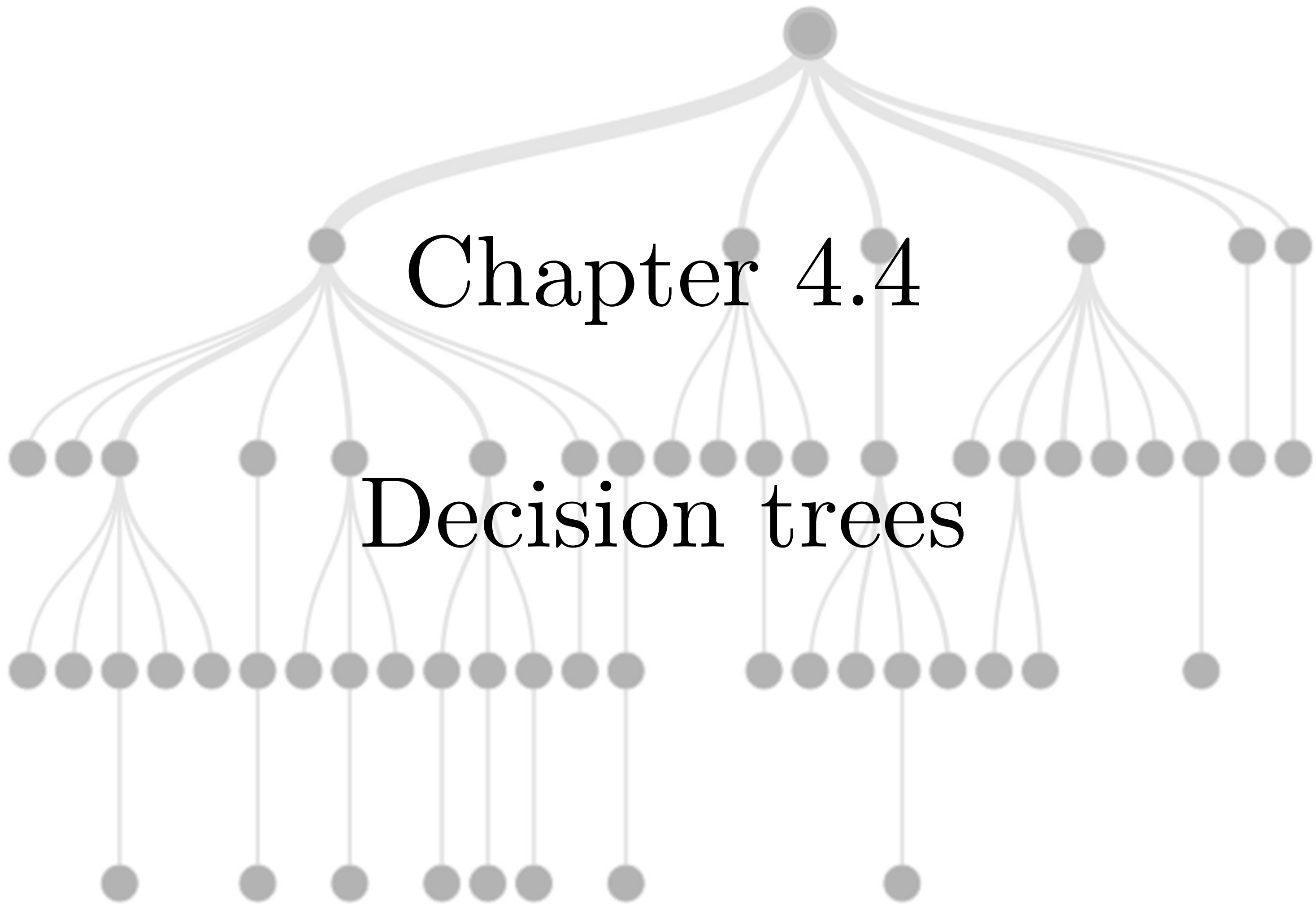


Chapter 4.4

Decision trees

Recommended bibliography: *Pattern Classification (2nd ed) by R. O. Duda, P. E. Hart and D. G. Stork, John Wiley & Sons, 2000*, Chapter 8

Credits: Some figures are taken from *Pattern Classification (2nd ed) by R. O. Duda, P. E. Hart and D. G. Stork, John Wiley & Sons, 2000* with the permission of the authors

Course overview

Pre-processing → Feature selection → Classification

Techniques
PCA (lecture 3) MDA (lecture 3)

	Density function for classes	Classifier
Supervised	Parametrized Gaussian	Linear discriminant Quadratic discriminant[1] (lecture 2)
	Non-parametric	K-nearest neighbours[2] Fuzzy logic (lecture 4)
	Unknown	SVM[3] Neural networks[3] Decision trees[4] (lecture 4)
Unsupervised	Parametrized Gaussian	K-means (lecture 6)
	Unknown	Clustering (lecture 6)

1. Useful only if covariance matrices are not rank deficient.
2. Useful with the number of features is very large, even larger that the number of training vectors.
3. Imposes a structure to the classifier irrespective of the training data base.
4. Useful when non-numeric features are present.

INDEX

1 Introduction

- Previous studied classification methods work with real-valued feature vectors and compute some metric from them: Distance, Similarity, etc. Decision Tree-based methods are non metric.

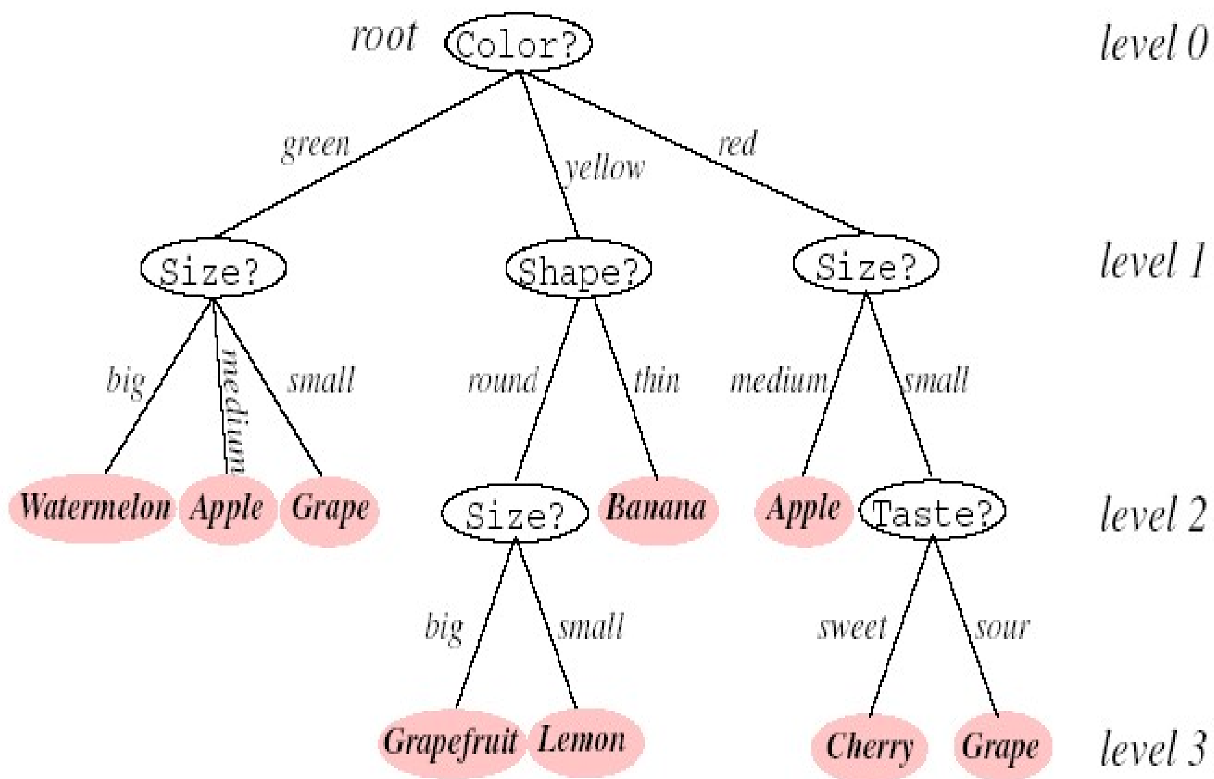
- Alternatives: List of properties or discrete features, forming a d-tuple (pattern of attributes).

