HST.953x Workshop 2.09: Decision Tree Exercises

H. David Shea

15 Jul 2021

Contents

(Note: Updated and modified from "A Language, not a Letter: Learning Statistics in R".)

Description

Decision trees are made up to two parts: nodes and leaves.

- **Nodes**: represent a decision test, examine a single variable and move to another node based on the outcome
- **Leaves**: represent the outcome of the decision.

Decision trees are useful to make various predictions. For example, to predict if an email is SPAM or not, to predict health outcomes, to predict what group an individual belongs to based on a variety of factors that are specified in the decision tree model.

Dataset

We will be using Edgar Anderson's Iris Data in the `iris` dataset for these examples. The dataset is a data frame with 150 cases (rows) and 5 variables (columns) named `Sepal.Length`, `Sepal.Width`, `Petal.Length`, `Petal.Width`, and `Species`.

```
base_dir <- here::here("")

#iris$class <- as.factor(iris$class)

summary(iris)
#>   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
#>  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
```

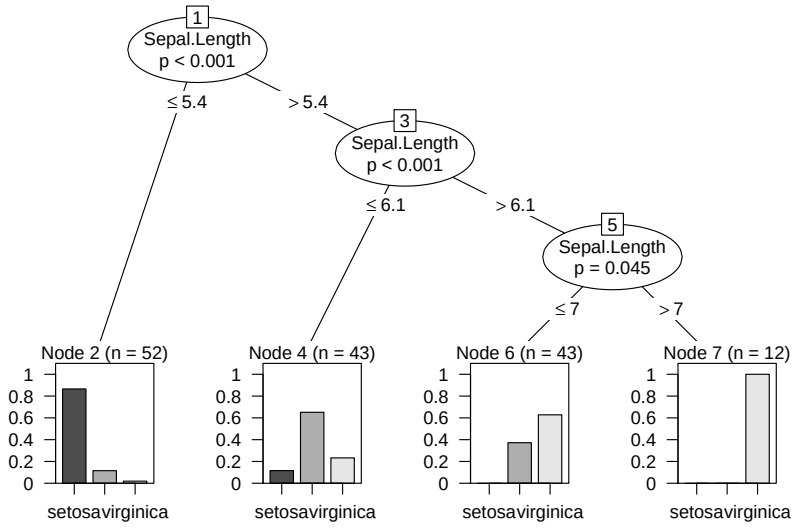
```
#>  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
#>  Median :5.800   Median :3.000   Median :4.350   Median :1.300
#>  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
#>  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
#>  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
#>        Species
#>  setosa    :50
#>  versicolor:50
#>  virginica :50
#>
#>
#>
```

Single Variable Decision Trees

In this section we will use each dimension of the flower separately to create individual decision tress to try to predict the `Species`.

To see how well `Sepal.Length` predicts which `Species` of iris a flower is, we create the following decision tree.

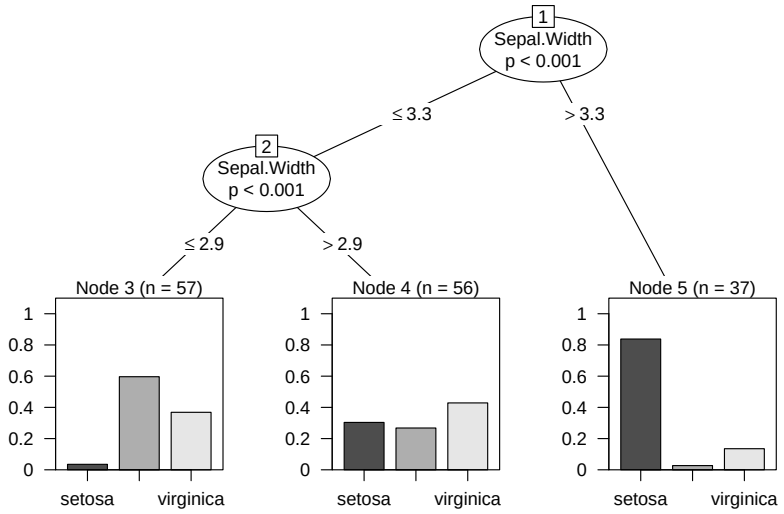
```
tree1 <- ctree(Species ~ Sepal.Length, data = iris)

plot(tree1)
```



Overall, the decision tree tells us that sertosa iris flowers tend to have shorter `Sepal.Length`, versicolor iris flowers have middle range `Sepal.Length`, and virginica iris flowers tend to have the longest `Sepal.Length`.

To see how well `Sepal.Width` predicts which `Species` of iris a flower is, we create the following decision tree.

