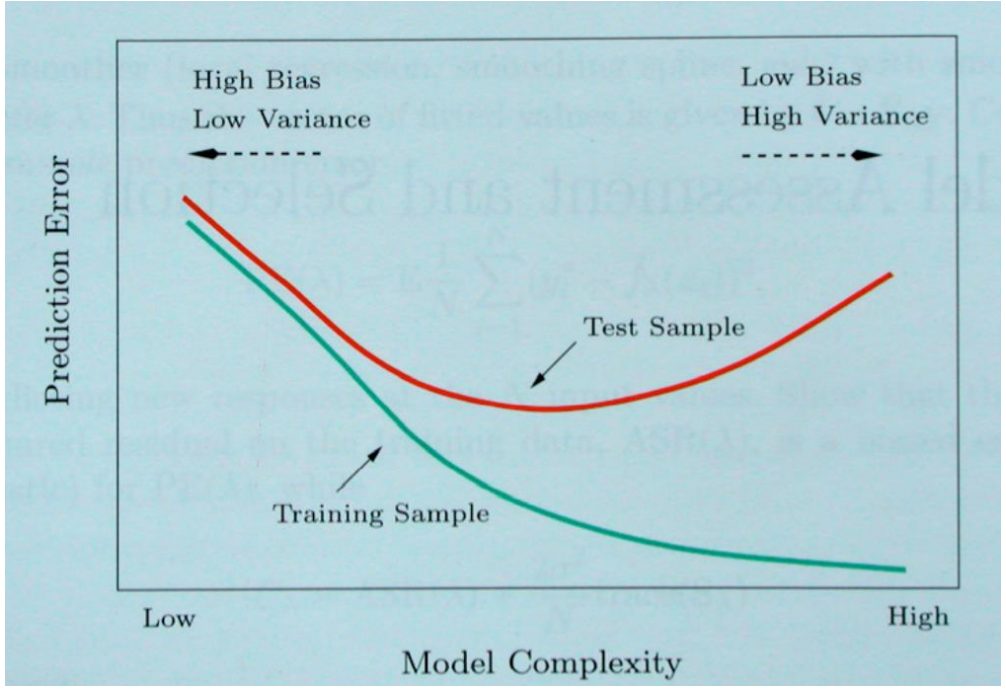
# CSE 445
# Lecture 14

Ensemble learning - 1: Voting, Bagging & Pasting, Random Forest Intro

# Bias/variance tradeoff

- Bias/Variance tradeoff



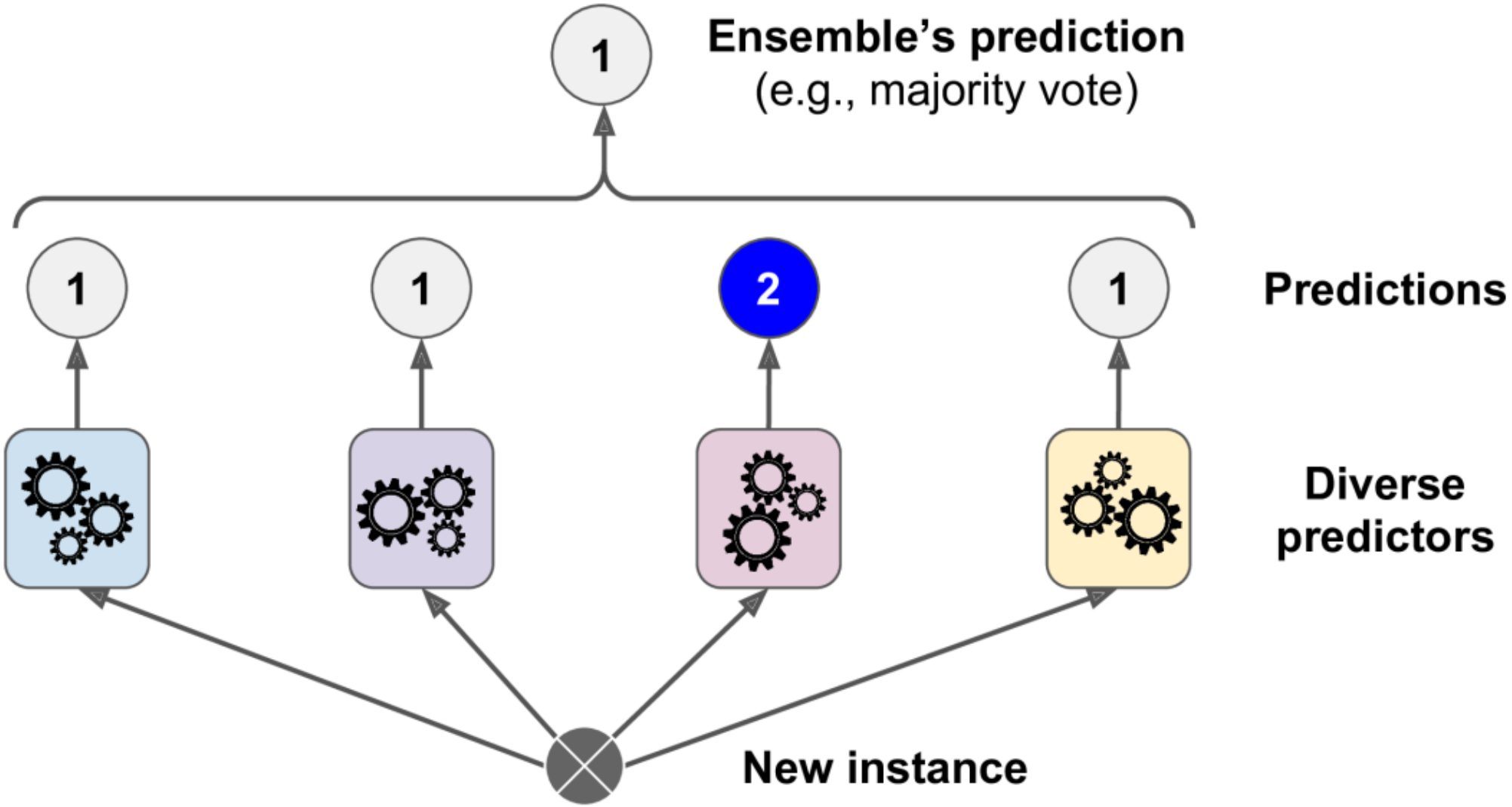
# Bias/variance tradeoff

- Can we reduce variance without increasing bias?
- Averaging reduces the variance as var($\bar{x}$) = var($x$)$/N$
- Therefore, if we average over models we can achieve our goal
- But
  - We have just one training sample
  - How can we get multiple models?

# Use different models: Voting classifiers

- Suppose we have trained a few classifiers, each one achieving about 80% accuracy
- For example, we can use a Logistic Regression classifier, an SVM classifier, a Random Forest classifier, a K-Nearest Neighbors classifier, and perhaps a few more
- A very simple way to create an even better classifier is to aggregate the predictions of each classifier and predict the class that gets the most votes
- This majority vote classifier is called a *hard voting* classifier

# Hard voting classifier



# Hard voting vs Soft voting

- In **hard voting** (also known as **majority voting**), every individual classifier votes for a class, and the majority wins
  - In statistical terms, the predicted target label of the ensemble is the mode of the distribution of individually predicted labels
- In **soft voting**, every individual classifier provides a probability value that a specific data point belongs to a particular target class
  - The predictions are weighted by the classifier's importance and summed up
  - Then the target label with the greatest sum of weighted probabilities wins the vote

# Manipulate the dataset: Bootstrapping

- In statistics, bootstrapping refers to a resample method that consists of repeatedly drawn, with replacement, samples from data to form other smaller datasets, called bootstrapping samples
- It's as if the bootstrapping method is making a bunch of simulations to our original dataset so in some cases we can generalize the mean and the standard deviation