- Discrete problems solved with Decision Trees: Rule-based or Syntactic Pattern Recognition.

Nomenclature. RFV: Real Feature Vector
AFV: Attribute Feature Vector

Example 1: Attribute Feature Vector (AFV)

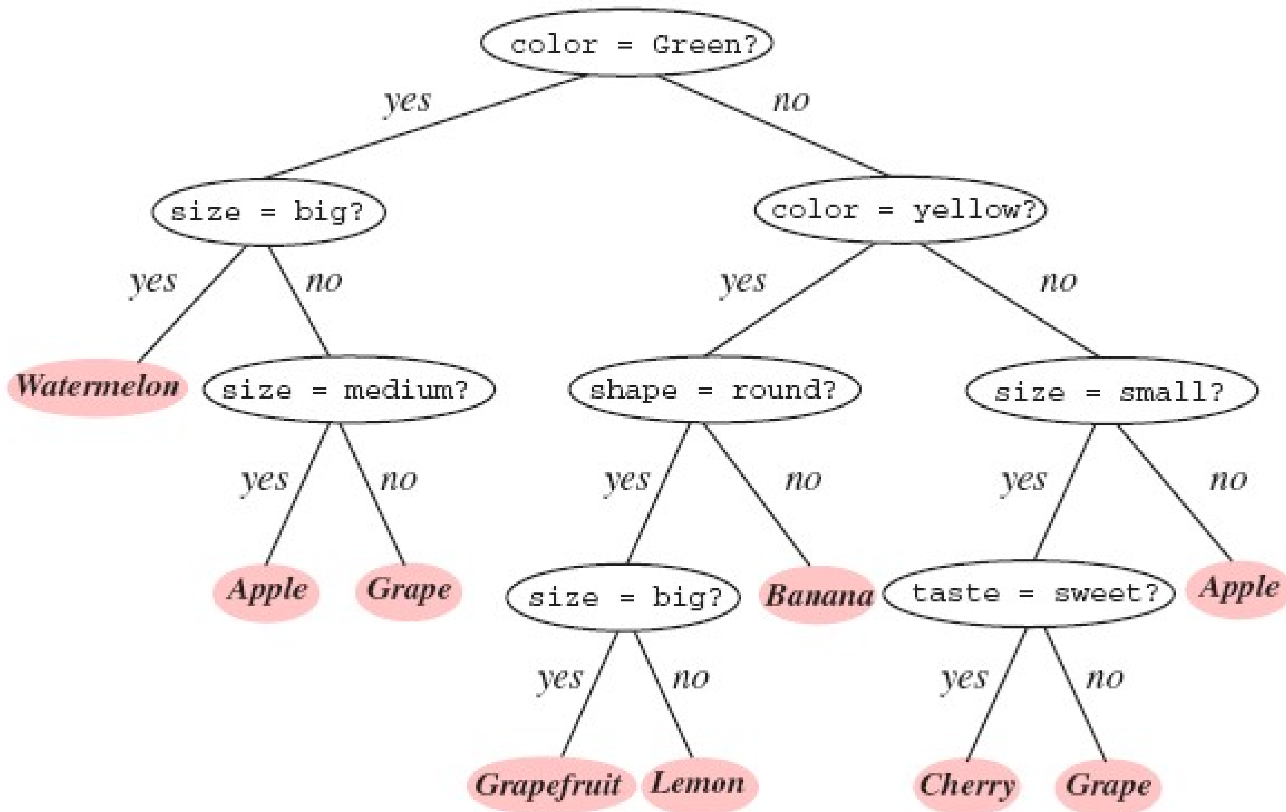
- Sequence of questions to classify a pattern
- Root node and successive branches linked to other nodes
- Links must be mutually distinct and exhaustive
- Questions finish at leaf nodes

