Data Mining 2024
Classification Trees (1)

Ad Feelders

Universiteit Utrecht

Classification

Predict the class of an object on the basis of some of its attributes.
For example, predict:

- Good/bad credit for loan applicants, using
  - income
  - age
  - ...
- Spam/no spam for e-mail messages, using
  - % of words matching a given word (e.g. "free")
  - use of CAPITAL LETTERS
  - ...
- Music Genre (Rock, Techno, Death Metal, ...) based on audio features and lyrics.

Building a classification model

The basic idea is to build a classification model using a set of training examples. Each training example contains attribute values and the corresponding class label.

There are many techniques to do that:

- Statistical Techniques
  - Discriminant Analysis
  - Logistic Regression
- Data Mining/Machine Learning
  - Classification Trees
  - Bayesian Network Classifiers
  - Neural Networks
  - Support Vector Machines
  - ...

Strong and Weak Points of Classification Trees

Strong points:

- Are easy to interpret (if not too large).
- Select relevant attributes automatically.
- Can handle both numeric and categorical attributes.

Weak point:

- Single trees are usually not among the top performers.

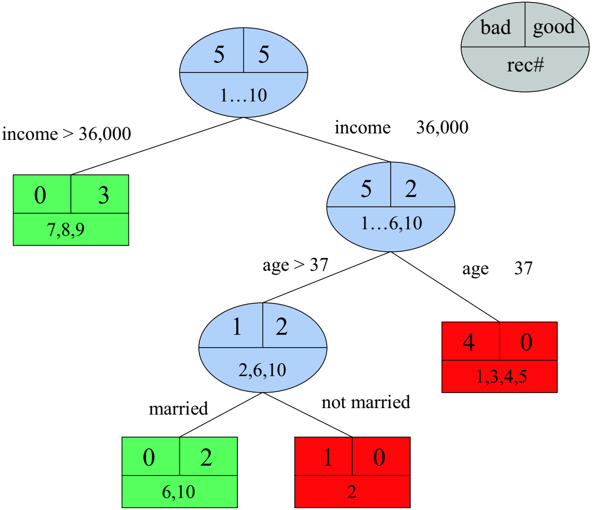
However:

- Averaging multiple trees (bagging, boosting, random forests) can bring them back to the top!
- But ease of interpretation suffers as a consequence.

Example: Loan Data

Record	age	married?	own house	income	gender	class
1	22	no	no	28,000	male	bad
2	46	no	yes	32,000	female	bad
3	24	yes	yes	24,000	male	bad
4	25	no	no	27,000	male	bad
5	29	yes	yes	32,000	female	bad
6	45	yes	yes	30,000	female	good
7	63	yes	yes	58,000	male	good
8	36	yes	no	52,000	male	good
9	23	no	yes	40,000	female	good
10	50	yes	yes	28,000	female	good

Credit Scoring Tree

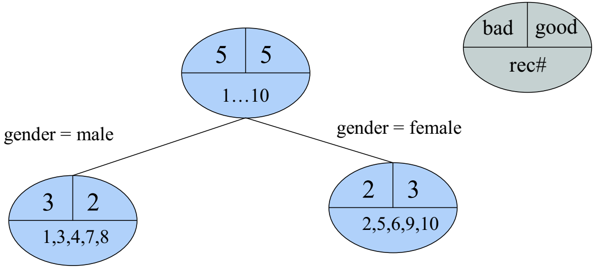


Cases with income $> 36,000$

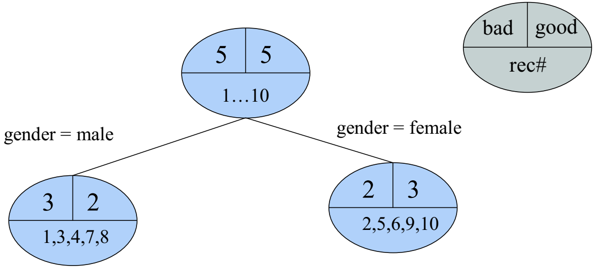
Record	age	married?	own house	income	gender	class
1	22	no	no	28,000	male	bad
2	46	no	yes	32,000	female	bad
3	24	yes	yes	24,000	male	bad
4	25	no	no	27,000	male	bad
5	29	yes	yes	32,000	female	bad
6	45	yes	yes	30,000	female	good
7	63	yes	yes	58,000	male	good
8	36	yes	no	52,000	male	good
9	23	no	yes	40,000	female	good
10	50	yes	yes	28,000	female	good

Partitioning the attribute space

Why not split on gender in the root node?