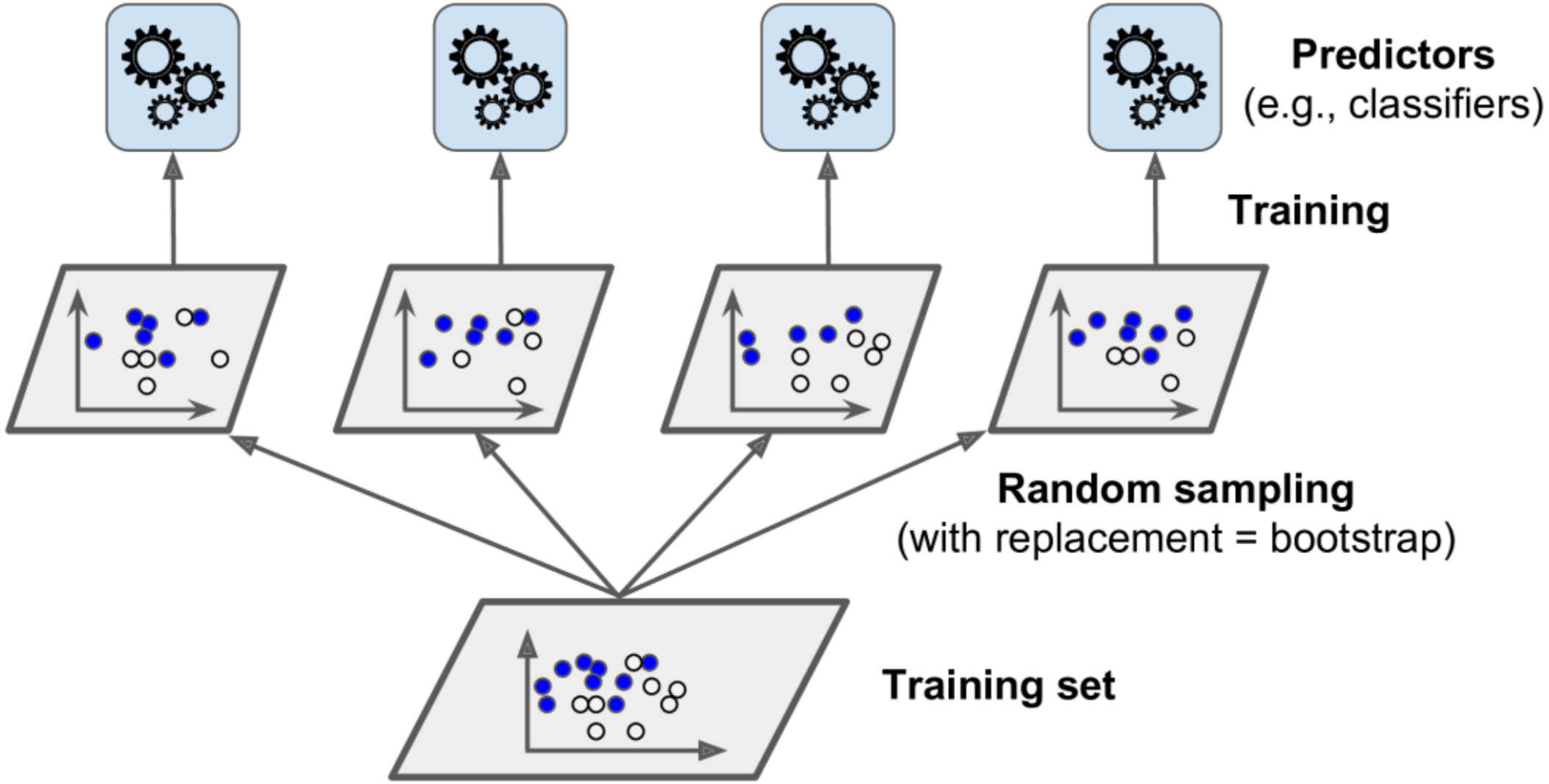
# Bootstrapping example

- For example, let's say we have a set of observations: [2, 4, 32, 8, 16]
- If we want each bootstrap sample containing *n* observations, the following are valid samples:
  - *n*=3: [32, 4, 4], [8, 16, 2], [2, 2, 2]…
  - *n*=4: [2, 32, 4, 16], [2, 4, 2, 8], [8, 32, 4, 2]…
- Since we drawn data with replacement, the observations can appear more than once in a single sample

# Bagging & Pasting: sampling the dataset

- One way to get a diverse set of classifiers is to use different training algorithms – as discussed for voting
- Another approach is to use the same training algorithm for every predictor, but to train them on different random subsets of the training set
- When sampling is performed *with* replacement, this method is called *bagging* (short for *bootstrap aggregating*)
- When sampling is performed *without* replacement, it is called *pasting*
- Both bagging and pasting allow training instances to be sampled several times across multiple predictors, but only bagging allows training instances to be sampled several times for the same predictor

# Bagging & Pasting



# Bagging & Pasting: Prediction

- Once all predictors are trained, the ensemble can make a prediction for a new instance by simply aggregating the predictions of all predictors
- The aggregation function is typically the *statistical mode* (i.e., the most frequent prediction, just like a hard voting classifier) for classification, or the average for regression
- Each individual predictor has a higher bias than if it were trained on the original training set, but aggregation reduces both bias and variance
- Generally, the net result is that the ensemble has a similar bias but a lower variance than a single predictor trained on the original training set

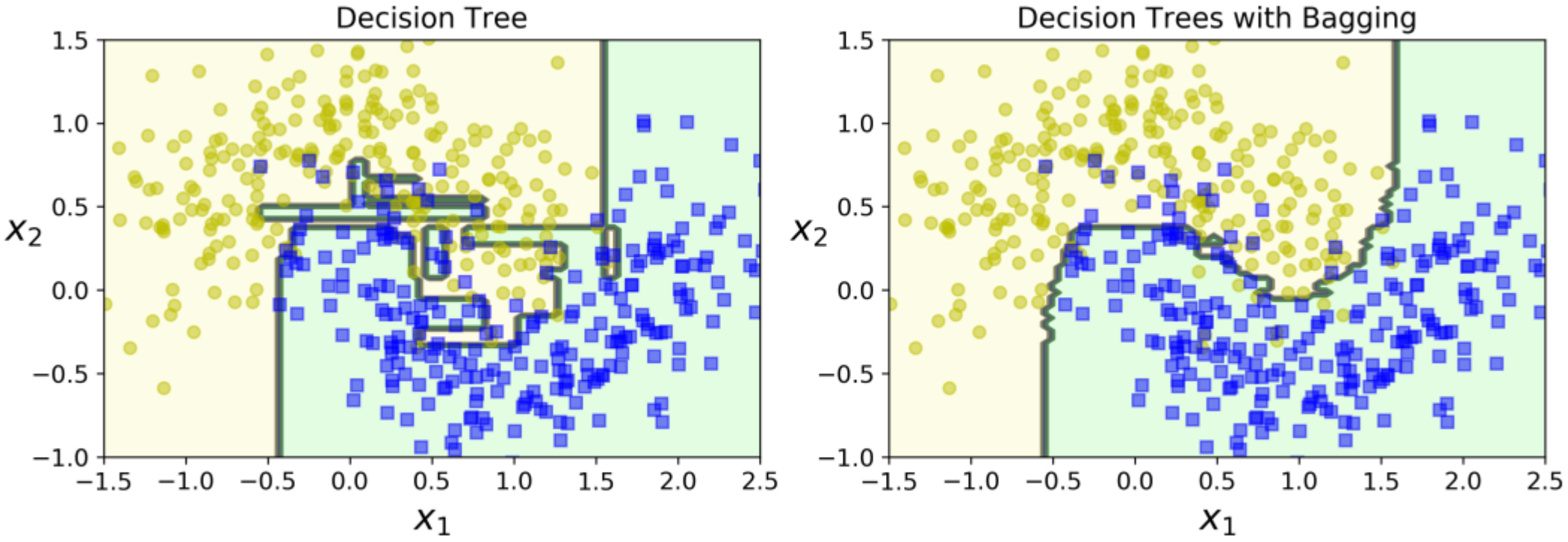
# Bagging & Pasting: Implementation

- Scikit-Learn offers a simple API for both bagging and pasting with the BaggingClassifier class (or BaggingRegressor for regression)
- The code on next slide trains an ensemble of 500 Decision Tree classifiers
- Each trained on 100 training instances randomly sampled from the training set with replacement (this is an example of bagging, for pasting, set bootstrap=False)
- The n_jobs parameter tells Scikit-Learn the number of CPU cores to use for training and predictions (–1 tells Scikit-Learn to use all available cores)

# Bagging & Pasting: Implementation

```
from sklearn.ensemble import BaggingClassifier
from sklearn.tree import DecisionTreeClassifier

bag_clf = BaggingClassifier(DecisionTreeClassifier(), n_estimators=500,max_samples=100,
                            bootstrap=True, n_jobs=-1)
bag_clf.fit(X_train, y_train)
y_pred = bag_clf.predict(X_test)
```



# Bagging vs Pasting

- Bootstrapping introduces more diversity in the subsets that each predictor is trained on, so bagging ends up with a slightly higher bias than pasting
- This means that predictors end up being less correlated so the ensemble's variance is reduced
- Overall, bagging often results in better models, which explains why it is generally preferred
- However, we can use cross validation to evaluate both bagging and pasting and select the one that works best