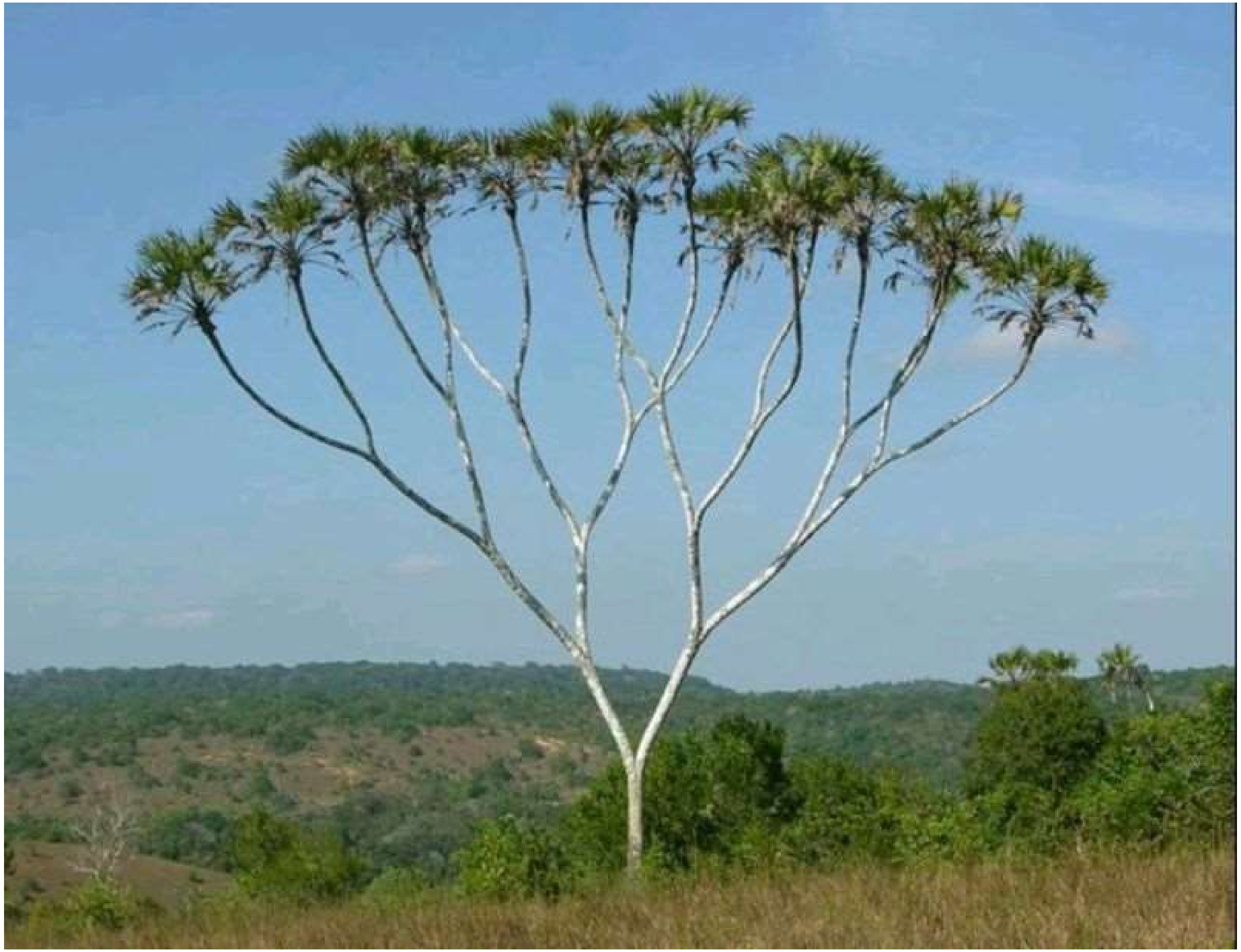


- Pattern of attributes

$$\mathbf{x} = \begin{bmatrix} <color> \\ <size> \\ <shape> \\ <taste> \end{bmatrix}$$

- A tree with arbitrary branching can be represented by an equivalent binary tree





Example 2: Attribute Feature Vector (AFV)

Patterns may not be so closely related to the classes

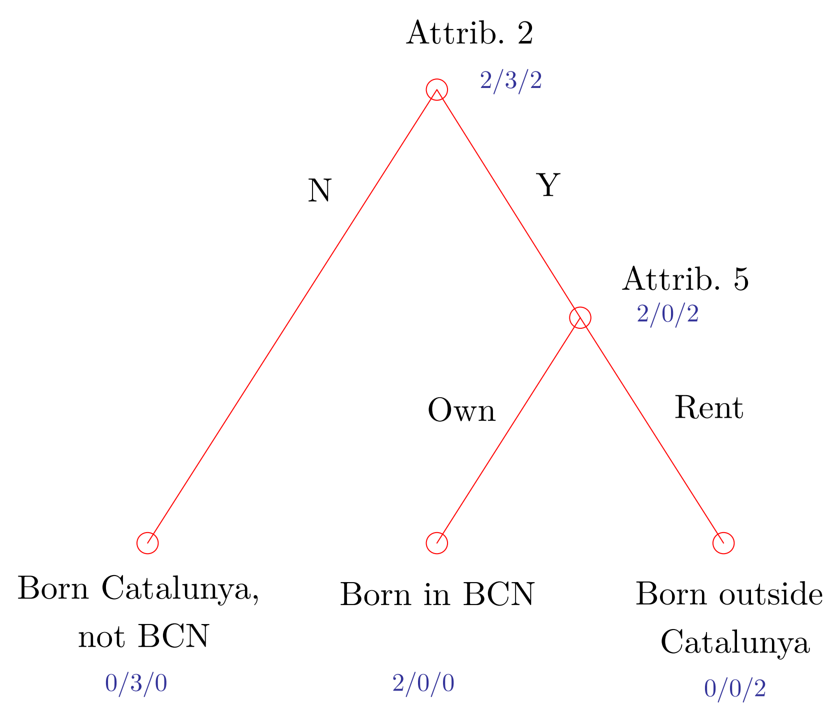
3 Classes:

- Class 1: Born in BCN
- Class 2: Born in Catalunya except BCN
- Class 3: Born outside Catalunya

Questions (associated to attributes)

1. Do you live in BCN from Monday to Friday?
2. Do you live in BCN in weekend?
3. Does your family (parents) live in BCN?
4. Are you registered in BCN?
5. Do you live in your own house/apartment or in a rented one?

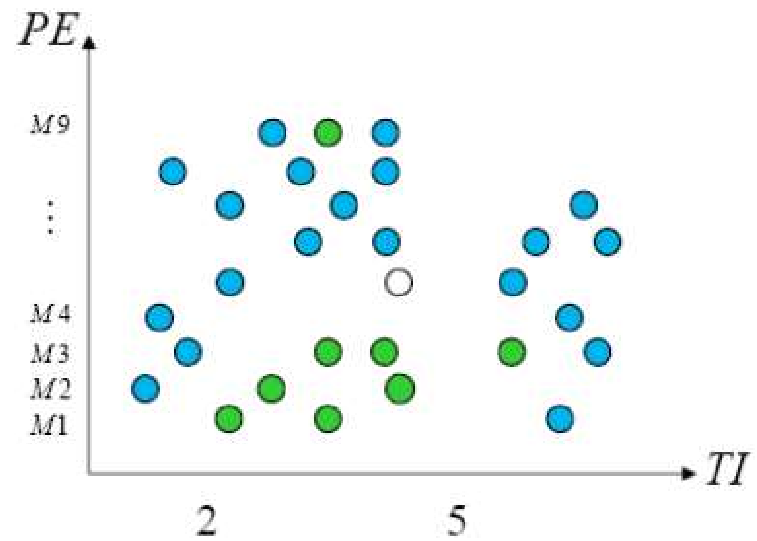
		Class						
		3	3	2	2	2	1	1
Attributes	BCN Mon-Fri	Y	Y	Y	N	Y	Y	Y
	BCN weekend	Y	Y	N	N	N	Y	Y
	Parents BCN	N	N	N	N	N	N	Y
	Registered in BCN	Y	N	N	N	N	Y	Y
	Apartment	Rent	Rent	Own	Own	Rent	Own	Own



