Decision Trees

Predictive Modeling & Statistical Learning

Gaston Sanchez

Introduction

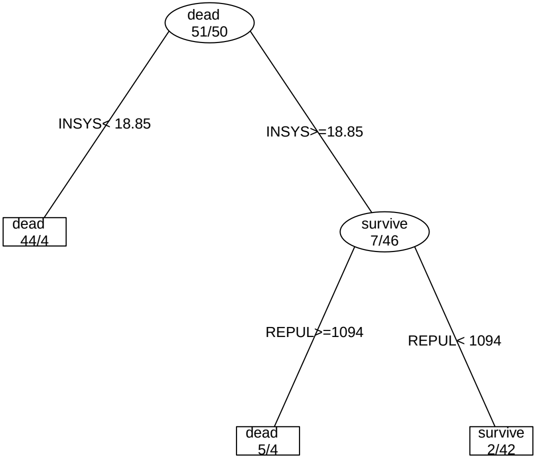
About decision trees

Last time we introduced decision trees and talked about them in a “superficial” way.

In these slides we are going to discuss all the details about how the actual tree-growing process takes places.

Keep in mind that most of what we will talk about today is not in ISL or in APM.

Classification Tree for `infarctus` data



About decision trees

Building a tree involves successively dividing the set of objects with the help of predictors $X_1, X_2, \ldots, X_p$, with the goal of obtaining final segments as much homogeneous as possible with respect to the response variable Y.

The successive divisions (or splits) of the set of objects is based on the principle of **recursive partitioning**.

About decision trees

“We start by choosing the variable which, by its categories, provides the best separation of the objects in each class, thus creating subsets, called nodes, each containing the largest possible proportion of objects in a single class.

The same operation is then repeated on each new node obtained, until no further separation of the objects is posible or desirable.

The construction is such that each of the terminal nodes (the leaves) mainly consists of the individuals of a single class. The set of rules for all the leaves forms the classification model.” (Tuffery, 2011, p. 313)

Building Trees

How to build a tree?

Two main concerns in tree-building methods:

- How to choose the splits?
- How big the tree should grow?

Building Tree Recipe

General algorithm (page 309, ISL) to build a tree (main steps):

1. Use recursive binary splitting to grow a large tree on the training data
2. Apply cost complexity pruning to the large tree in order to obtain a sequence of best subtrees, as a function of α
3. Use k-fold cross-validation to choose α.
4. Return the subtree from step 2 that corresponds to the chosen value of α.

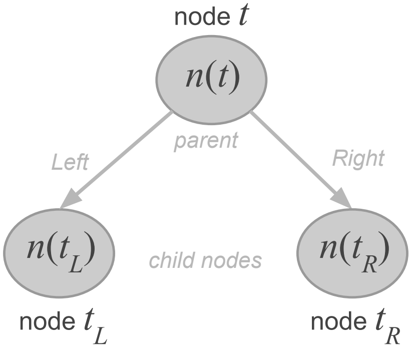
There's a lot going on in each of the steps!!!

To grow a tree ...

There are 2+1 main issues to consider:

- A **recursive splitting** mechanism.
  (we'll focus on binary splits)
- Decide whether a node is declared as *terminal*, or needs to be split.
- Give a class label to each terminal node.

Binary split of a node



A parent node t with $n(t)$ elements is split in two child nodes: t_L and t_R with $n(t_L)$ and $n(t_R)$ elements, respectively.

Splitting Mechanism

Getting the partitions: the soul of the tree growing process lies in the splitting mechanism:

- Establish the set of admissible splits for each node
- Define a split criterion (based on a measure of impurity)
- Measure the quality of a split
- Select the “best” split

About the splitting mechanism

- The main idea is to partition the set of objects into subsets.
- The partitions (or splits) are based on one predictor at a time.
- The goal is obtain nodes or segments that are more pure than the parent.
- In other words: get one or both more homogenous child nodes than the parent.