# Out-of-bag scoring

- If we are using bagging, there's a chance that a sample would never be selected, while others may be selected multiple time
- The probability of not selecting a specific row is (n–1)/n, where n is the number of samples
  - We are selecting one row out of n samples
- Therefore, the probability of not picking n rows in n draws is (1–1/n)^n
- When the value of n is big, we can approximate this probability to 1/e, which is approximately 0.3678. This means that when the dataset is big enough, 37% of its samples are never selected and we could use it to test our model
- This is called Out-of-Bag scoring, or OOB Scoring

# Random Forest

- Random Forest is an ensemble of Decision Trees, generally trained via the bagging method (or sometimes pasting), typically with max_samples set to the size of the training set
- Instead of building a BaggingClassifier and passing it a DecisionTreeClassifier, we use the RandomForestClassifier class, which is more convenient and optimized for Decision Trees
- Similarly, there is a RandomForestRegressor class for regression tasks
- The following code trains a Random Forest classifier with 500 trees

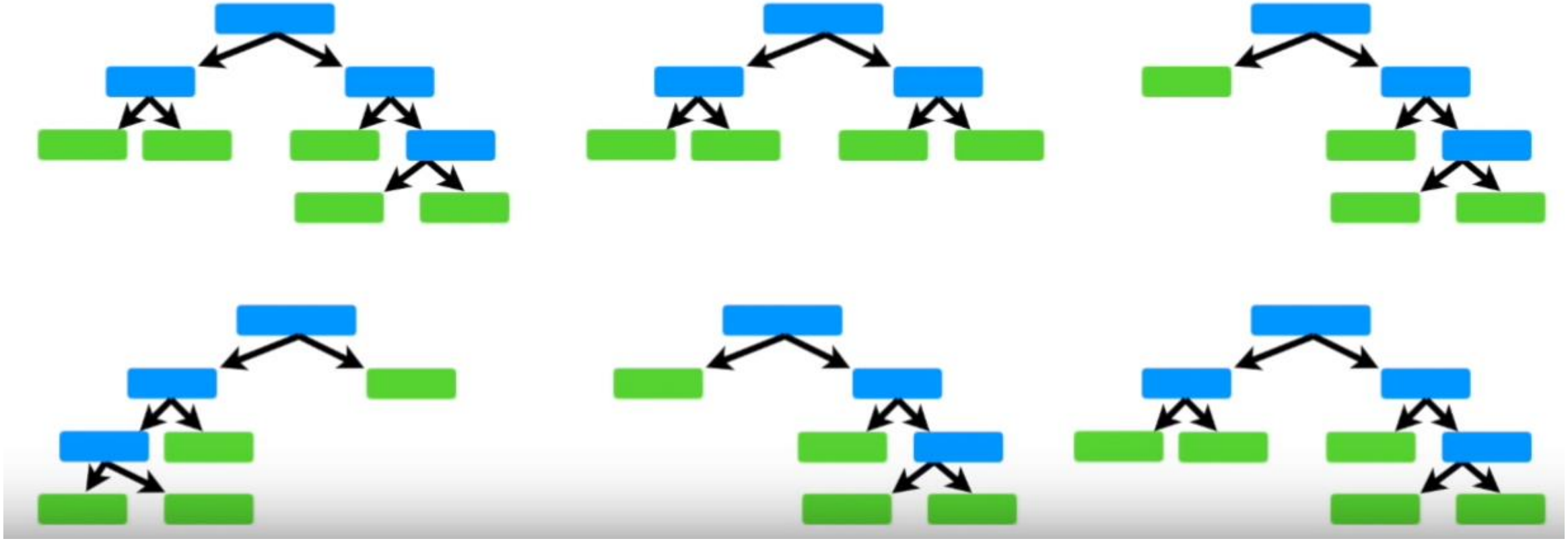
# Random Forest Implementation

```
from sklearn.ensemble import RandomForestClassifier
rnd_clf = RandomForestClassifier(n_estimators=500, max_leaf_nodes=16, n_jobs=-1)
rnd_clf.fit(X_train, y_train)
y_pred_rf = rnd_clf.predict(X_test)
```

- With a few exceptions, a RandomForestClassifier has all the hyperparameters of a DecisionTreeClassifier
  - To control how trees are grown
  - And, all the hyperparameters of a BaggingClassifier to control the ensemble

- The Random Forest algorithm introduces extra randomness when growing trees
- Instead of searching for the very best feature when splitting a node, it searches for the best feature among a random subset of features
- This results in a greater tree diversity, which (once again) without trading a higher bias for a lower variance, generally yielding an overall better model

# Random Forest: Building the model



# Random Forest: Building the model
# Create a bootstrap 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…

# Random Forest: Building the model

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.