Why not split on gender in the root node?



Intuitively: learning the value of gender doesn't provide much information about the class label.

Impurity of a node

- We strive towards nodes that are *pure* in the sense that they only contain observations of a single class.
- We need a measure that indicates "how far" a node is removed from this ideal.
- We call such a measure an *impurity* measure.

Impurity function

The impurity $i(t)$ of a node t is a function of the relative frequencies of the classes in that node:

$$i(t) = \phi(p_1, p_2, \ldots, p_J)$$

where the $p_j (j = 1, \ldots, J)$ are the relative frequencies of the J different classes in node t.

Sensible requirements of any quantification of impurity:

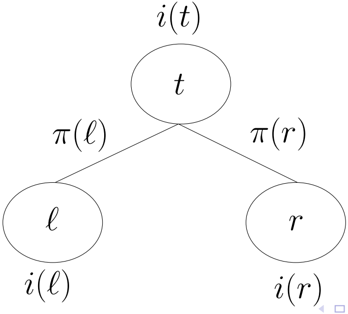
1. Should be at a maximum when the observations are distributed evenly over all classes.
2. Should be at a minimum when all observations belong to a single class.
3. Should be a symmetric function of $p_1, \ldots, p_J$.

Quality of a split (test)

We define the quality of binary split s in node t as the *reduction* of impurity that it achieves

$$\Delta i(s,t) = i(t) - \{\pi(\ell)i(\ell) + \pi(r)i(r)\}$$

where ℓ is the left child of t, r is the right child of t, $\pi(\ell)$ is the proportion of cases sent to the left, and $\pi(r)$ the proportion of cases sent to the right.



Well known impurity functions

Impurity functions we consider:

- Resubstitution error
- Gini-index
- Entropy

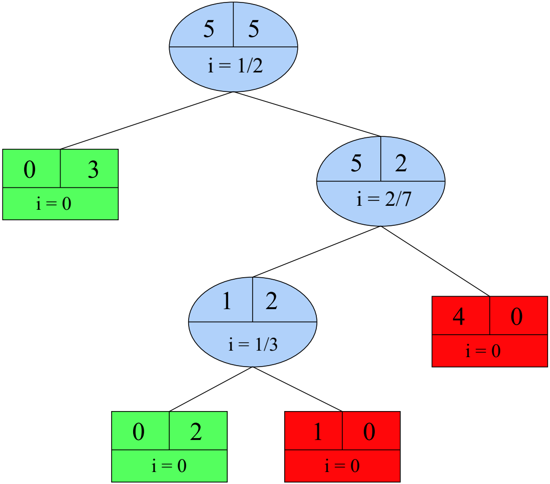
Resubstitution error

Measures the fraction of cases that is classified incorrectly if we assign every case in node t to the majority class in that node. That is

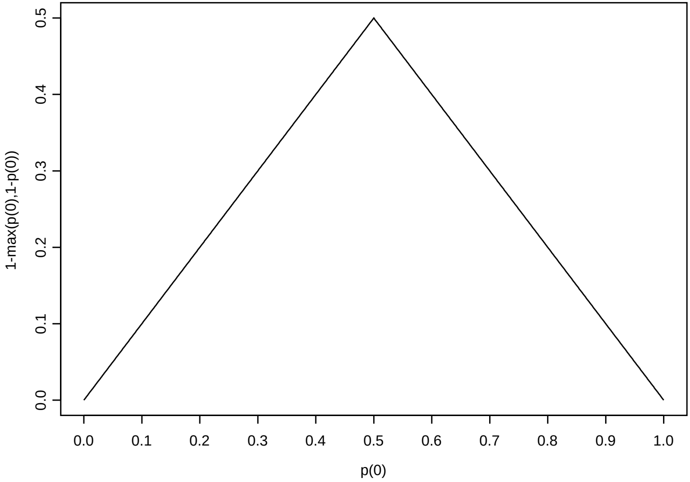
$$i(t) = 1 - \max_j p(j|t)$$

where $p(j|t)$ is the relative frequency of class j in node t.

Resubstitution error: credit scoring tree



Graph of resubstitution error for two-class case



Resubstitution error

Questions:

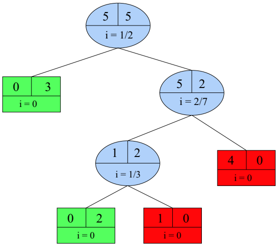
- Does resubstitution error meet the sensible requirements?