Recursive Partitioning

Recursivity

- For each available descriptor variable X_j
  - Determine the possible number of partitions
  - For each partition:
    - Measure impurity of nodes
    - Combine impurities into impurity of partition
  - Select best candidate partition
- Select “best of the best” partitions
- Repeat this process on each generated node

General Description

The usual process to build a binary tree is as follows. A search of all the predictor variables is done in order to find the variable that best splits the data in two segments. This process is applied separately to each subgroup, and so on recursively until the subgroups either reach a minimum size or until no improvement can be made.

Stop Criteria

Usually there is one or more stopping criteria. The stopping rules are based on some restriction such as:

- Run out of predictors
- Minimum number of elements in a node
- Number of final nodes in the tree
- Maximum depth level for a tree
- Threshold based on statistical significance
- Threshold for gain impurity

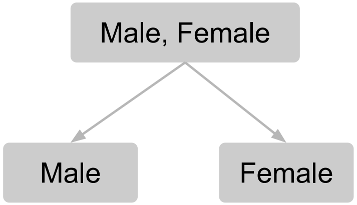
Number of Binary Splits

Number of Binary Splits

The number of binary splits depends on the nature of the predictor

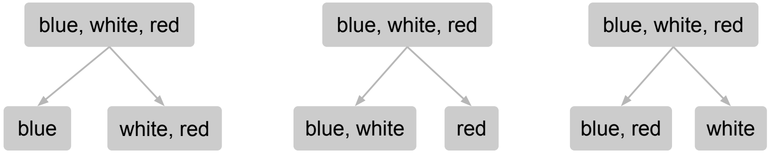
Predictor	2-way splits
binary (2 levels)	1
nominal (q levels)	$2^{q-1} - 1$
ordinal (q levels)	$q - 1$
quantitative (q unique values)	$q - 1$

Example: binary splits for binary variable



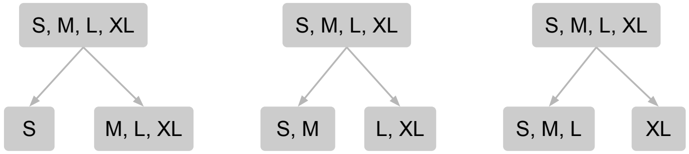
A binary variable has one possible binary split

Example: binary splits for nominal variable



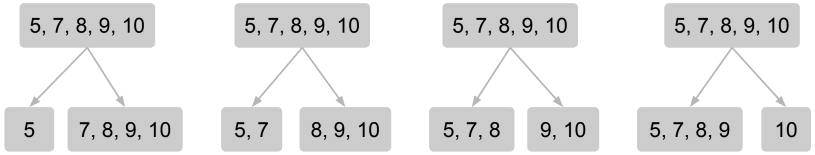
A nominal variable with $q = 3$ levels has $2^{q-1} - 1 = 3$ binary splits

Example: binary splits for ordinal variable



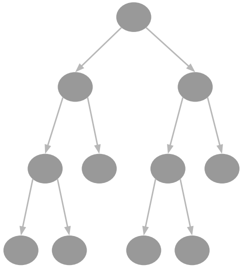
An ordinal variable with $q = 4$ levels has $q - 1 = 3$ binary splits

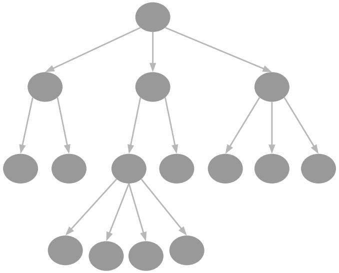
Example: binary splits for quantitative variable



A quantitative variable with $q = 5$ different values has $q - 1 = 4$ binary splits

About Binary Trees

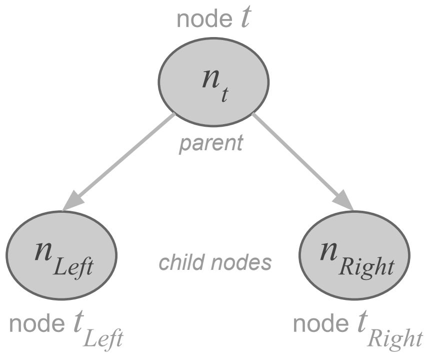

Binary tree
(2-way splits)


N-ary tree
(n-way splits)

Binary trees are preferred over n-ary trees because the latter tends to produce nodes of small size, and overfitting.