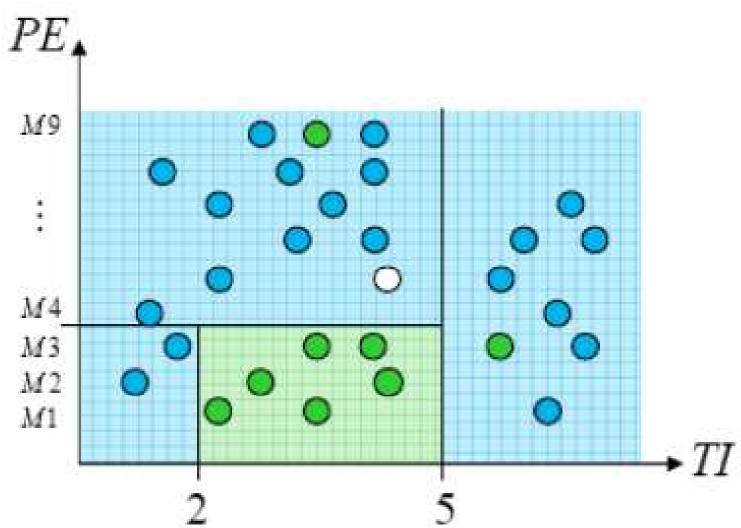
Example 3: Decision boundaries

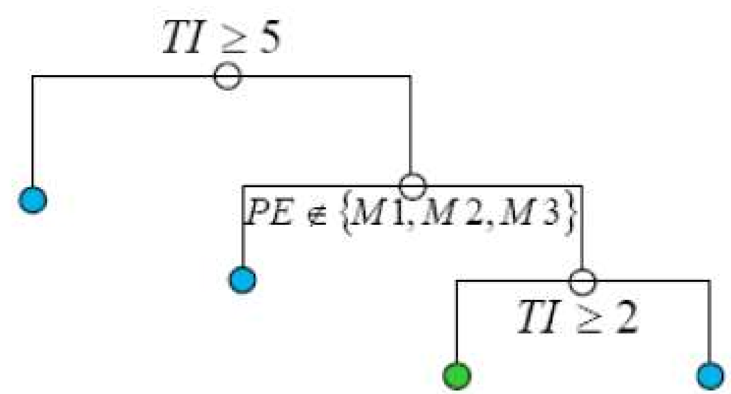
Data base

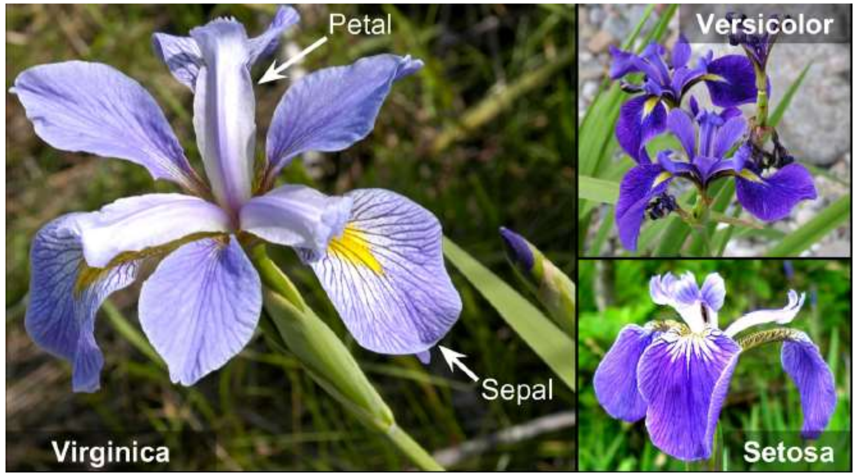
TI	*PE*	*Response*
1.0	*M2*	good
2.0	*M1*	bad
...	...	...
4.5	*M5*	?