# Random Forest: Building the model

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 |
|---|---|---|---|---|

This is the first sample that we randomly select…

# Random Forest: Building the model

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 |

This is the second randomly selected sample from the original dataset…

# Random Forest: Building the model

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 |

…so here it is in the bootstrapped

# Random Forest: Building the model

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 |

# Random Forest: Building the model

**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 |

# Random Forest: Building the model

**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 |

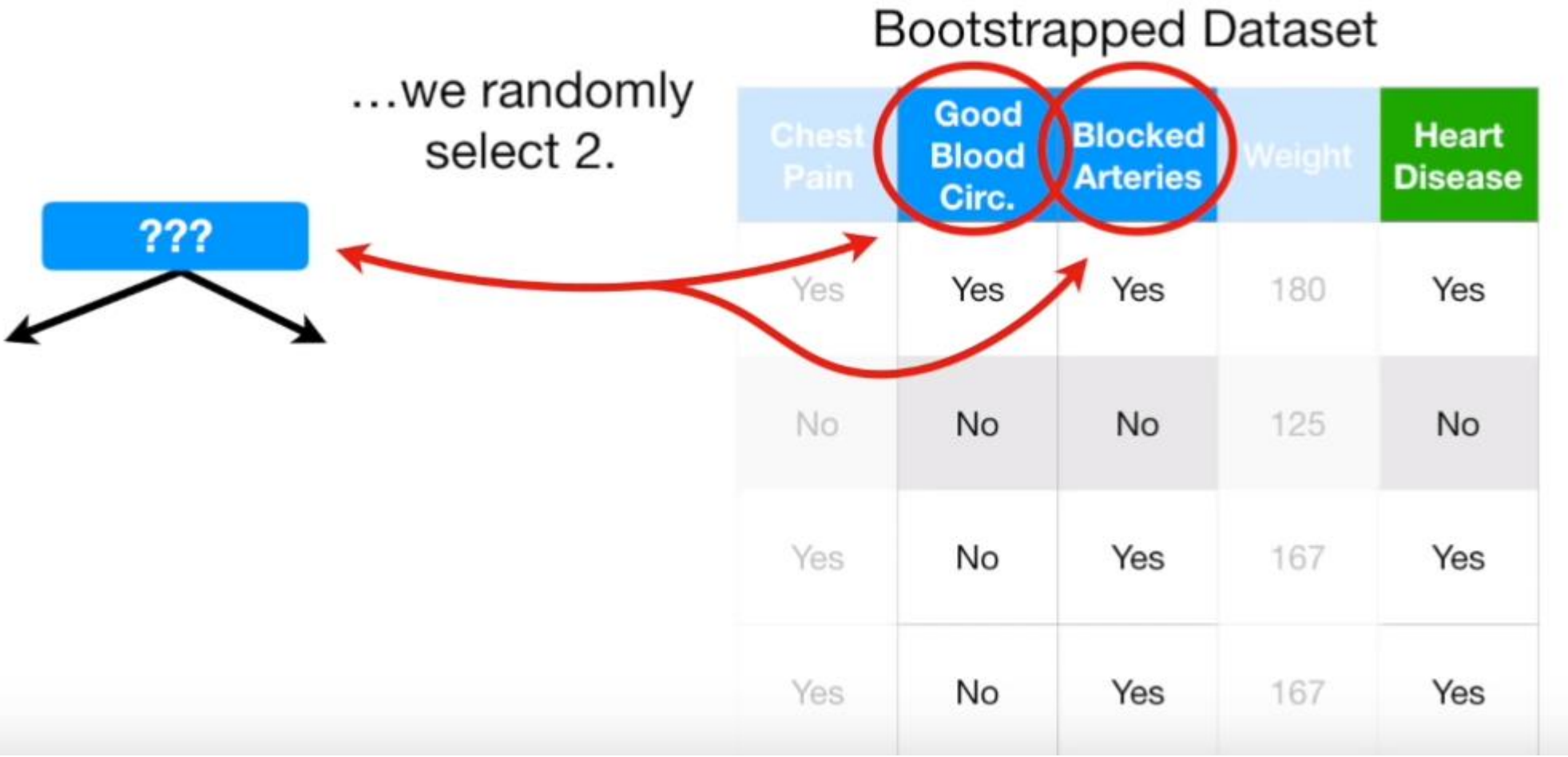
# Random Forest: Building the model

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 |

# Random Forest: Building the model



# Random Forest: Building the model


Bootstrapped Dataset

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

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

# Random Forest: Building the model

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 |

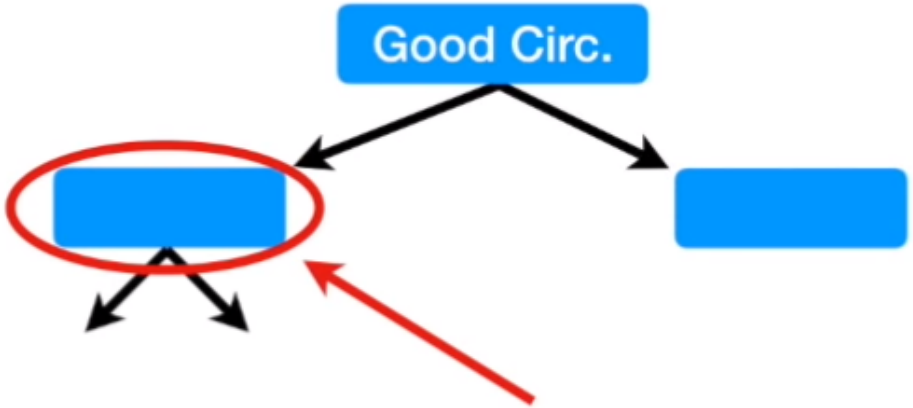
# Random Forest: Building the model

Since we used **Good Blood Circulation**, I'm going to grey it out so that we focus on the remaining variables.


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 |

# Random Forest: Building the model

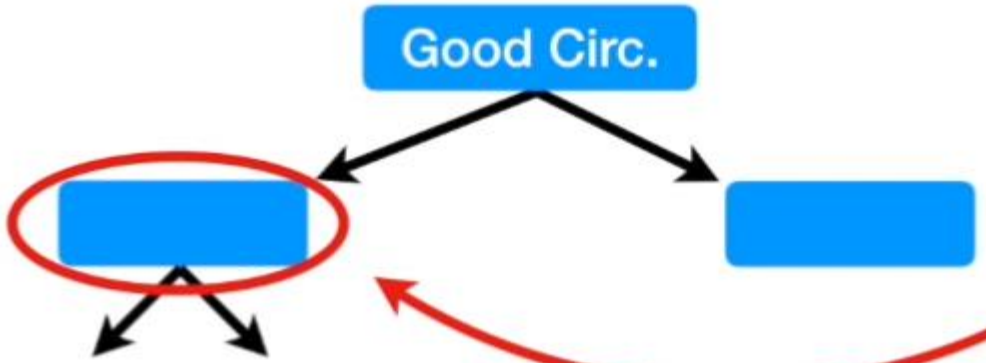



Now we need to figure out how to split samples at this node.

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 |

# Random Forest: Building the model

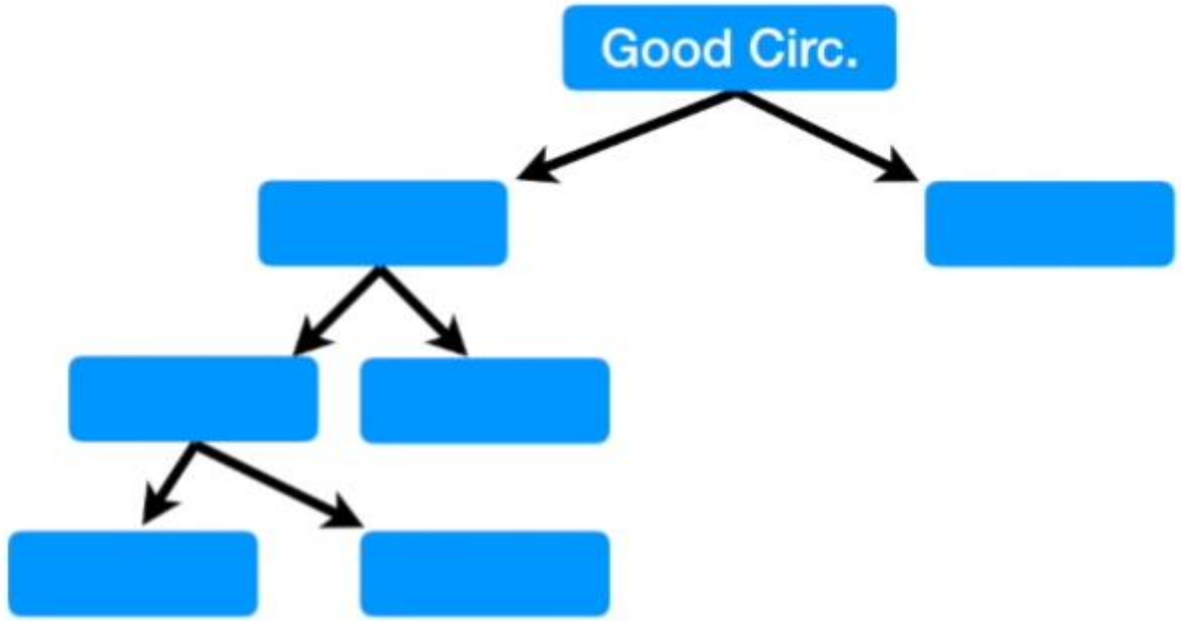



Bootstrapped Dataset

| 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 |

Just like for the root, we randomly select 2 variables as candidates, instead of all 3 remaining columns.

# Random Forest: Building the model




Bootstrapped Dataset

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

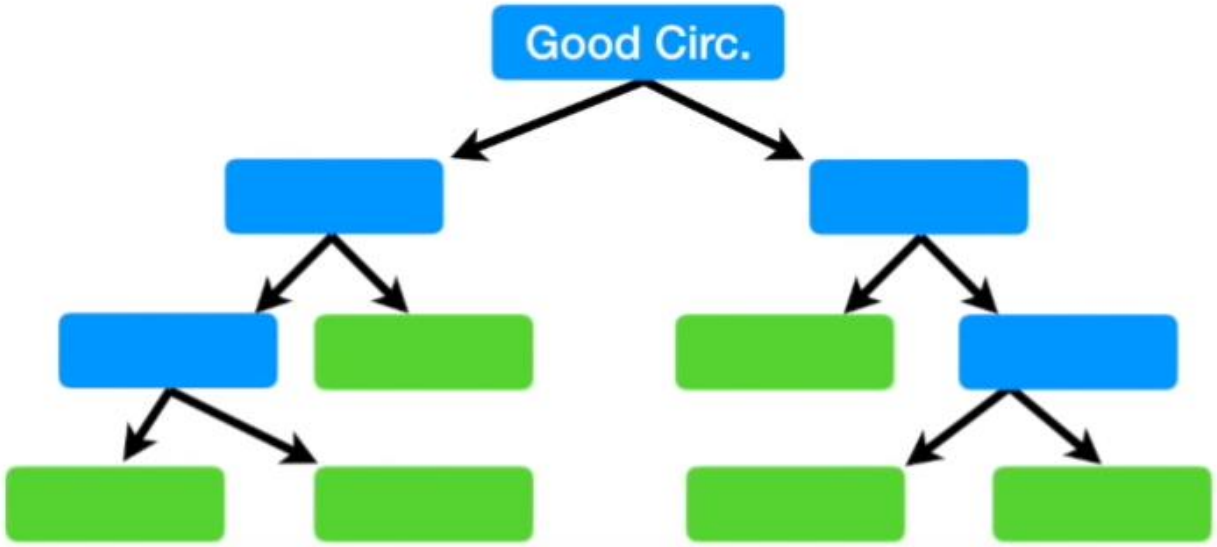
And we just build the tree as usual, but only considering a random subset of variables at each step.