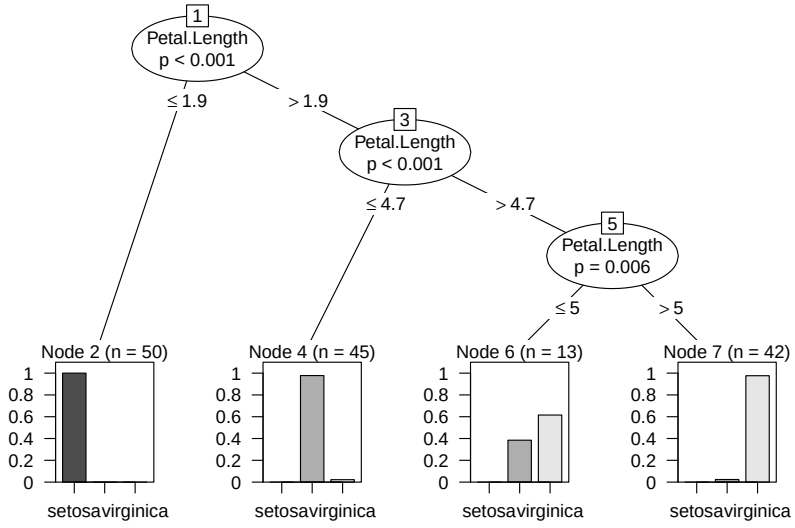
```
tree2 <- ctree(Species ~ Sepal.Width, data = iris)

plot(tree2)
```



The results are much more mixed on `Sepal.Width`. Main conclusions would be that setosa iris tend to have wider `Sepal.Width`, versicolor tend to have more narrow `Sepal.Width`, and virginica have more variety in sepal`Sepal.Width`.

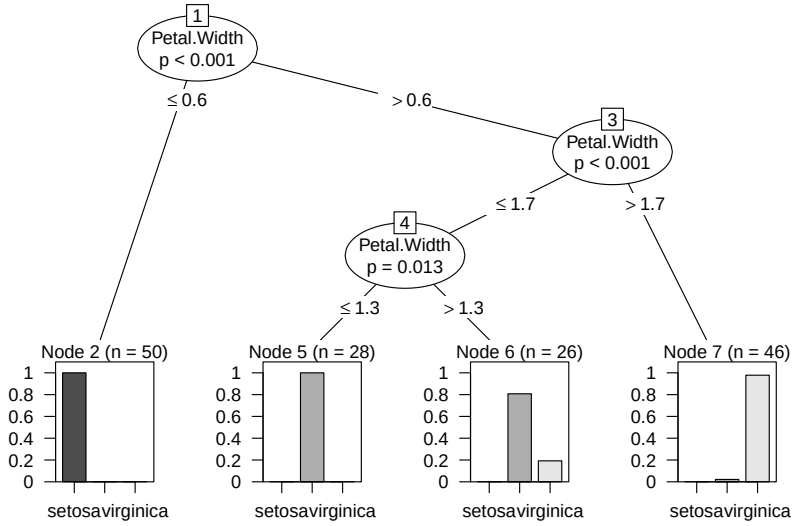
To see how well `Petal.Length` predicts which `Species` of iris a flower is, we create the following decision tree.

```
tree3 <- ctree(Species ~ Petal.Length, data = iris)

plot(tree3)
```



To see how well `Petal.Length` predicts which `Species` of iris a flower is, we create the following decision tree.

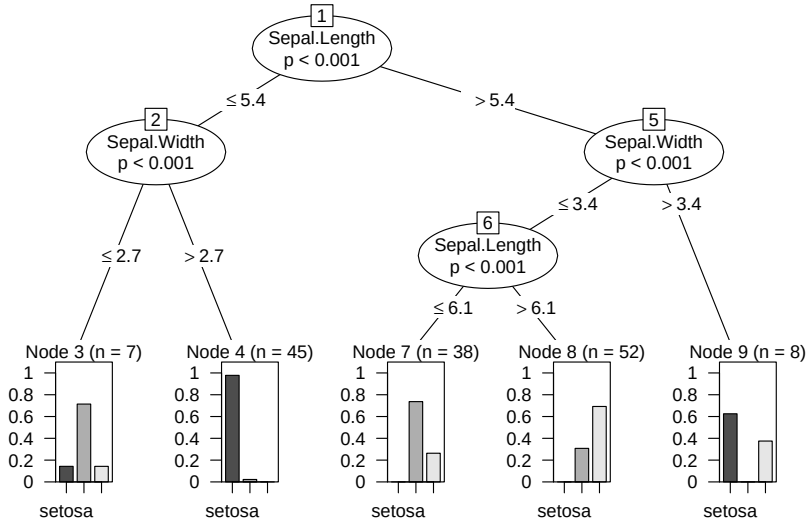
```
tree4 <- ctree(Species ~ Petal.Width, data = iris)

plot(tree4)
```



