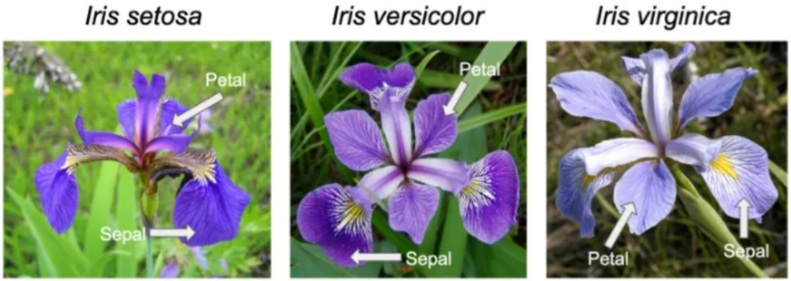
Decision Trees

Ryan Miller

Grinnell College
Statistics

Introduction

The famous statistician Ronald Fisher collected 50 samples from each of three species of Iris:



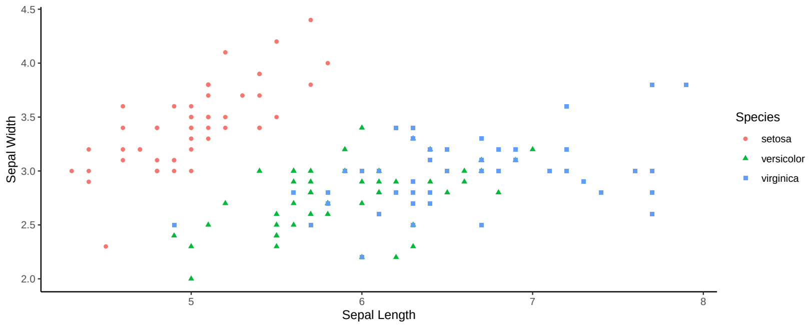
He measured the dimensions of the sepal and petal of each sample.

Image source: https://peaceadegbite1.medium.com/iris-flower-classification-60790e9718a1

Grinnell College
Statistics

Introduction

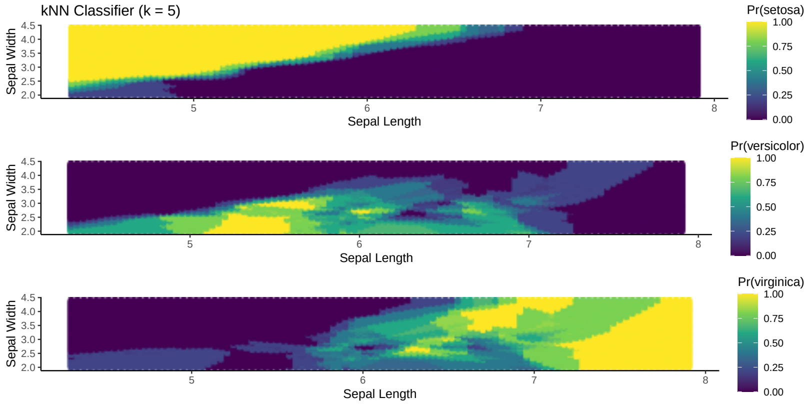
Shown below are the sepal dimensions of these flowers:



Can sepal dimensions be used to accurately classify different species of iris?

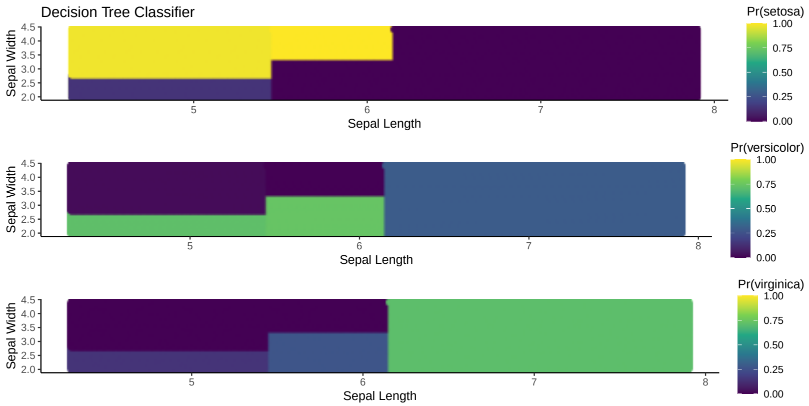
Classification with *k*-nearest neighbors