Split Criteria

Split of a node



A parent node t with n_t elements is split in two child nodes: t_{Left} and t_{Right} with n_{Left} and n_{Right} elements, respectively.

Selection of the optimal partition

We choose to split a segment (or node) t in a way that makes the "unorganized" objects more organized (from heteregenous to more homogeneous).

One way to organize this messines is to measure the information. Using information theory, you can measure the information before and after the splits.

Measures of Impurity

Classfication

For classification trees, the most common criteria are

- Shannon's Entropy
- Gini's Impurity

Regression

For regression trees, the typical measure is the variance

Think about these indices as measures of impurity, heterogeneity or "disorder" in a set.

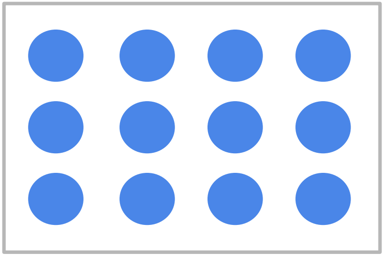
Entropy

Entropy

To understand Entropy of a set, it is better to think in terms of probability.

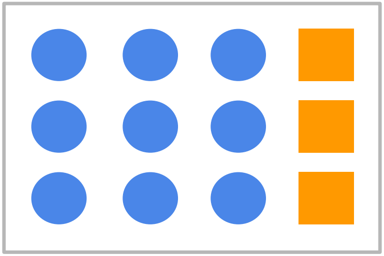
More concretely, think about the uncertainty associated with guessing the result if you were to make a random selection from a given set.

Entropy



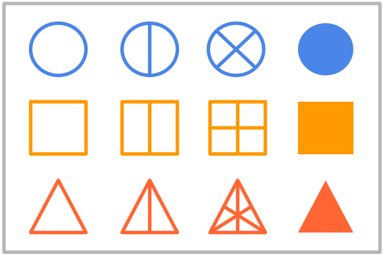
Consider a set with 12 objects (i.e. blue balls). If you randomly select an object from this set, you would have zero uncertainty. So this set has zero entropy.

Entropy



Now consider set 2. If you randomly select an object from this set, you would have some uncertainty about which object is selected because not all objects are the same.

Entropy



If you randomly select an object from this set, you would be very uncertain about any prediction as there are 12 possible outcomes, each of which is equally likely.

Entropy

- We can transform probabilities into *entropy* values. How?
- An outcome with a large probability should map to a low entropy value.
- An outcome with a small probability should map to a large entropy value.
- The logarithm (or log) function does almost exactly this transformation.
- Ignoring the negative values of $log_2(x), \forall x \in [0, 1]$, the magnitude of the number is a good entropy value.

Shannon's Entropy Formula

$$H(t) = -\sum_{k=1}^{h} \{p_k \times log_b(p_k)\}$$

Shannon's Entropy Formula

$$H(t) = -\sum_{k=1}^{h} \{p_k \times log_b(p_k)\}$$

where:

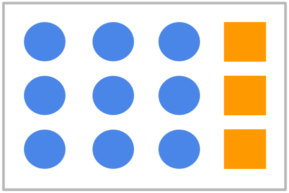
- p_k is the probability that the outcome of randomly selecting an element of set t is of the type k
- h is the number of different types of objects in the set
- b is an arbitrary logarithmic base
- the minus sign converts negative values to positive entropy values

Shannon's Entropy Formula

Entropy is the expected value of all the information of all possible values of objects.

The higher the entropy, the more mixed up the data is.

