# Random forests and selecting architectures

4 minutes

Experimentation with architectures is often a key focus of building effective modern models. We've done so to a basic level with decision trees, but the only limit to this is our imagination – and perhaps our computer's memory. In fact, thinking more broadly on decision trees resulted in a highly popular model architecture that reduces its decision trees' tendency to overfit data.

## What's a random forest?

A random forest is a collection of decision trees, which are used together to estimate which label a sample should be assigned. For example, if we were to train a random forest to predict medal winners, we might train 100 different decision trees. To make a prediction, we would use all trees independently. These would effectively 'vote' for whether the athlete would win a medal, providing a final decision.

### How is a random forest trained?

Random forests are built on the idea that while a single decision tree is highly biased, or overfit, if we train several decision trees, they'll be biased in different ways. This requires that each tree is trained independently, and each on a slightly different training set.

To train a single decision tree a certain number of samples, athletes in our scenario, are extracted from the full training set. Each sample can be selected more than once, and this takes place randomly. The tree is then trained in the standard way. This process is repeated for each tree. As each tree gets a different combination of training examples, each tree ends up trained, and biased, differently to the others.

### Advantages of random forest

The performance of random forests is often impressive and so comparisons are often best made against neural networks, which are another popular and high-performance model type. Unlike neural networks, random forest models are easy to train: modern frameworks provide helpful methods that let you do so in only a few lines of code. Random forests are also fast to train and don't need large datasets to perform well. This separates them from neural networks, which can often take minutes or days to train, substantial experience, and often require very large datasets. The architectural decisions for random forests are, while more complex than models such as linear regression, much simpler than neural networks.

### Disadvantages of random forest

The major disadvantage of random forests is that they're difficult to understand. Specifically, while these models are fully transparent – each tree can be inspected and understood – they often contain so many trees that doing so is virtually impossible.

## How can I customize these architectures?

Like several models, random forests have various architectural options. The easiest to consider is the size of the forest – how many trees are involved, along with the size of these trees. For example, it would be possible to request a forest to predict medal winners containing 100 trees, each with a maximum depth of six nodes. This means that the final decision as to whether an athlete will win a medal must be made with no more than six 'if' statements.

As we've already seen, increasing the size of a tree (in terms of depth or number of leaves) makes it more likely to overfit the data it's trained on. This limitation also applies to random forests. However, with random forests we can counter this by increasing the number of trees, assuming that each tree will be biased in a different way. We can also restrict each tree to only a certain number of features, or disallowing leaves to be created when it would make only a marginal difference to the training performance. The ability for a random forest to make good predictions isn't infinite. At some point, increasing the size and number of trees gives no further improvement due to the limited variety of training data that we've.

---

### Next unit: Exercise - Selecting random forest architectures

Continue >