Resubstitution error

Questions:

- Does resubstitution error meet the sensible requirements?
- What is the impurity reduction of the second split in the credit scoring tree if we use resubstitution error as impurity measure?

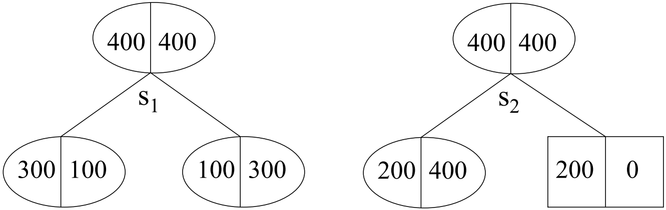
Impurity Reduction



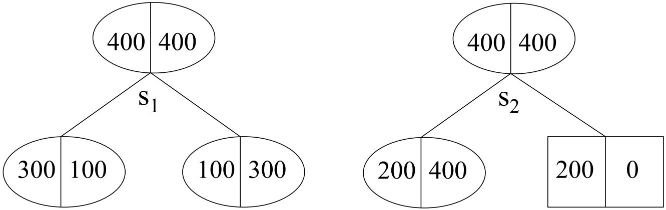
Impurity reduction of second split (using resubstitution error):

$$\begin{aligned}\Delta i(s,t) &= i(t) - \{\pi(\ell)i(\ell) + \pi(r)i(r)\} \\ &= \frac{2}{7} - \left(\frac{3}{7} \times \frac{1}{3} + \frac{4}{7} \times 0\right) \\ &= \frac{2}{7} - \frac{1}{7} = \frac{1}{7}\end{aligned}$$

Which split is better?



Which split is better?



These splits have the same resubstitution error, but s_2 is commonly preferred because it creates a leaf node.

Class of suitable impurity functions

- Problem: resubstitution error only decreases at a *constant* rate as the node becomes purer.
- We need an impurity measure which gives greater rewards to purer nodes. Impurity should decrease at an *increasing* rate as the node becomes purer.
- Hence, impurity should be a strictly *concave* function of $p(0)$.

We define the class $\mathcal{F}$ of impurity functions (for two-class problems) that has this property:

1. $\phi(0) = \phi(1) = 0$ (minimum at $p(0) = 0$ and $p(0) = 1$)
2. $\phi(p(0)) = \phi(1 - p(0))$ (symmetric)
3. $\phi''(p(0)) < 0, 0 < p(0) < 1$ (strictly concave)

Impurity function: Gini index

For the two-class case the Gini index is

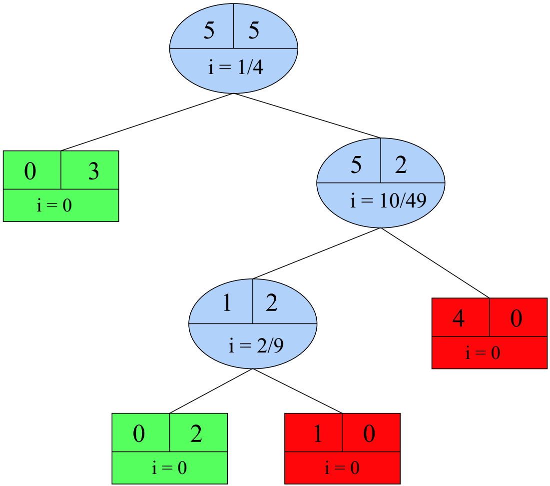
$$i(t) = p(0|t)p(1|t) = p(0|t)(1 - p(0|t))$$

Question 1: Check that the Gini index belongs to $\mathcal{F}$.

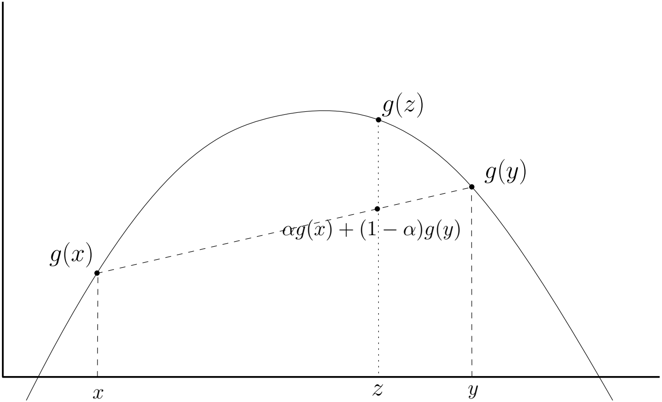
Question 2: Check that if we use the Gini index, split s_2 is indeed preferred.