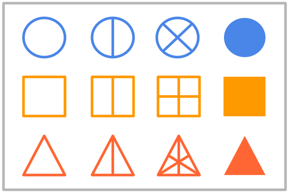
Calculating Entropy



Entropy of set t with two types of objects with probabilities: $p_1 = 9/12$ and $p_2 = 3/12$:

$$H(t) = -\left\{\frac{9}{12} \times \log_2(\frac{9}{12})\right\} - \left\{\frac{3}{12} \times \log_2(\frac{3}{12})\right\} = 0.8112781$$

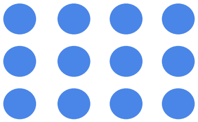
Calculating Entropy

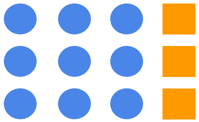


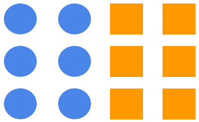
Entropy of set t with 12 types of objects, each having probability: $p_k = 1/12$

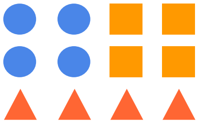
$$H(t) = -\sum_{k=1}^{12}\left\{\frac{1}{12} \times \log_2(\frac{1}{12})\right\} = 3.584963$$

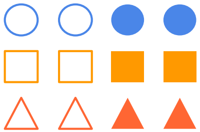
Entropies for different sets

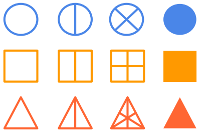

Entropy = 0


Entropy = 0.81


Entropy = 1


Entropy = 1.58


Entropy = 2.58


Entropy = 3.58

Gini's Impurity

Gini Impurity Formula

$$Gini(t) = \sum_{k=1}^{h} p_k(1 - p_k)$$

Gini Impurity Formula

$$Gini(t) = \sum_{k=1}^{h} p_k(1 - p_k)$$

where:

- p_k is the probability that the outcome of randomly selecting an element of set t is of the type k
- h is the number of different types of objects in the set t

Gini Impurity Formula

You may find alternative formulas for Gini impurity:

$$Gini(t) = \sum_{k=1}^{h} p_k(1 - p_k) = 1 - \sum_{k=1}^{h} p_k^2 = \sum_{k \neq l} p_k p_l$$

where:

- p_k is the probability that the outcome of randomly selecting an element of set t is of the type k
- h is the number of different types of objects in the set t

Gini Impurity Formula

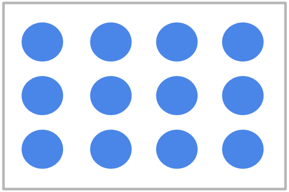
The Gini Impurity

$$Gini(t) = \sum_{k=1}^{h} p_k(1 - p_k) = \sum_{k \neq l} p_k p_l$$

measures how often a randomly chosen element from the set t would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset.

The Gini Impurity can be computed by summing the probability p_k of an item with label k being chosen times the probability $1 - p_k$ of a mistake in categorizing that item.

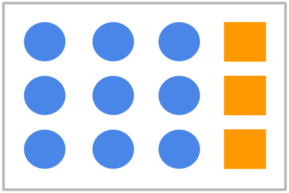
Calculating Gini Impurity



Gini impurity of set t with one type of object with probability $p_1 = 1$:

$$Gini(t) = (1)(1 - 1) = 0$$

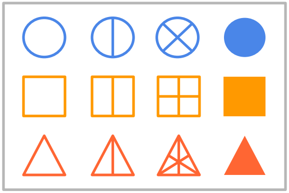
Calculating Gini Impurity



Gini impurity of set t with two types of objects with probabilities: $p_1 = 9/12$ and $p_2 = 3/12$:

$$Gini(t) = \left(\frac{9}{12}\right)\left(1 - \frac{9}{12}\right) + \left(\frac{3}{12}\right)\left(1 - \frac{3}{12}\right) = 0.375$$

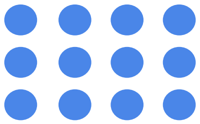
Calculating Gini Impurity

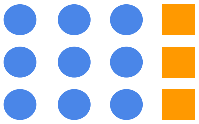


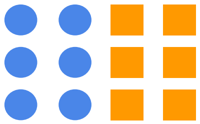
Gini impurity of set t with 12 types of objects, each having probability: $p_k = 1/12$