# Random Forest: Building the model

We built a tree…

1) Using a bootstrapped dataset

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

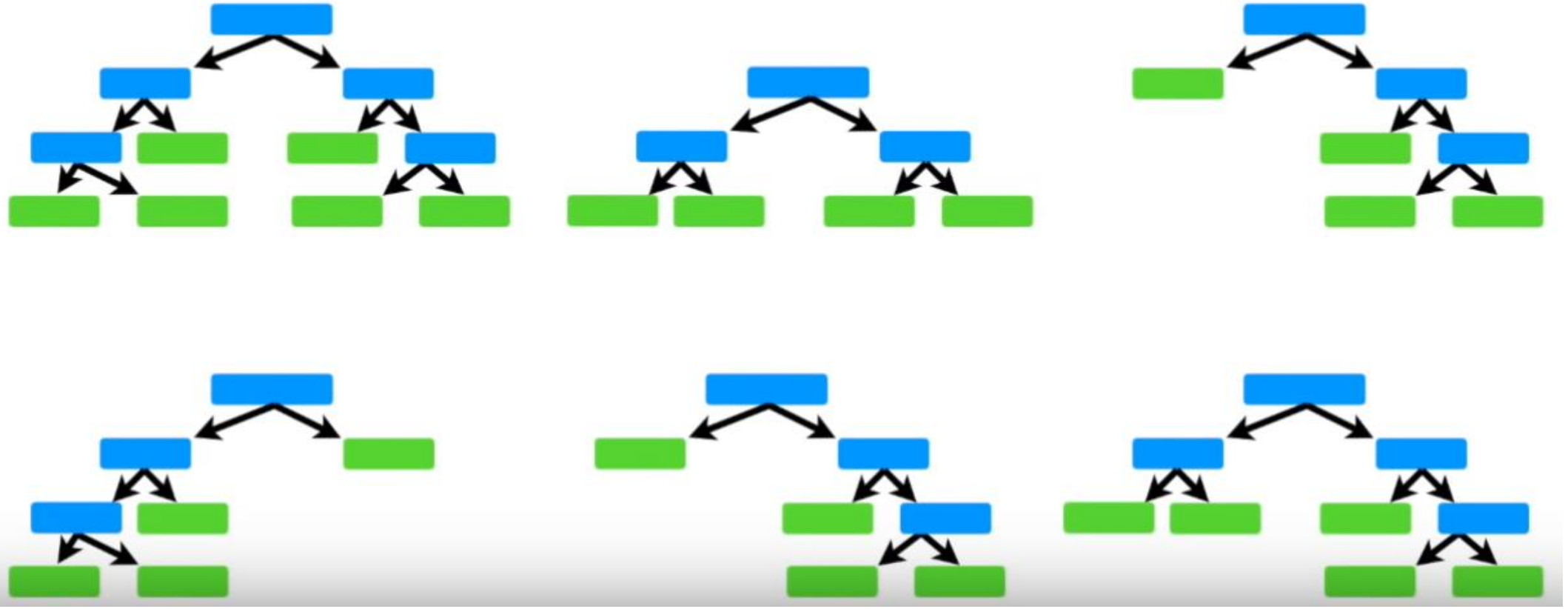



Bootstrapped Dataset

| 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 |

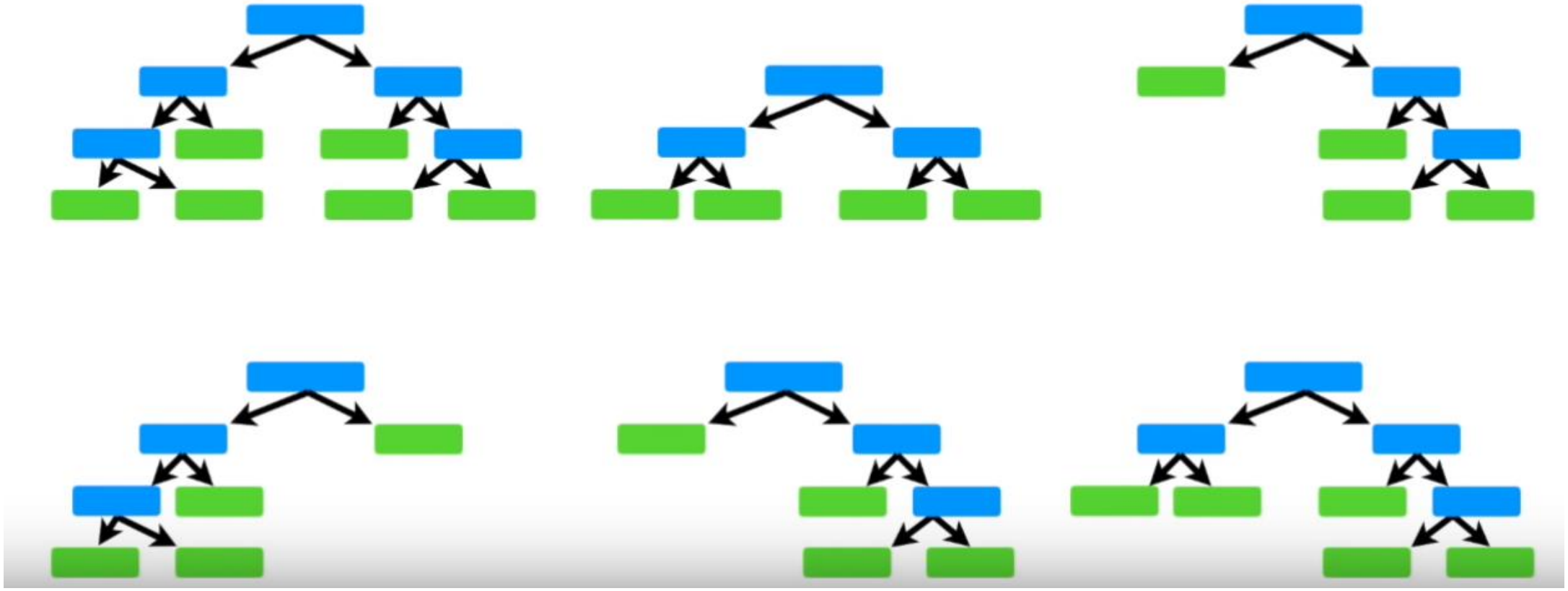
# Random Forest: Building the model

**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.



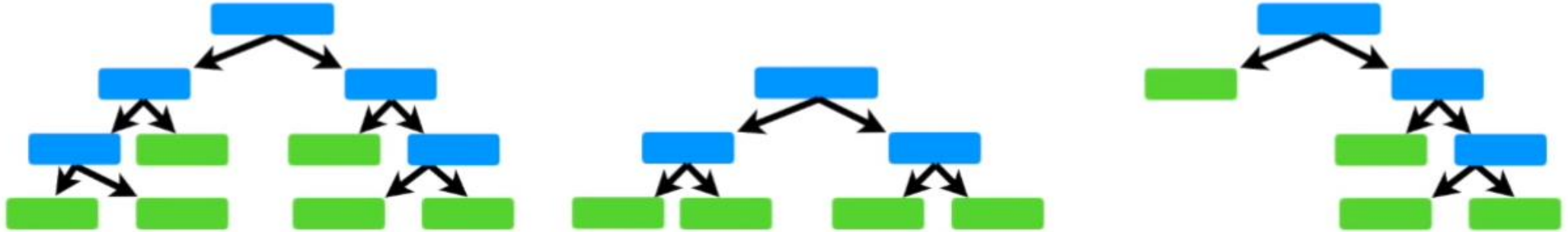
# Random Forest: Building the model

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

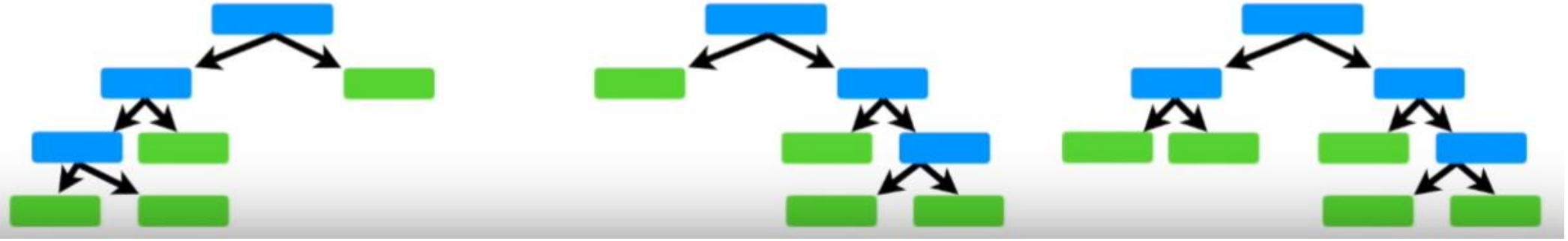


# Random Forest: Building the model

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.