Note: The variance of a Bernoulli random variable with probability of success p is $p(1 - p)$. Hence we are attempting to minimize the variance of the class distribution.

Gini index: credit scoring tree



Can impurity increase?



A concave function g. For any x and y, the line segment connecting $g(x)$ and $g(y)$ is below the graph of g. $z = \alpha x + (1-\alpha)y$.

$$g(\alpha x + (1-\alpha)y) \geq \alpha g(x) + (1-\alpha)g(y)$$

Can impurity increase?

Is it possible that a split makes things worse, i.e. $\Delta i(s, t) < 0$?

Not if $\phi \in \mathcal{F}$. Because ϕ is a concave function, we have

$$\phi(p(0|\ell)\pi(\ell) + p(0|r)\pi(r)) \geq \pi(\ell)\phi(p(0|\ell)) + \pi(r)\phi(p(0|r))$$

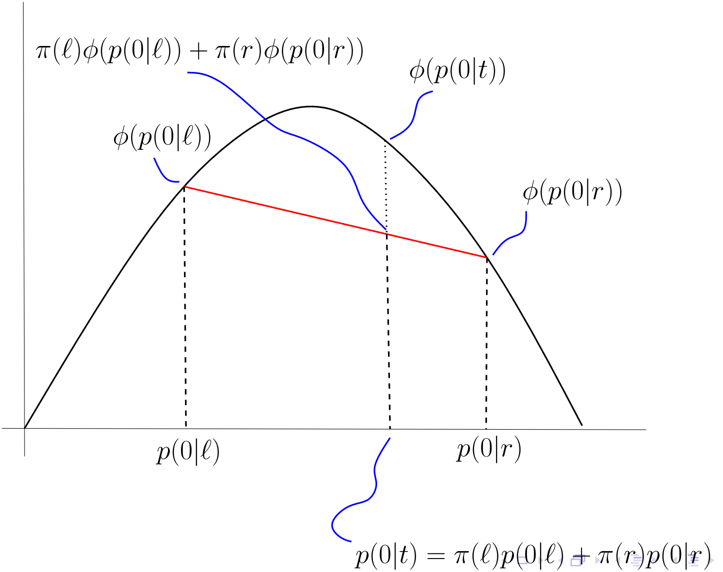
Since

$$p(0|t) = p(0|\ell)\pi(\ell) + p(0|r)\pi(r)$$

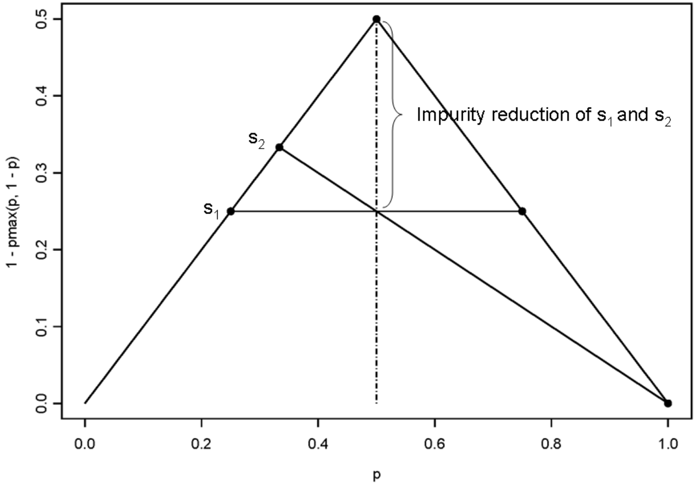
it follows that

$$\phi(p(0|t)) \geq \pi(\ell)\phi(p(0|\ell)) + \pi(r)\phi(p(0|r))$$

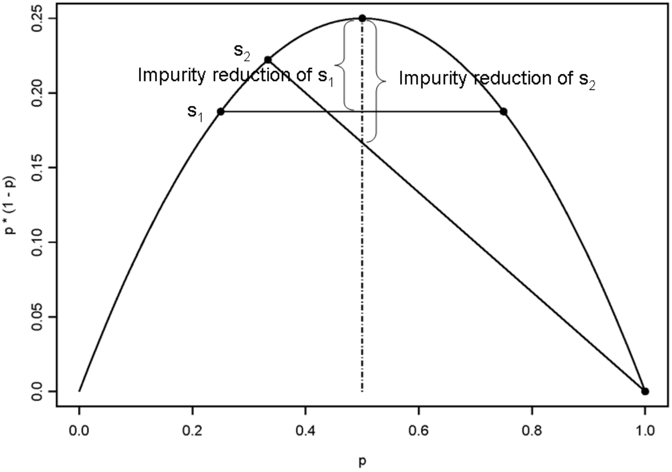
Can impurity increase? Not if ϕ is concave.