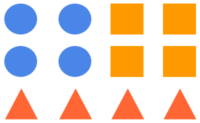
$$Gini(t) = 12 \times \left(\frac{1}{12}\right)\left(1 - \frac{1}{12}\right) = 0.916$$

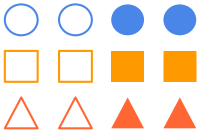
Gini Impurities for different sets

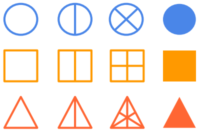

Gini = 0


Gini = 0.375


Gini = 0.5

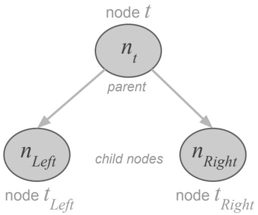

Gini = 0.666


Gini = 0.833


Gini = 0.916

Selection of the optimal partition

Maximize the decrement of impurity Δ_{imp} between the parent node t and its children ($Left$ and $Right$ partitions):



$$\Delta_{imp}(t) = imp(t) - \left(\frac{n_{Left}}{n_t}\right) imp(t_{Left}) - \left(\frac{n_{Right}}{n_t}\right) imp(t_{Right})$$

where: $n_t = n_{Left} + n_{Right}$

Stopping Rules

Tree Growing Process

Definition of node termination rule:
Pre-pruning: implies that the decision of when to stop the growth of a tree is made prospectively.
Post-pruning: refers to reducing the size of a fully expanded tree by pruning some of its branches retrospectively.

Cost Complexity

Notation

Symbol	Description
n	number of objects (size of root node)
t	a given segment or node
n_k	total number of objects of class k
$n(t)$	number of objects in segment t
$n_k(t)$	number of objects of class k in segment t
p_k	probability or proportion of objects in class k
$p(t)$	probability of belonging to segment t
$p(k\|t)$	probability of belonging to class k inside segment t
$\gamma(k\|h)$	cost of missclassiflying an object of class h to class k

Probabilities in terms of frequency values

$$p_j = \frac{n_j}{n}$$

$$p(k|t) = \frac{n_j(t)}{n(t)}$$

$$p(t) = \frac{n(t)}{n}$$

Probabilities in terms of frequency values

The mean cost of classifying to group G_k an object randomly selected in segment t:

$$c_k(t) = \sum_h \gamma(k|h)p(h|t) = \sum_{h \neq k} \frac{n_h(t)}{n(t)}$$

$c_k(t)$ is the proportion of misclassification inside node t when we label it as class G_k

To minimize $c_k(t)$, the objects of segment t should be classified to the the most frequent category of the node.

Misclassification Cost

Define:

$$c(t) = \underset{k}{min}\left\{c_k(t)\right\}$$

The cost of misclassfication in a segment t is:

$$C(t) = p(t)c(t) = \frac{1}{n}\left[n(t) - \underset{k}{max}\left\{n_k(t)\right\}\right]$$

Tree for `iris` data

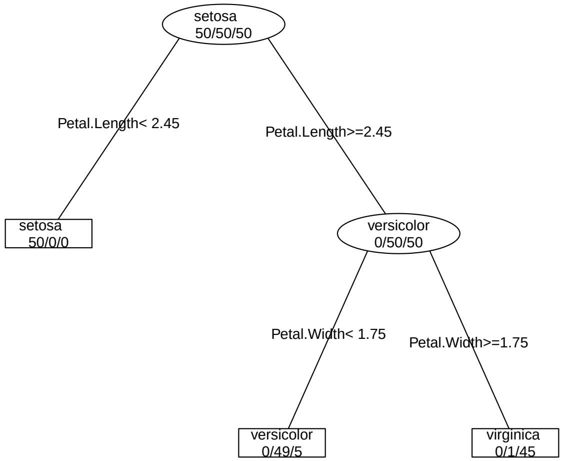
```
# fit tree
iris_tree <- rpart(Species ~ ., data = iris)

iris_tree
```

```
## n= 150
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 150 100 setosa (0.33333333 0.33333333 0.33333333)
##   2) Petal.Length< 2.45 50   0 setosa (1.00000000 0.00000000 0.00000000) *
##   3) Petal.Length>=2.45 100  50 versicolor (0.00000000 0.50000000 0.50000000)
##     6) Petal.Width< 1.75 54   5 versicolor (0.00000000 0.90740741 0.09259259) *
##     7) Petal.Width>=1.75 46   1 virginica (0.00000000 0.02173913 0.97826087) *
```