# Random Forest: Prediction

Well, first we get a new patient…

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

# Random Forest: Prediction

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

…we've got all the measurements…

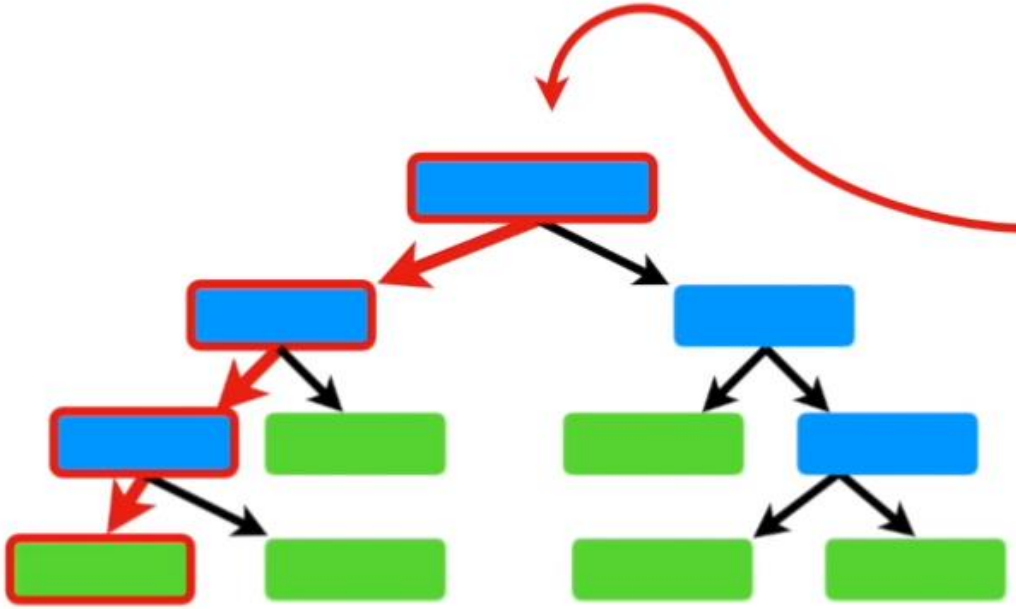
# Random Forest: Prediction

| 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.

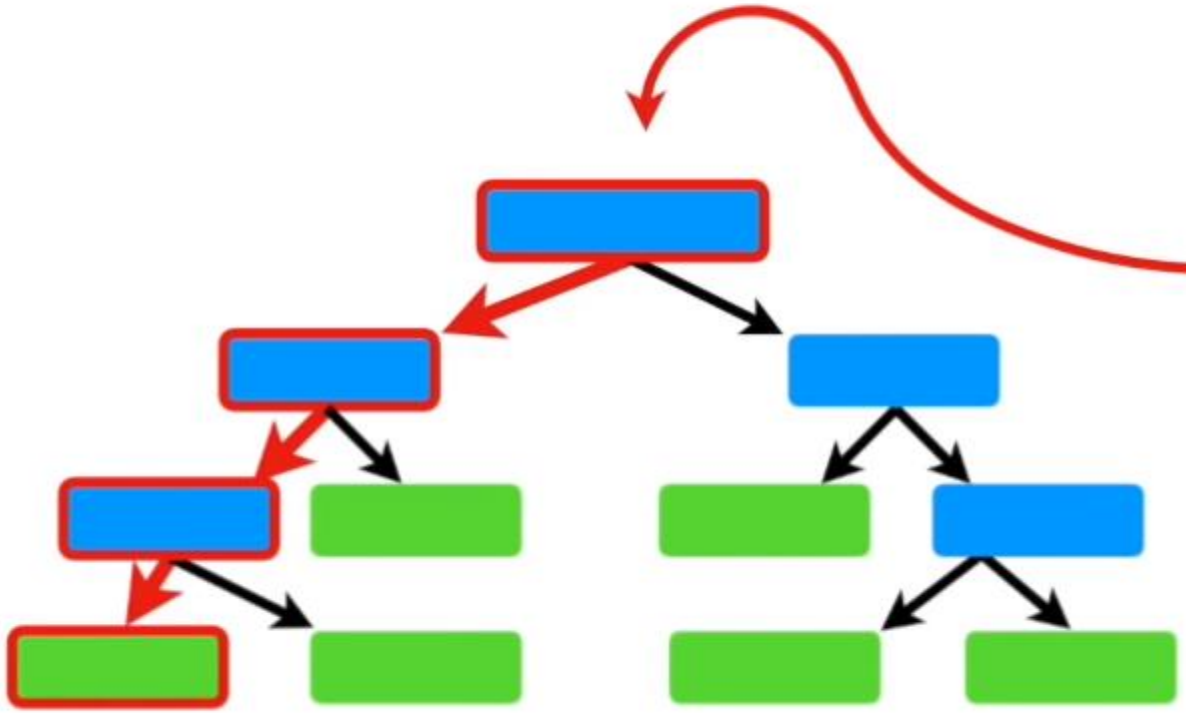
# Random Forest: Prediction

So we take the data and run it down the first tree that we made…



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

# Random Forest: Prediction



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

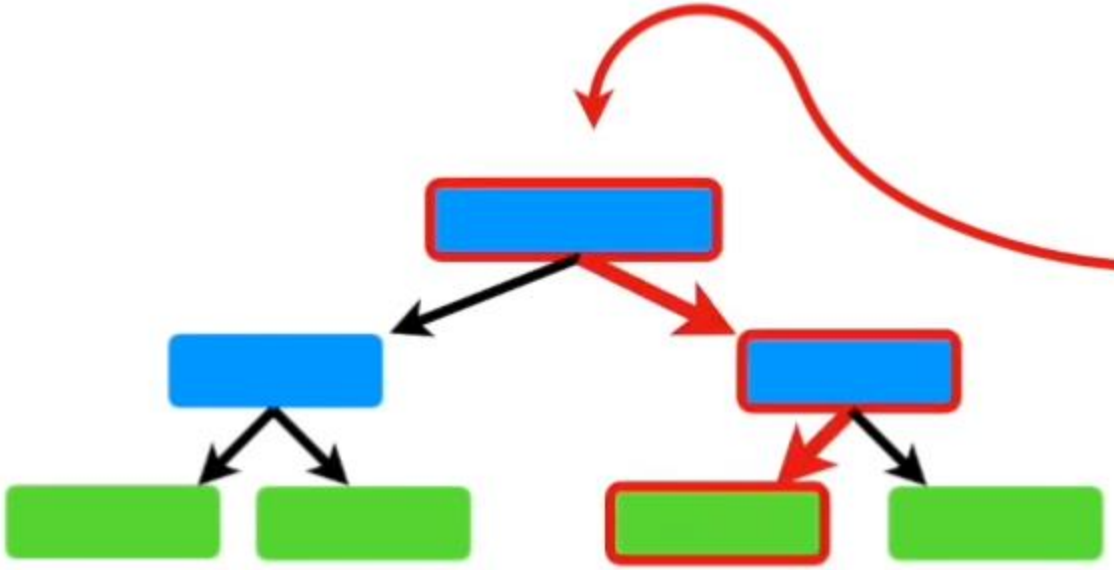
| Heart Disease | |
|---|---|
| Yes | No |
| 1 | 0 |

The first tree says "Yes"…

…and we keep track of that here.

# Random Forest: Prediction

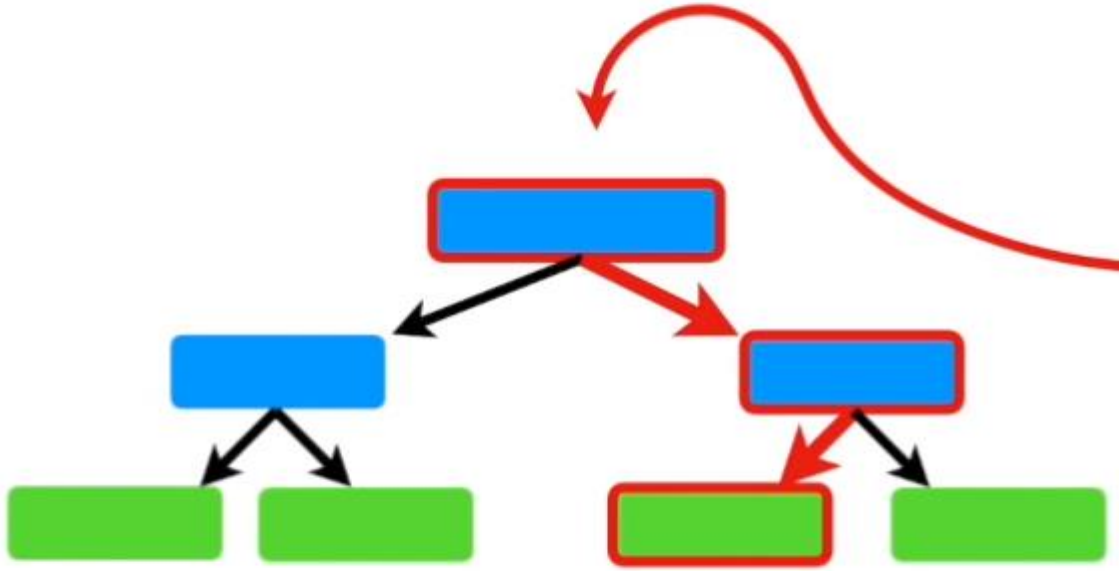
Now we run the data down the second tree that we made…