Split s_1 and s_2 with resubstitution error



Split s_1 and s_2 with Gini



Impurity function: Entropy

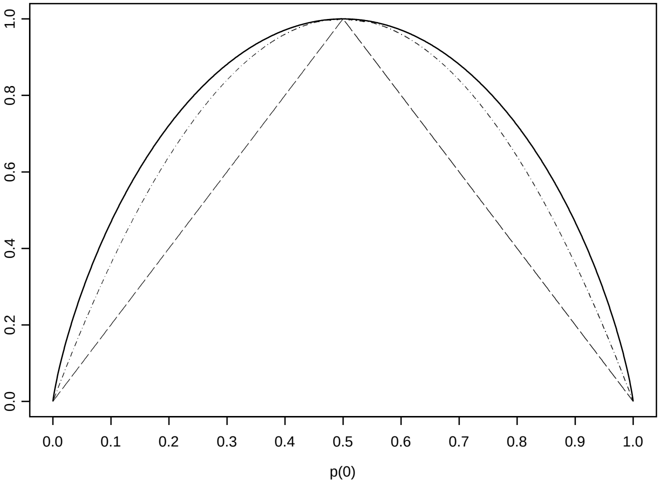
For the two-class case the entropy is

$$i(t) = -p(0|t)\log p(0|t) - p(1|t)\log p(1|t)$$

Question: Check that entropy impurity belongs to $\mathcal{F}$.

Remark: this is the average amount of information generated by drawing (with replacement) an example at random from this node, and observing its class.

Three (rescaled) impurity measures



Entropy (solid), Gini (dot-dash) and resubstitution (dash) impurity.

The set of splits considered

1. Each split depends on the value of only a *single* attribute.
2. If attribute x is numeric, we consider all splits of type $x \leq c$ where c is (halfway) between two consecutive values of x in their sorted order.
3. If attribute x is categorical, taking values in $\{b_1, b_2, \ldots, b_L\}$, we consider all splits of type $x \in S$, where S is any non-empty proper subset of $\{b_1, b_2, \ldots, b_L\}$.

Splits on numeric attributes

There is only a finite number of distinct splits, because there are at most n distinct values of a numeric attribute in the training sample (where n is the number of examples in the training sample).

Example: possible splits on income in the root for the loan data

Income	Class	Quality (split after) $0.25-$
24	B	$0.1(1)(0)+0.9(4/9)(5/9) = 0.03$
27	B	$0.2(1)(0) + 0.8\ (3/8)(5/8) = 0.06$
28	B,G	$0.4(3/4)(1/4) + 0.6(2/6)(4/6) = 0.04$
30	G	$0.5(3/5)(2/5) + 0.5(2/5)(3/5) = 0.01$
32	B,B	$0.7(5/7)(2/7) + 0.3(0)(1) = 0.11$
40	G	$0.8(5/8)(3/8) + 0.2(0)(1) = 0.06$
52	G	$0.9(5/9)(4/9) + 0.1(0)(1) = 0.03$
58	G	

Splits on a categorical attribute

For a categorical attribute with L distinct values there are $2^{L-1} - 1$ distinct splits to consider. Why?

Splits on a categorical attribute

For a categorical attribute with L distinct values there are $2^{L-1} - 1$ distinct splits to consider. Why?

There are $2^L - 2$ non-empty proper subsets of $\{b_1, b_2, \ldots, b_L\}$.

But a subset and the complement of that subset result in the same split, so we should divide this number by 2.

Splitting on categorical attributes: shortcut

For two-class problems, and $\phi \in \mathcal{F}$, we don't have to check all $2^{L-1} - 1$ possible splits. Sort the $p(0|x = b_\ell)$, that is,

$$p(0|x = b_{\ell_1}) \leq p(0|x = b_{\ell_2}) \leq \ldots \leq p(0|x = b_{\ell_L})$$

Then one of the $L - 1$ subsets

$$\{b_{\ell_1}, \ldots, b_{\ell_h}\}, \quad h = 1, \ldots, L - 1,$$

is the optimal split. Thus the search is reduced from computing $2^{L-1} - 1$ splits to computing only $L - 1$ splits.