Trained tree

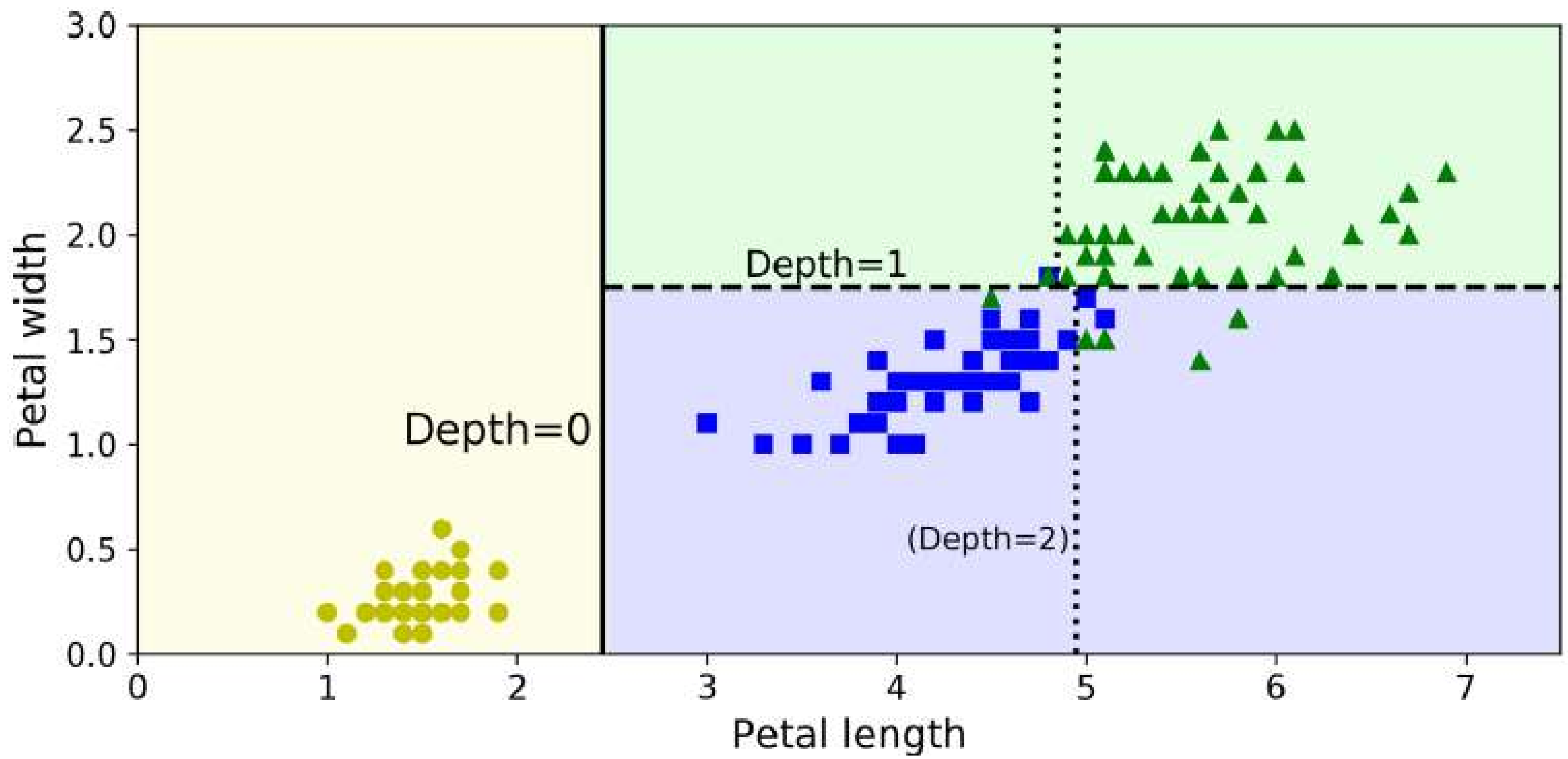




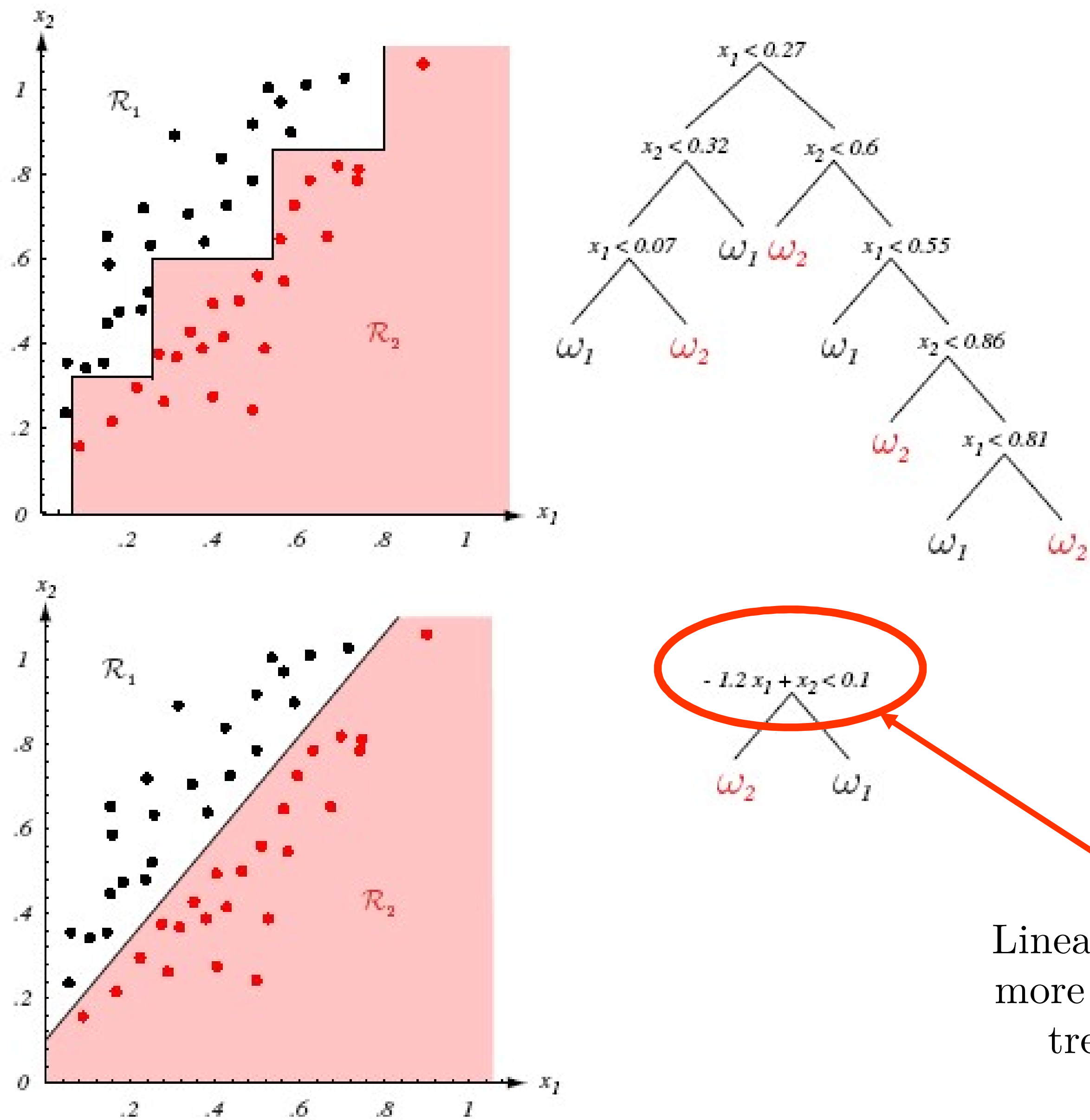


Decision boundaries are step-wise functions.

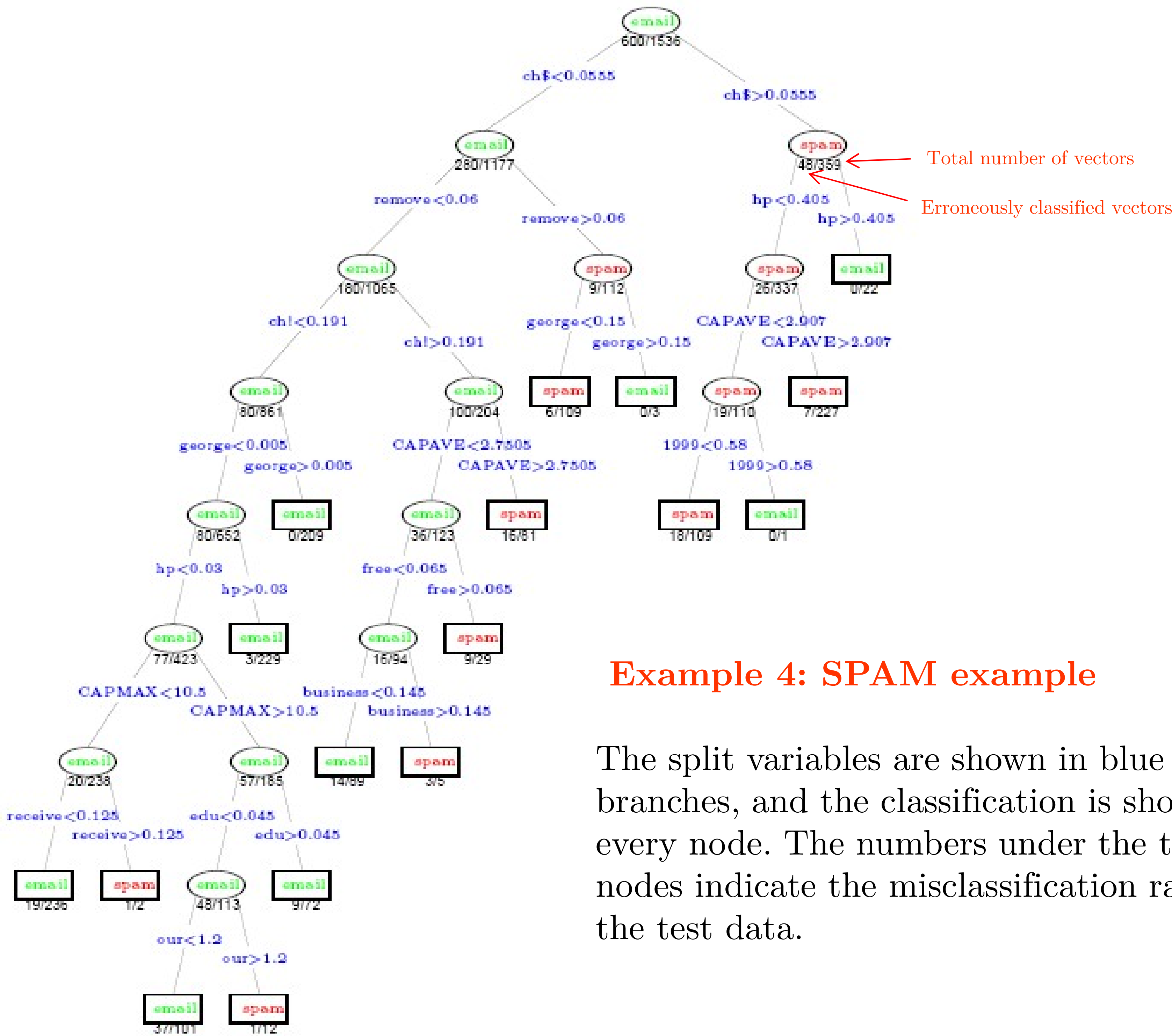
Decisions are easily explained: those features not considered in the queries of tree nodes, do not impact on the decisions (and viceversa).