Two Variable Decision Trees

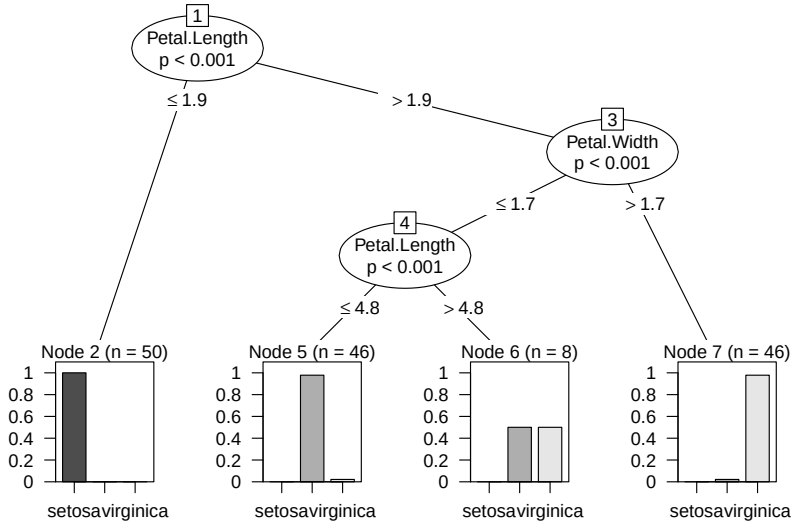
`Sepal.Length` + `Sepal.Width`:

```
tree5 <- ctree(Species ~ Sepal.Length + Sepal.Width, data = iris)

plot(tree5)
```



`Petal.Length` + `Petal.Width`:

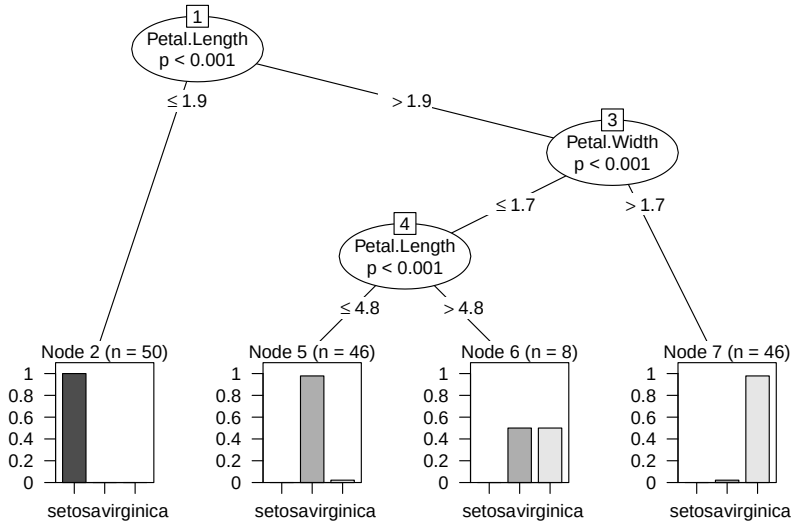
```
tree6 <- ctree(Species ~ Petal.Length + Petal.Width, data = iris)

plot(tree6)
```



A Decision Trees with All Predictors

`Sepal.Length` + `Sepal.Width` + `Petal.Length` + `Petal.Width`:

```
tree7 <- ctree(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width, data = iris)

plot(tree7)
```



Notice that the decision tree is identical to decision tree using `Petal.Length` + `Petal.Width` and that those are the only two factors that are used in the decision nodes. This tells us that these two factors are most important when distinguishing to which type of iris class each flower belongs. The factors `Sepal.Length` and `Sepal.Width` are not necessary to predict to which class the flowers belong.

Avoid Overfitting Decision Trees

There are two approaches to avoid overfitting a decision tree to your data.

1. **pre-pruning**: prevents the tree from growing earlier, before the training data is perfectly classified
2. **post-trimming**: or post-pruning, tree is perfectly classified then after the tree is created prune or trim the tree

Post-trimming is the most common approach because it's often difficult to estimate when to stop growing the tree. The important thing is to define the criteria which determines the correct final tree size.

1. **validation set**: use a different data set, other than the training set, to evaluate the post-trimming nodes from the decision tree. Often the dataset is broken in to two datasets, the training set and the validation set. The decision tree is constructed on the training set, then any post-trimming is done on the validation set.
2. **statistical testing**: create the decision tree using the training set, then apply statistical tests (error estimation or chi square) to determine if pruning a node or expanding a node produces an improvement beyond the training set. For more information on these statistical tests, see the "Overfitting Data" in the references and resources section.