We can train a KNN classifier, but notice how the classification boundaries are highly irregular and seem somewhat haphazard:



Decision trees on the iris data

Decision trees learn a series of recursive binary splitting rules to make classification decisions:



Grinnell College Statistics

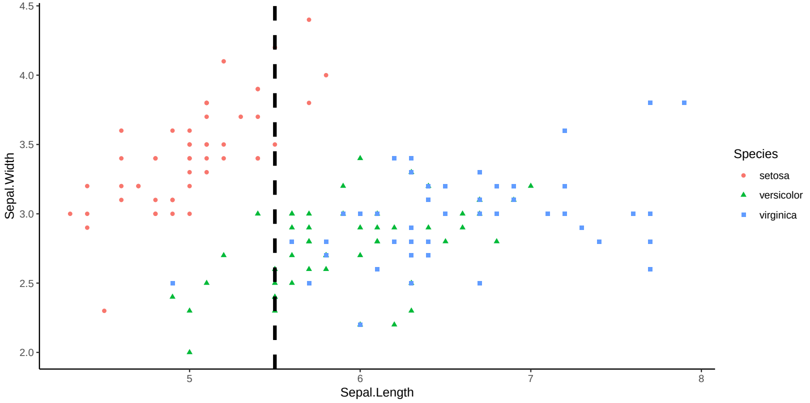
Decision trees

Decision trees are trained by recursively partitioning the p-dimensional space defined by the explanatory variables:

1) Starting with a "parent" node, search for a splitting rule that maximizes the *homogeneity* or *purity* of the "child" nodes
2) Next, considering each node that hasn't yet been split, find another splitting rule that maximizes *purity*
3) Repeat until a stopping criteria has been reached

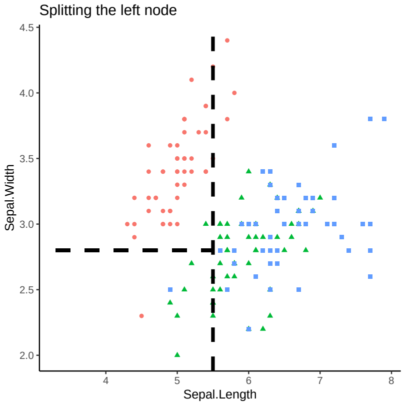
We'll soon discuss how splitting rules are determined and how the algorithm terminates, but we'll first see what this looks like for the Iris data.

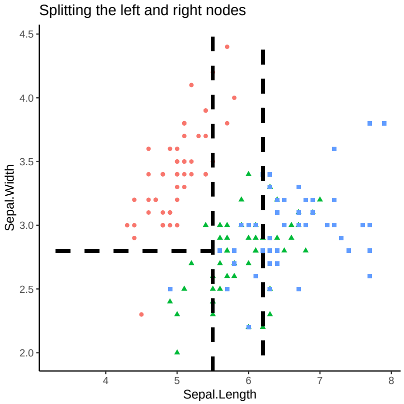
Example (first split)



Grinnell College
Statistics

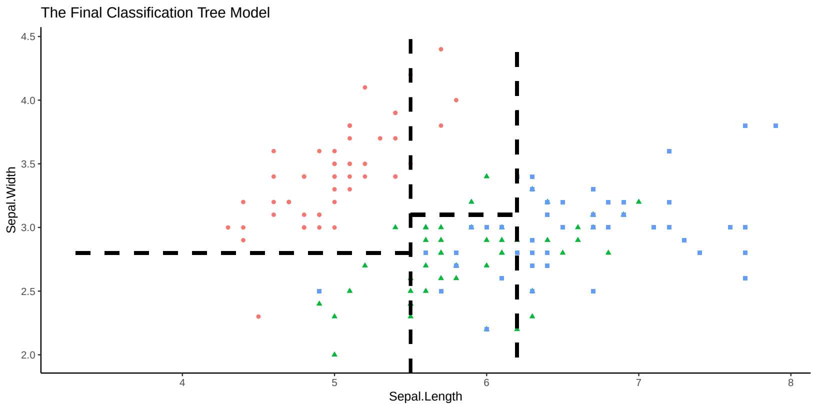
Example (second split)