Boundary shapes are given by the choice of features in queries...



Linear transform is more efficient than tree classifier



Example 4: SPAM example

The split variables are shown in blue in the branches, and the classification is shown in every node. The numbers under the terminal nodes indicate the misclassification rates on the test data.

2 CART: Classification And Regression Trees

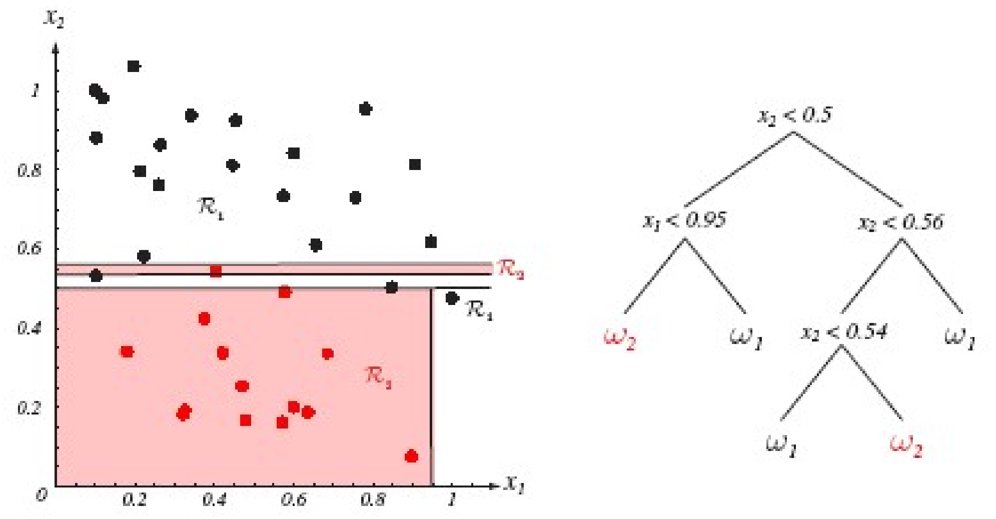
Binary decisions for RFV and AFV

- A Training Labeled Dataset is used to create a classification tree
- A decision tree progressively splits the set of training samples into smaller and smaller subsets
- When all the samples in a subset have the same category the branch of the tree is terminated
- A branch can be alternatively terminated with a mixture subset and declared leaf using CARTs
- A feature can be tested once or more with different thresholds (RFV)
- Objective: To obtain a small binary tree

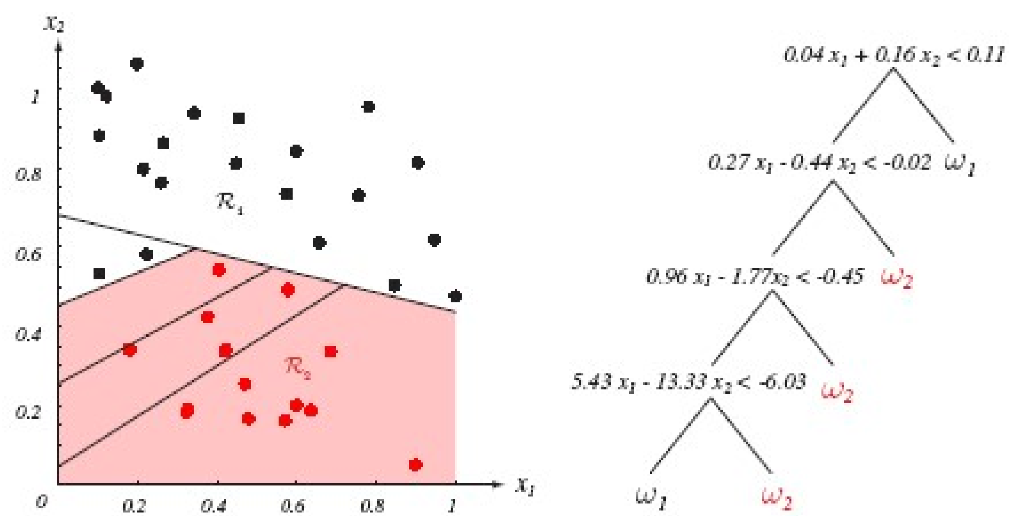
- Relevant questions when working with CARTS:
  1. Should the properties be binary-valued or multi-valued?
  2. Which property must be tested at each node?
  3. When a node should be declared a leaf?
  4. How to assign labels to leaf nodes?
  5. How to prune the tree if it becomes too large?

Question 1: Every decision can be represented using just binary decisions, so we will concentrate on binary trees with queries involving only one property (decision boundaries are perpendicular to coordinate axis)

E.g. queries of the type
`(size=medium) AND (NOT(color=yellow))?`
will not be found.



x_2
x_1
$\mathcal{R}_1$
$\mathcal{R}_2$
$x_2 < 0.5$
$x_1 < 0.95$
$x_2 < 0.56$
ω_2
ω_1
$x_2 < 0.54$
ω_1
ω_1
ω_2



x_2
x_1
$\mathcal{R}_1$
$\mathcal{R}_2$
$0.04\,x_1 + 0.16\,x_2 < 0.11$
$0.27\,x_1 - 0.44\,x_2 < -0.02$ ω_1
$0.96\,x_1 - 1.77x_2 < -0.45$ ω_2
$5.43\,x_1 - 13.33\,x_2 < -6.03$ ω_2
ω_1 ω_2

Query selection

Question 2. Property tested at each node?

We seek a property to be tested at each node that makes the immediate descendent node as pure as possible. Impurity is minimized.

- ENTROPY IMPURITY: 'infcrit'

$$i_E(N) = -\sum_j P(\omega_j)\log_2 P(\omega_j) \qquad P(\omega_j) = \frac{n_j}{N}$$

Fraction of feature vectors at a given node belonging to category ω_j

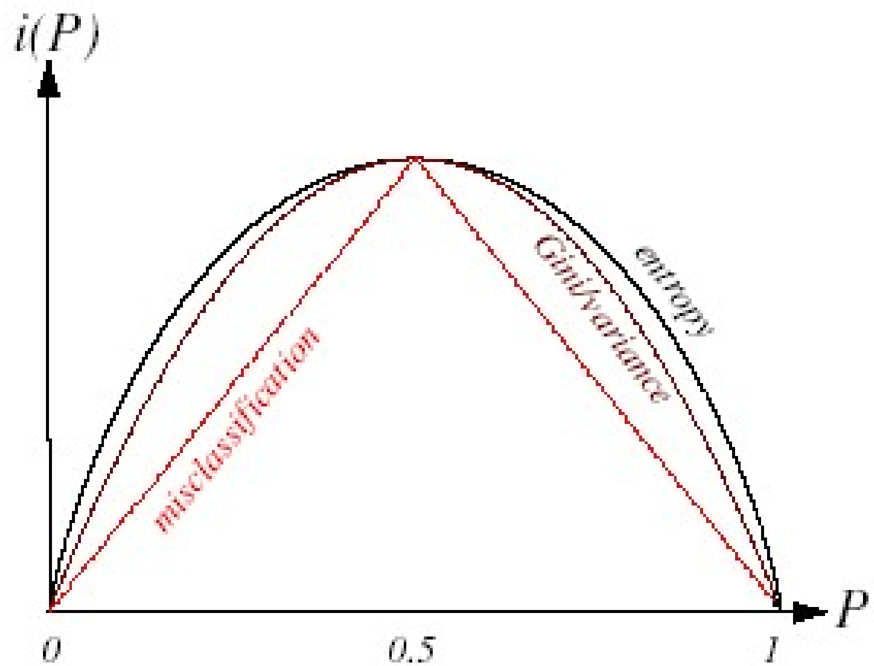
- GINI IMPURITY: 'maxcrit'

$$i_G(N) = \sum_i P(\omega_i)\left(1 - P(\omega_i)\right) = 1 - \sum_i P^2(\omega_i)$$