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

| Heart Disease | |
|---|---|
| Yes | No |
| 1 | 0 |

# Random Forest: Prediction



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

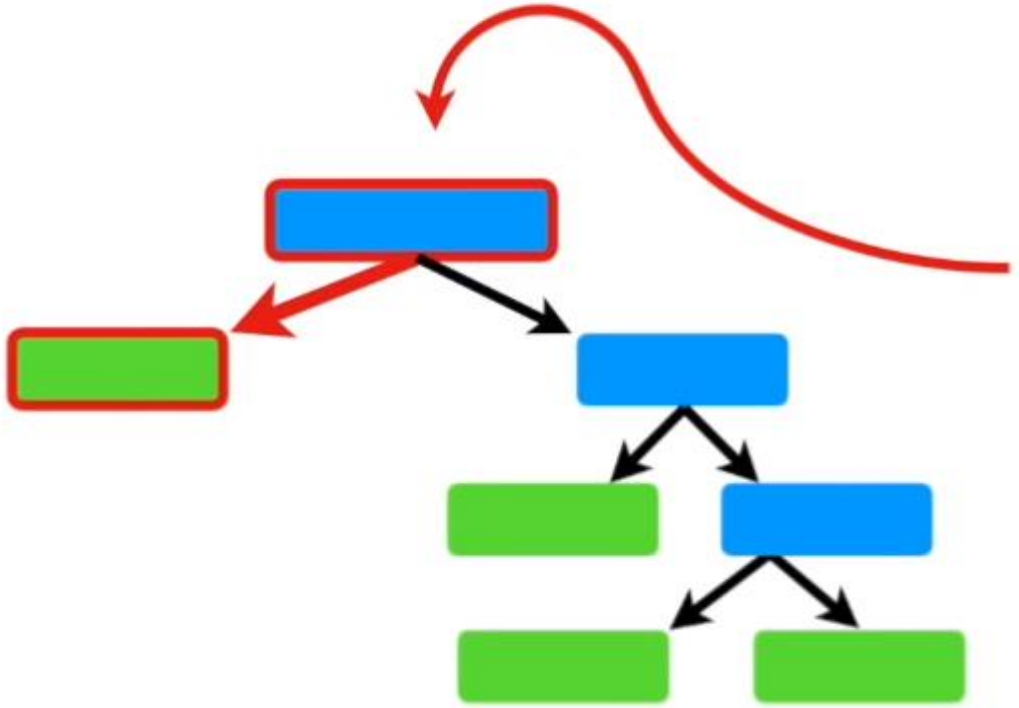
The second tree also says “Yes”…

| Heart Disease | |
|---|---|
| Yes | No |
| 2 | 0 |

…and we keep track of that here.

# Random Forest: Prediction

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 |
| 3 | 0 |

# Random Forest: Prediction

| 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 |

# Random Forest: Prediction

| 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.

# Random Forest: Prediction

**Bootstrapping the data plus using the aggregate 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 |

# Random Forest: Evaluation

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 |

# Random Forest: Evaluation

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 |

# Random Forest: Evaluation

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 |

# Random Forest: Evaluation

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 |

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

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

# Random Forest: Evaluation

Original Dataset

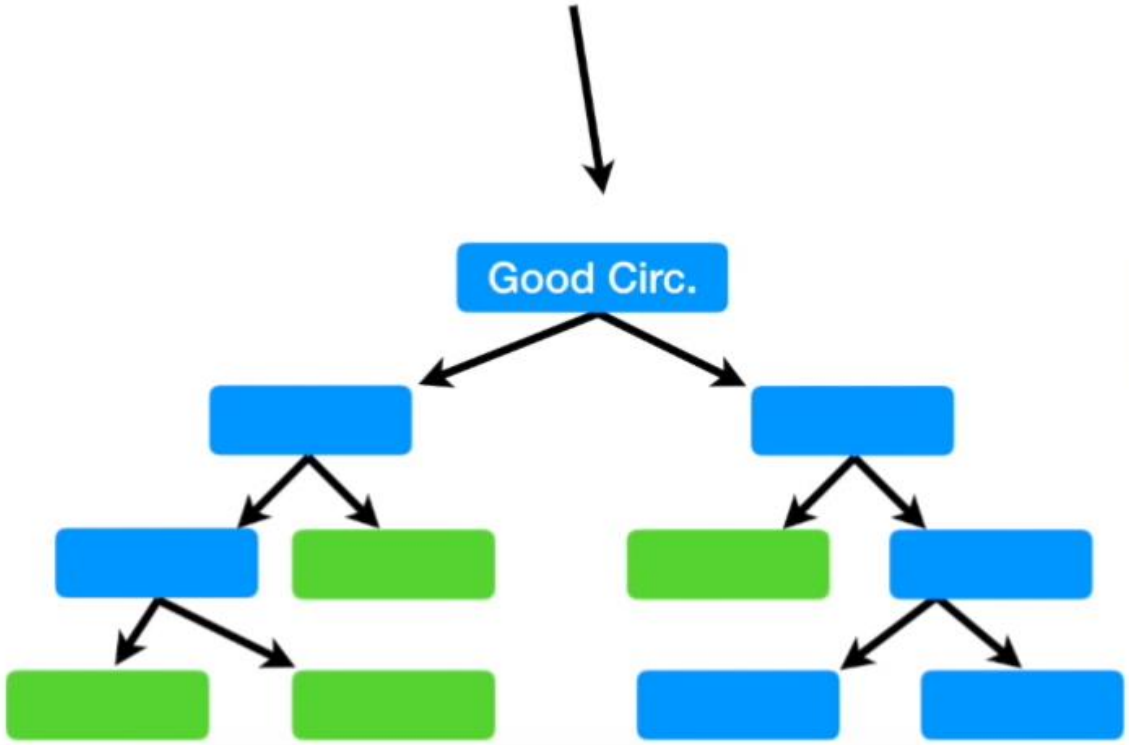
| Chest Pain | Good Blood Circ. | Blocked Arteries | Weight | Heart Disease |
|---|---|---|---|---|
| No | No | No | 125 | No |
| Yes | Yes | Yes | 180 | Yes |
| Yes | Yes | No | 210 | No |
| Yes | No | Yes | 167 | Yes |

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

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

# Random Forest: Evaluation

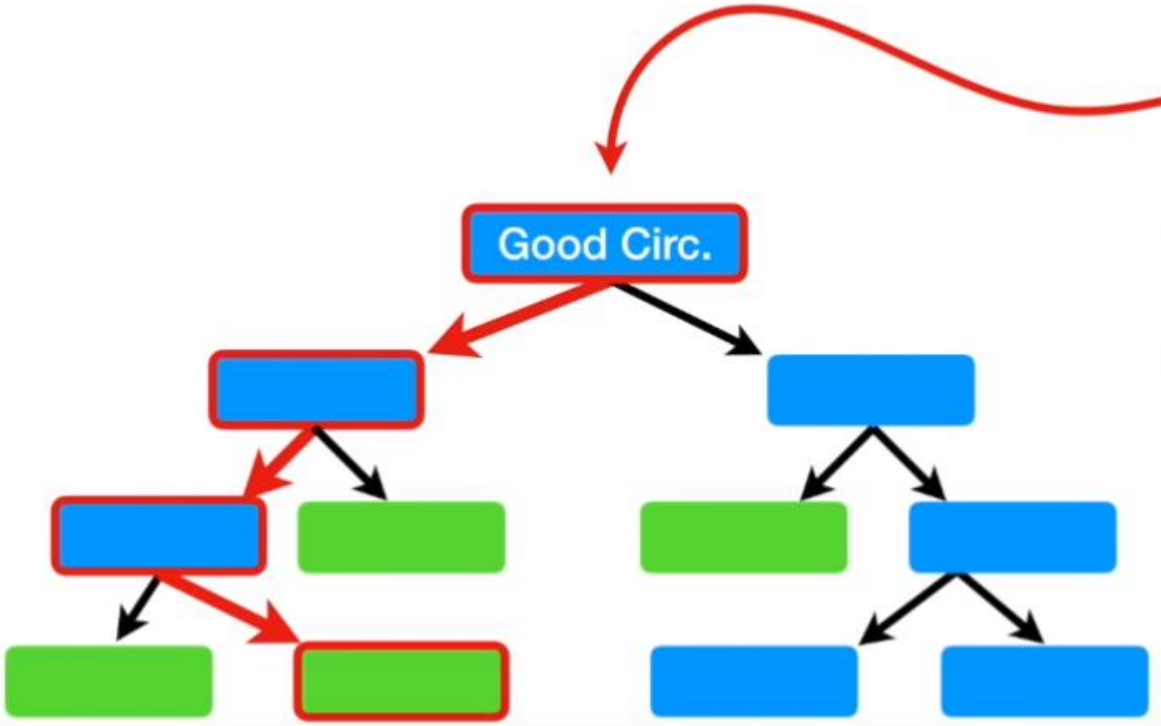
Since the Out-Of-Bag data was not used to create this tree…