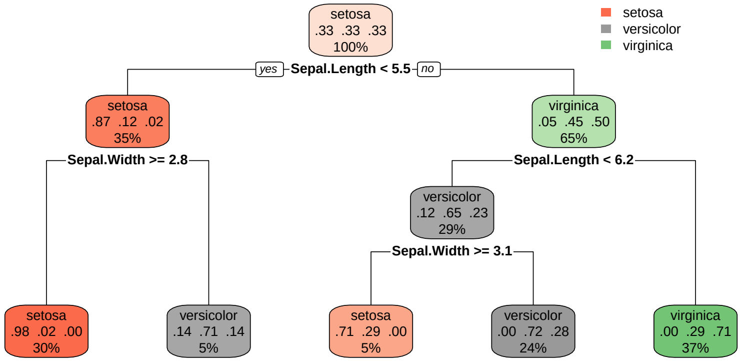
Grinnell College Statistics

Example (final model)



Grinnell College
Statistics

Example (full tree)



Grinnell College
Statistics

Splitting criteria

Decision trees must learn their splits using an objective criteria, the most common criteria is *Gini impurity*:

$$Gini = \sum_{j=1}^{k} p_j(1-p_j) = 1 - \sum_{j=1}^{k} p_j^2$$

- For binary classification, this reduces to $p_1(1-p_1)+p_2(1-p_2)$
- Other splitting criteria exist; however, Gini impurity is the default in `sklearn`

Gini Gain

- Gini impurity gives us a measure of the impurity of a tree at given depth
  - In our example tree, the Gini impurity of the starting node was $1 - (\frac{1}{3}^2 + \frac{1}{3}^2 + \frac{1}{3}^2) = \frac{2}{3}$

Grinnell College
Statistics

Gini Gain

- Gini impurity gives us a measure of the impurity of a tree at given depth
    - In our example tree, the Gini impurity of the starting node was $1 - (\frac{1}{3}^2 + \frac{1}{3}^2 + \frac{1}{3}^2) = \frac{2}{3}$
- The best split is the one that produces the largest decrease (or *Gini gain*) in the resulting child nodes
    - In our example, the Gini impurity after the first split was:

$$0.35 * [1 - (0.87^2 + 0.12^2 + 0.02^2)] + \\ 0.65 * [1 - (0.05^2 + 0.45^2 + 0.50^2)] = 0.434$$

- Thus, the Gini gain was 0.233

Grinnell College
Statistics

Decision Tree Learning

Building a tree involves iterating between two steps:

1. Finding the optimal splitting point within each variable
2. Selecting the variable to split on

Even if an exhaustive approach were taken, this can be done fairly quickly as there are $n-1$ possible split points per variable and p variables

- Modern tree solving algorithms are beyond the scope of this course, but as you might image they do not check every possible split

Grinnell College
Statistics

Regression tasks (numerical outcomes)

Applying the decision tree algorithm to data with a numeric outcome involves the following changes:

- Predicted outcomes are the average outcome in a node
- Mean squared error is used instead of Gini impurity as a measure of impurity

Other splitting criteria, such as mean absolute error or Poisson deviance are available in `sklearn`

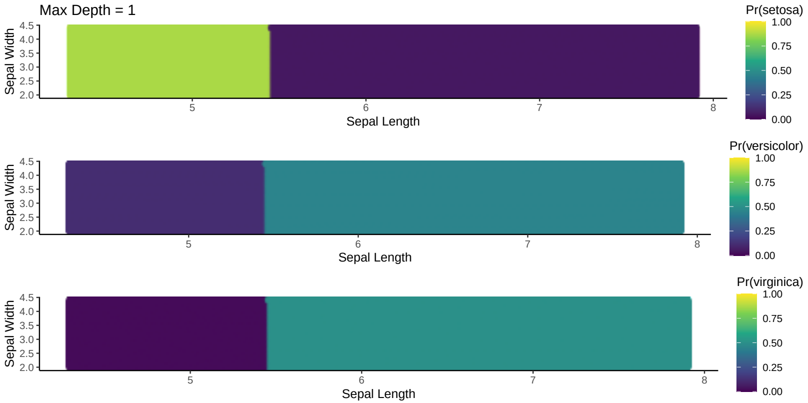
Stopping the algorithm

Decision trees can be grown until every terminal node is perfectly pure; however, such trees will be very overfit to the training data. We can manipulate the bias-variance trade-off in a fitted tree in the following ways:

1. Restricting the maximum depth of the tree (ie: the number of sequential rules)
2. Allowing only nodes of sufficient size be eligible for splitting
3. Requiring a certain improvement in purity for a split to occur

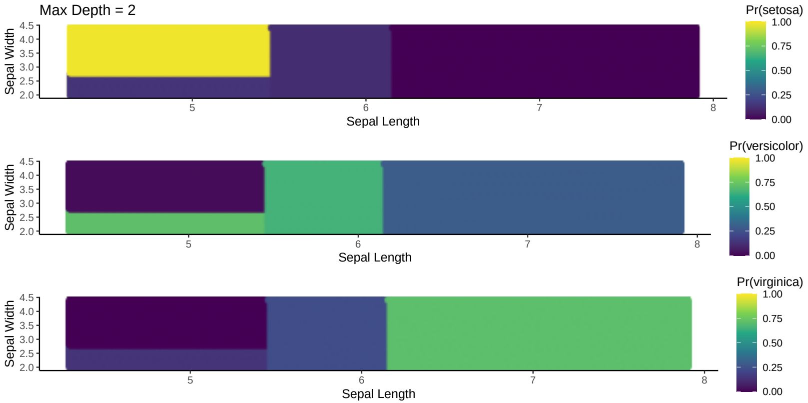
Generally, maximum depth is the most important factor to consider as it directly relates to the complexity of a tree

Examples (bias-variance tradeoff)



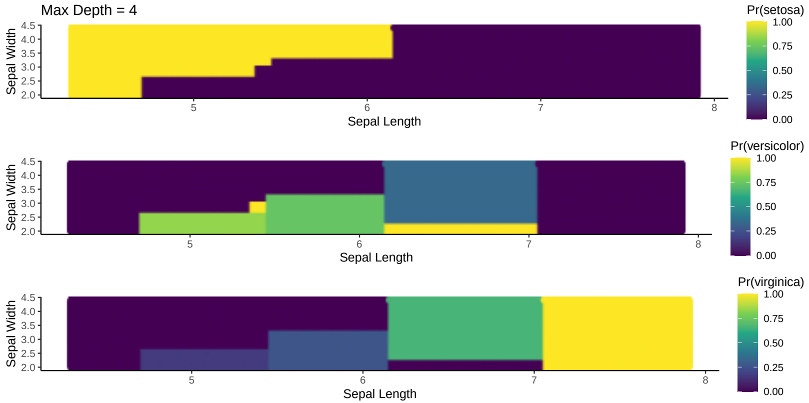
Grinnell College Statistics

Examples (bias-variance tradeoff)



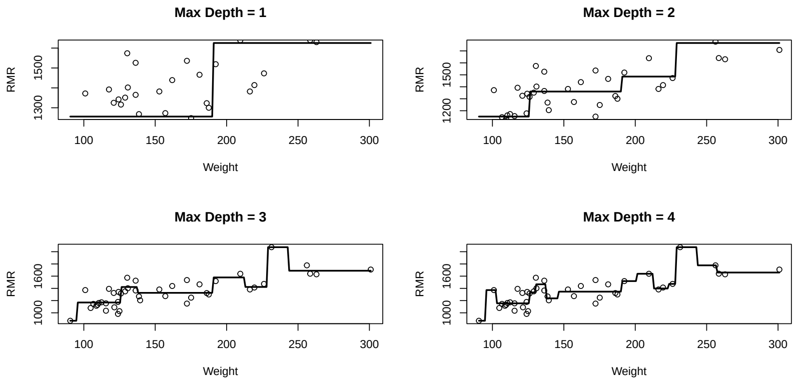
Grinnell College
Statistics

Examples (bias-variance tradeoff)



Examples (bias-variance tradeoff)

Resting metabolism data (learning how to predict RMR using body weight):



Practical guidance

We've now introduced two "classic" machine learning algorithms, KNN and decision trees:

- Decision trees
  - structured, but can still learn non-linear patterns
  - more easily interpretable
  - fairly robust to outliers and missing values
  - prone to overfitting at higher max depths
- KNN
  - very flexible, but produces irregular/inconsistent classification boundaries
  - sensitive to re-scaling
  - computationally inefficient with large data sets

Cross-validation provides an objective strategy for choosing between these models, but we should be aware of their strengths/weaknesses.