- MISSCLASSIFICATION IMPURITY:

$$i_M(N) = 1 - \max_j P(\omega_j)$$

- Entropy, Gini and Misclassification impurities for $c = 2$ classes.



BINARY TREES

Given a node N, what feature should be chosen for the test?
"$T > T_o$" for RFV or "*Is T equal to a label?*" for AFV

- **Criterion**: chose the one decreasing impurity as much as possible at node n (local optimization)

Number of vectors sent to the left node

Number of vectors sent to the right node

$$\Delta i(n) = i(n) - \frac{N_L}{N} i(n_L) - \frac{N_R}{N} i(n_R) < 1 \text{ bit}$$

$$N = N_L + N_R$$

- Sometimes several decisions imply same impurity variation. With real values, if the impurity measure does not vary in (x_L, x_R) the threshold is chosen as (RFV):

$$x_s = \frac{N_L}{N} x_L + \frac{N_R}{N} x_R$$

- Practical problem: If two different patterns have the same attributes and come from different categories, the impurity at leafs cannot be reduced to zero. Typical for AFV.

- The particular choice of an impurity function rarely affects the final structure for the tree.

When to stop splitting?

Question 3. When a node should be declared a leaf?

If the training set is very big, the obtained tree can be overfitted.
In extreme splitting, each leaf corresponds to a single training vector.

Different stopping criteria can be defined:

1. Threshold in impurity: The impurity loss of the best split must be higher than a threshold. This generates unbalanced trees if the complexity of the data varies throughout the range of input values.

2. Minimum Description Length (MDL): A criterion function is minimized:

$$\alpha \cdot \text{size} + \sum_{N \text{ leaf}} i(N)$$

   where size is the number of nodes and α is a tuning positive parameter that governs the tradeoff between tree size and the goodness to fit to the data. Large values of α result in smaller trees.

3. Validation techniques: A fraction of β% of the training data is used to test the error on the tree generated from the (1-β)% vectors set. The splitting process finishes when the error on the validation data is minimized. Use k-folding cross-validation!

4. Stop when a node presents less than γ% number of vectors.

Question 4. How to assign labels to leaf nodes?

It is the simplest step: assign the label of the category that has more vectors represented.

3 Pruning CART

Question 5. How to prune a tree if it is too large?

Stopped splitting lacks of sufficient look ahead, and stopping may be decided too early for good classification accuracy...

Pruning

- First: step the tree is grown fully or until some maximum size.
- Second: some sub-trees are substituted for leaves provided that the classification error does not increase "too much".
- The final tree may be unbalanced.

Pruning strategy 1 (Pessimistic or Quinlan Criteria PEP)

- It is a TOP-DOWN approach:
  - T_i are all possible (non-terminal) subtrees of T.
  - Subtrees T_i are built by replacing one or more subtrees by leaves.
  - $|T_i|$ is the number of leaves from subtree T_i.
  - Find the subtree T_α that minimize:

Errors at subtree T_i

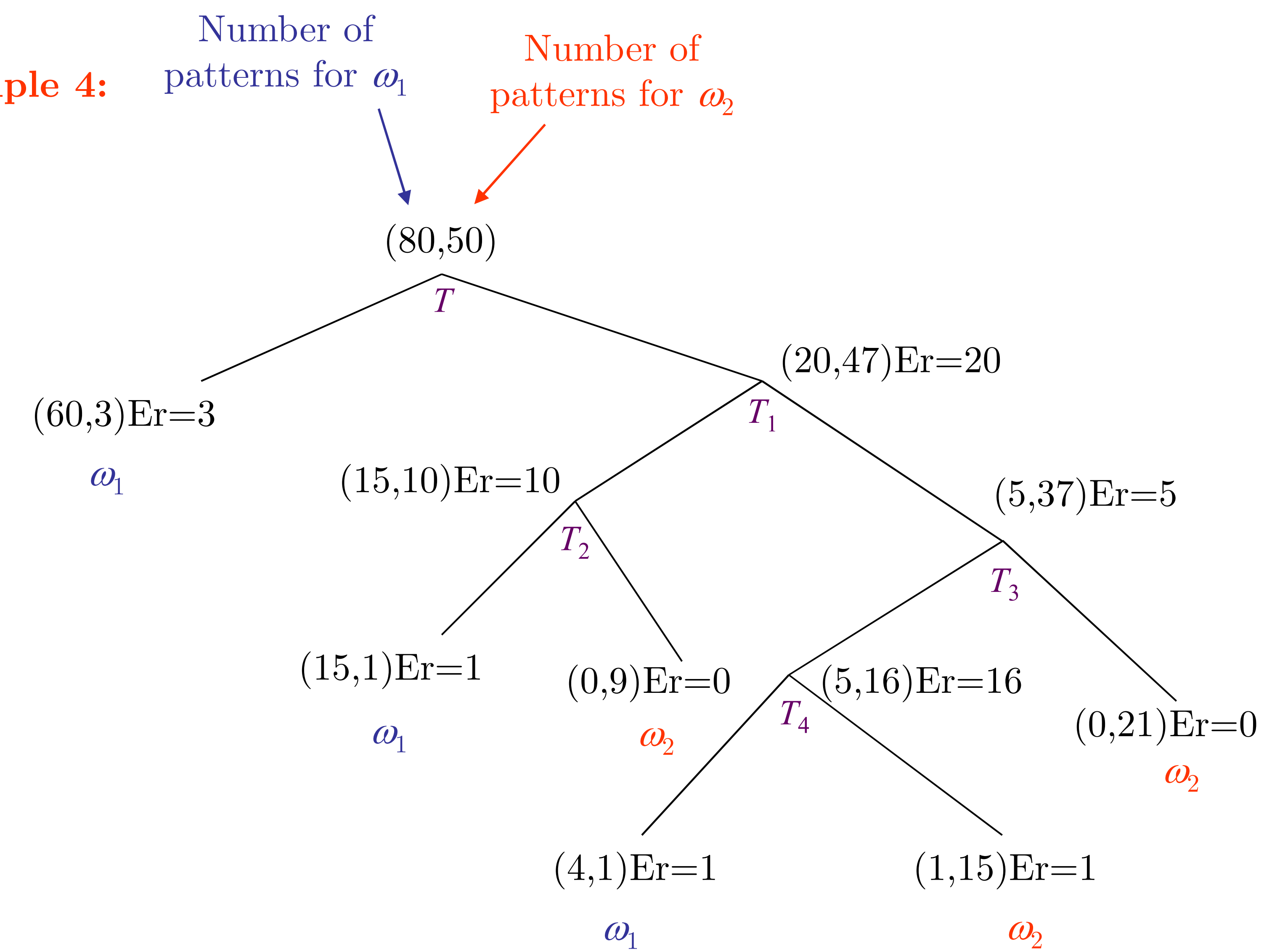
$$C(T_i) = \sum_{n=1}^{|T_i|} E_{i,n} + \alpha |T_i|$$

among all possible subtrees. Replace tree T by T_α. For a given value of α start with the node producing the minimum increase in error.