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

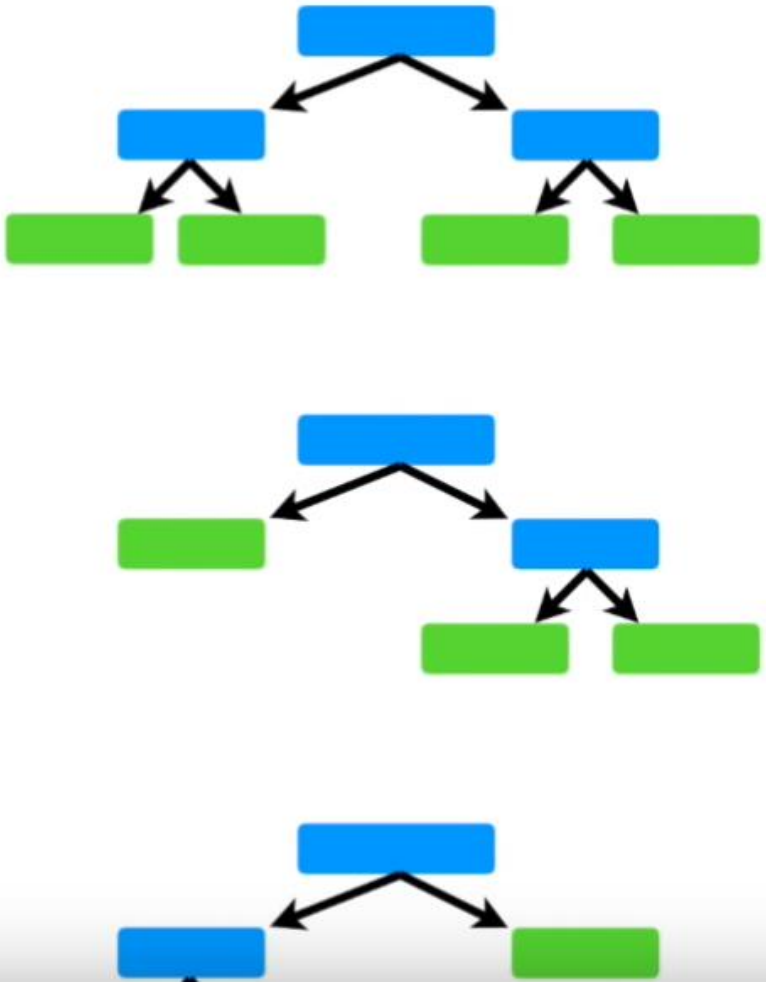
# Random Forest: Evaluation

…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 |

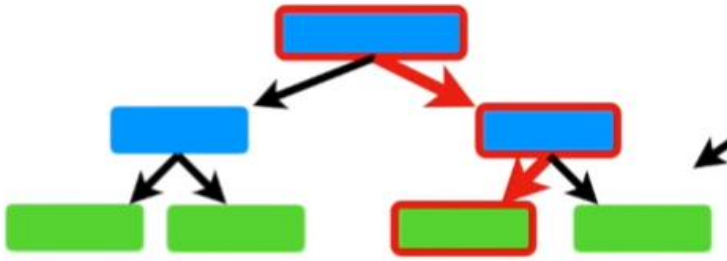
# Random Forest: Evaluation



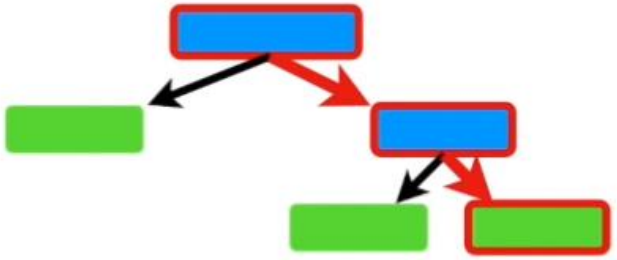
Then we run this Out-Of-Bag sample through all of the other trees that were built without it…

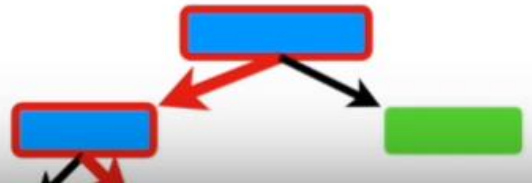
| Chest Pain | Good Blood Circ. | Blocked Arteries | Weight | Heart Disease |
|---|---|---|---|---|
| Yes | Yes | No | 210 | No |

# Random Forest: Evaluation



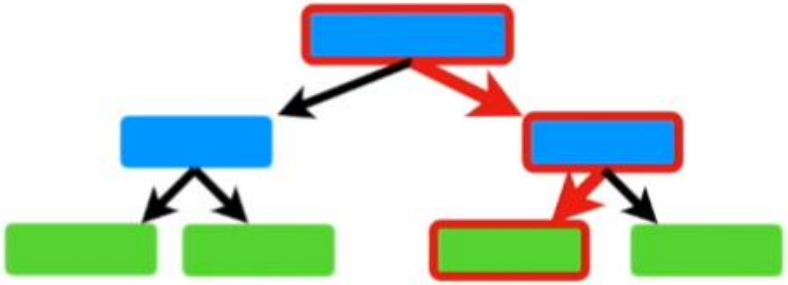
This tree incorrectly labeled the Out-of-Bag sample "**Yes**".

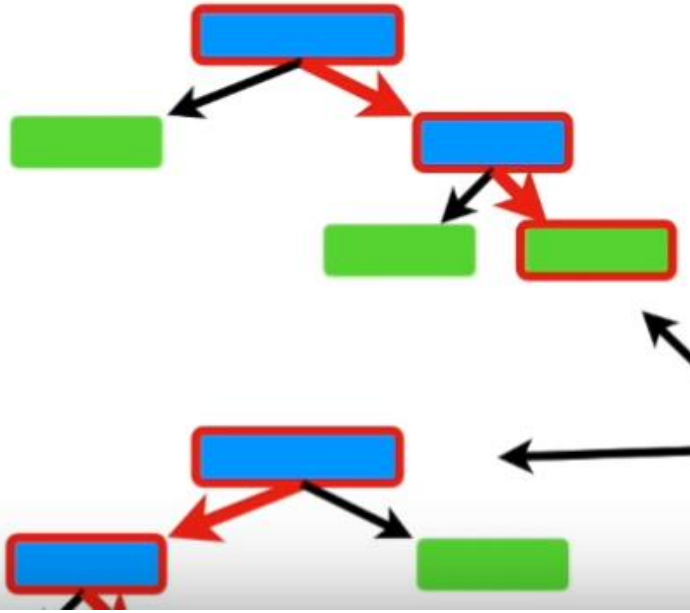




| Chest Pain | Good Blood Circ. | Blocked Arteries | Weight | Heart Disease |
|---|---|---|---|---|
| Yes | Yes | No | 210 | No |

# Random Forest: Evaluation





| 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**".

# Random Forest: Evaluation

| 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 |

# Random Forest: Evaluation

| 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 |

# Random Forest: Evaluation

| 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.

# Random Forest: Evaluation

| 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…

# Random Forest: Building the model

| 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 |

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.

# Random Forest: Building the model

| 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 |

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**"

# Random Forest: Building the model

OK, we now know how to:

1) Build a Random Forest

2) Use a Random Forest

3) Estimate the accuracy of a Random Forest.

# Random Forest: Building the model

OK, we now know how to:

1) Build a Random Forest

2) Use a Random Forest

3) Estimate the accuracy of a Random Forest.

However, now that we know how to do this…

…we can talk a little more about how to do this!

# Random Forest: Building the model

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 |

# Random Forest: Building the model

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 |

# Random Forest: Building the model

…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 |

# Random Forest: Building the model

…to random forest built using 3 variables per step…

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

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 |

# Random Forest: Building the model

In other words…

1) Build a Random Forest

2) Estimate the accuracy of a Random Forest.

…change the number of variables used per step…

# Random Forest: Building the model

In other words…

1) Build a Random Forest

2) Estimate the accuracy of a Random Forest.

…change the number of variables used per step…

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

# Next steps….

- Handle missing data… next class
- Feature selection using random forests
- More ensemble methods…