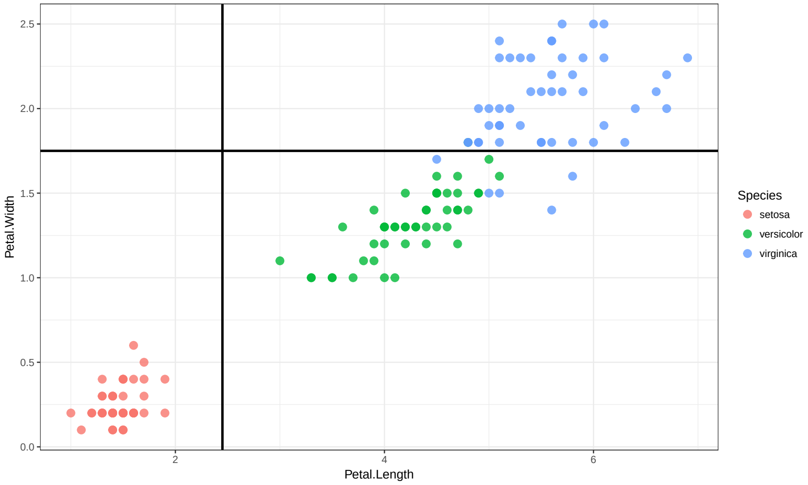
Tree for iris data

```
# fit tree
iris_tree <- rpart(Species ~ ., data = iris)

iris_tree


## n= 150
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 150 100 setosa (0.33333333 0.33333333 0.33333333)
##   2) Petal.Length< 2.45 50   0 setosa (1.00000000 0.00000000 0.00000000) *
##   3) Petal.Length>=2.45 100  50 versicolor (0.00000000 0.50000000 0.50000000)
##     6) Petal.Width< 1.75 54   5 versicolor (0.00000000 0.90740741 0.09259259) *
##     7) Petal.Width>=1.75 46   1 virginica (0.00000000 0.02173913 0.97826087) *
```

Tree for `iris` data



Space partitions for `iris` data



References

- **Classification and Regression Trees** by Breiman et al (1984).
- **Data Mining and Statistics for Decision Making** by Stephane Tuffery (2011). *Chapter 11: Classification and prediction methods*. Wiley
- **Fundamentals of Machine Learning for Predictive Data Analytics** by Kelleher et al (2015). *Chapter 4: Information-based Learning*.
- **Data Mining: Practical Machine Learning Tools and Techniques** by Witten and Frank (2005). *Chapter 4: Algorithms. The basic methods*. Morgan Kaufmann.

References (French literature)

- **Analyse discriminante** by Mireille Bardos (2001). *Chapter 4: Arbres de partitionnement*. Dunod, Paris.
- **Statistique Exploratoire Multidimensionnelle** by Lebart et al (2004). *Chapter 3, section 3.5: Segmentation*. Dunod, Paris.
- **Statistique explicative appliquee** by Nakache and Confais (2003). *Chapter 8: Methode de segmentation CART*. Editions Technip, Paris.
- **Statistique: Methodes pour decrire, expliquer et prevoir** by Michel Tenenhaus (2008). *Chapter 13: Methodes de segmentation*. Editions Technip, Paris.
- **Modelisation predictive et Apprentisaage statistique** by Stephane Tuffery (2015). *Chapter 9: L'arbre de decision CART*. Editions Technip, Paris.