Splitting on categorical attributes: example

Let x be a categorical attribute with possible values a, b, c, d. Suppose

$$p(0|x = a) = 0.6, p(0|x = b) = 0.4, p(0|x = c) = 0.2, p(0|x = d) = 0.8$$

Sort the values of x according to probability of class 0

$$c \qquad b \qquad a \qquad d$$

We only have to consider the splits: $\{c\}, \{c, b\}$, and $\{c, b, a\}$.

Intuition: put values with low probability of class 0 in one group, and values with high probability of class 0 in the other.

Splitting on numerical attributes: shortcut

Income	Class	Quality (split after) 0.25−
24	B	0.1(1)(0)+0.9(4/9)(5/9) = 0.03
27	B	0.2(1)(0) + 0.8 (3/8)(5/8) = 0.06
28	B,G	0.4(3/4)(1/4) + 0.6(2/6)(4/6) = 0.04
30	G	0.5(3/5)(2/5) + 0.5(2/5)(3/5) = 0.01
32	B,B	0.7(5/7)(2/7) + 0.3(0)(1) = 0.11
40	G	0.8(5/8)(3/8) + 0.2(0)(1) = 0.06
52	G	0.9(5/9)(4/9) + 0.1(0)(1) = 0.03
58	G	

Optimal split can only occur between consecutive values with *different* class distributions.

Splitting on numerical attributes

Income	Class	Quality (split after) 0.25−
24	B	
27	B	$0.2(1)(0) + 0.8\,(3/8)(5/8) = 0.06$
28	B,G	$0.4(3/4)(1/4) + 0.6(2/6)(4/6) = 0.04$
30	G	$0.5(3/5)(2/5) + 0.5(2/5)(3/5) = 0.01$
32	B,B	$0.7(5/7)(2/7) + 0.3(0)(1) = 0.11$
40	G	
52	G	
58	G	

Optimal split can only occur between consecutive values with *different* class distributions.

Segment borders: numeric example

A segment is a block of consecutive values of the split attribute for which the class distribution is identical. Optimal splits can only occur at segment borders.

Consider the following data on numeric attribute x and class label y.
The class label can take on two different values, coded as A and B.

x	8	8	12	12	14	16	16	18	20	20
y	A	B	A	B	A	A	A	A	A	B

The class probabilities (relative frequencies) are:

x	8	12	14	16	18	20
$P(A)$	0.5	0.5	1	1	1	0.5
$P(B)$	0.5	0.5	0	0	0	0.5

So we obtain the segments: $(8, 12)$, $(14, 16, 18)$ and (20).
Only consider the splits: $x \leq 13$ and $x \leq 19$
Ignore: $x \leq 10$, $x \leq 15$ and $x \leq 17$

Caveat

1. In the first practical assignment we use the parameters `nmin` and `minleaf` to stop tree growing early.
2. A split is not allowed to produce a child node with less than `minleaf` observations.
3. The segment borders algorithm doesn't combine very well with the `minleaf` constraint.
4. Better use the "brute force" approach in the assignment.

Basic Tree Construction Algorithm (control flow)

Construct tree

nodelist $\leftarrow$ {{training data}}
Repeat
 current node $\leftarrow$ select node from nodelist
 nodelist $\leftarrow$ nodelist $-$ current node
 if impurity(current node) > 0
 then
  S $\leftarrow$ set of candidate splits in current node
  s* $\leftarrow \arg\max_{s \in S}$ impurity reduction(s,current node)
  child nodes $\leftarrow$ apply(s*,current node)
  nodelist $\leftarrow$ nodelist $\cup$ child nodes
 fi
Until nodelist $= \varnothing$

Practical Assignment 1

- Teams of 3 students.
- Program a classification tree algorithm from scratch.
- In Python or R.
- Use of libraries for general data processing or manipulating tree data structures is allowed
  (e.g. `numpy`, `pandas` in Python, `dplyr`, `data.tree` in R).
- `https://ics-websites.science.uu.nl/docs/vakken/mdm/practicum.html`
- Use of generative AI tools is not allowed, except for improving the writing of the report.
- Start with "Getting started with ...".
- Computer lab in BBG, rooms 201/219/223.