- Large α result in smaller trees. With $\alpha = 0$ the solution is keeping all the subtrees of T (length is not penalizing).
- The optimum value of α is obtained by cross-validation.

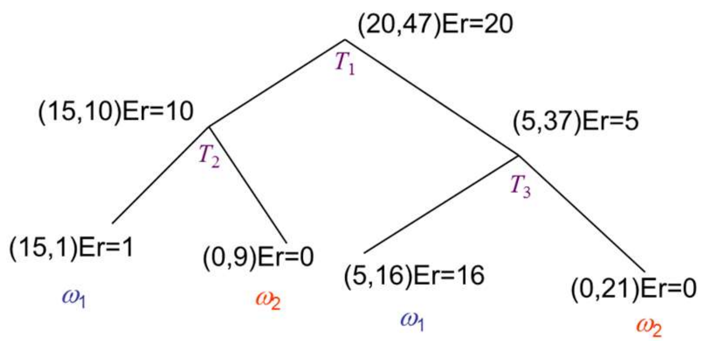
T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning*, Springer-Verlag, 2008, section 9.2.2.

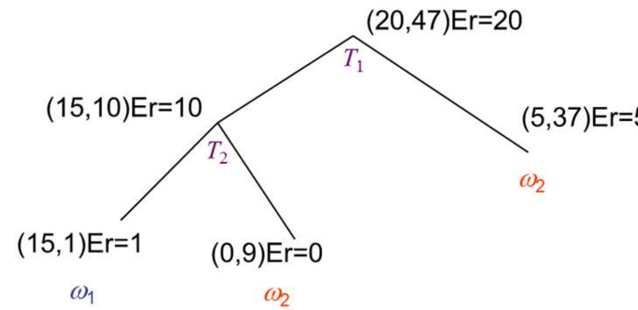
Example 4:

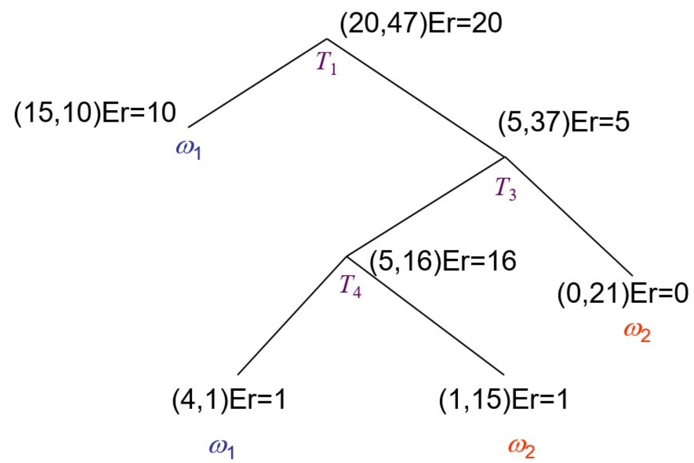


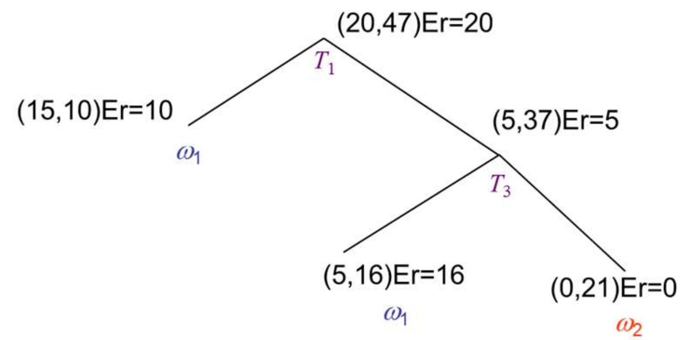
Exercise: Check which value of α implies that T_1 is replaced by a leaf.

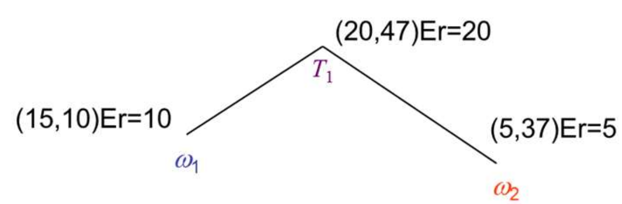
The 5 subtrees of T_1 in example 4...







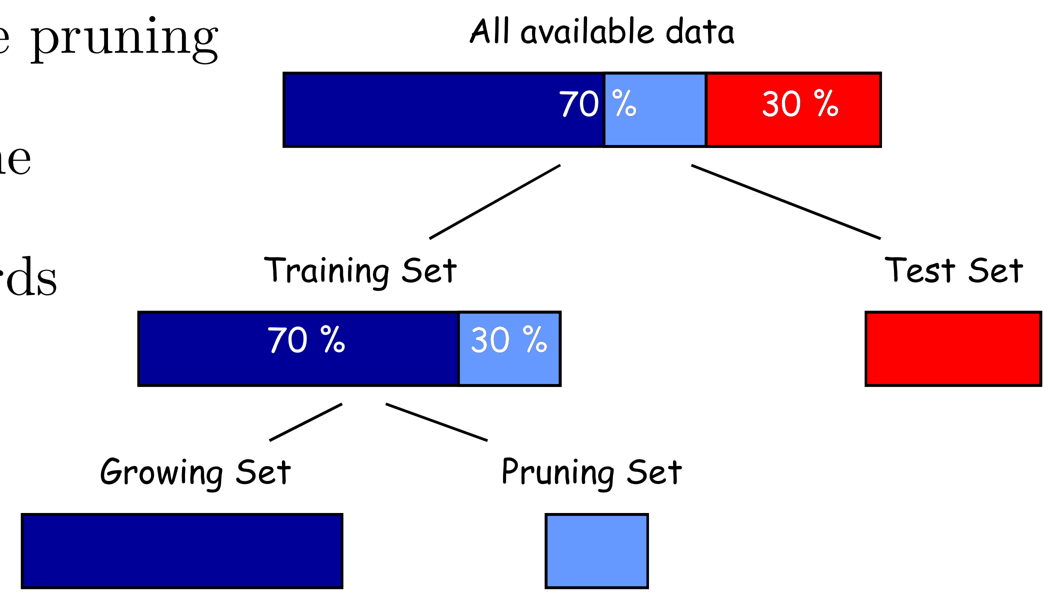




Pruning strategy 2 (Reduced Error Pruning REP)

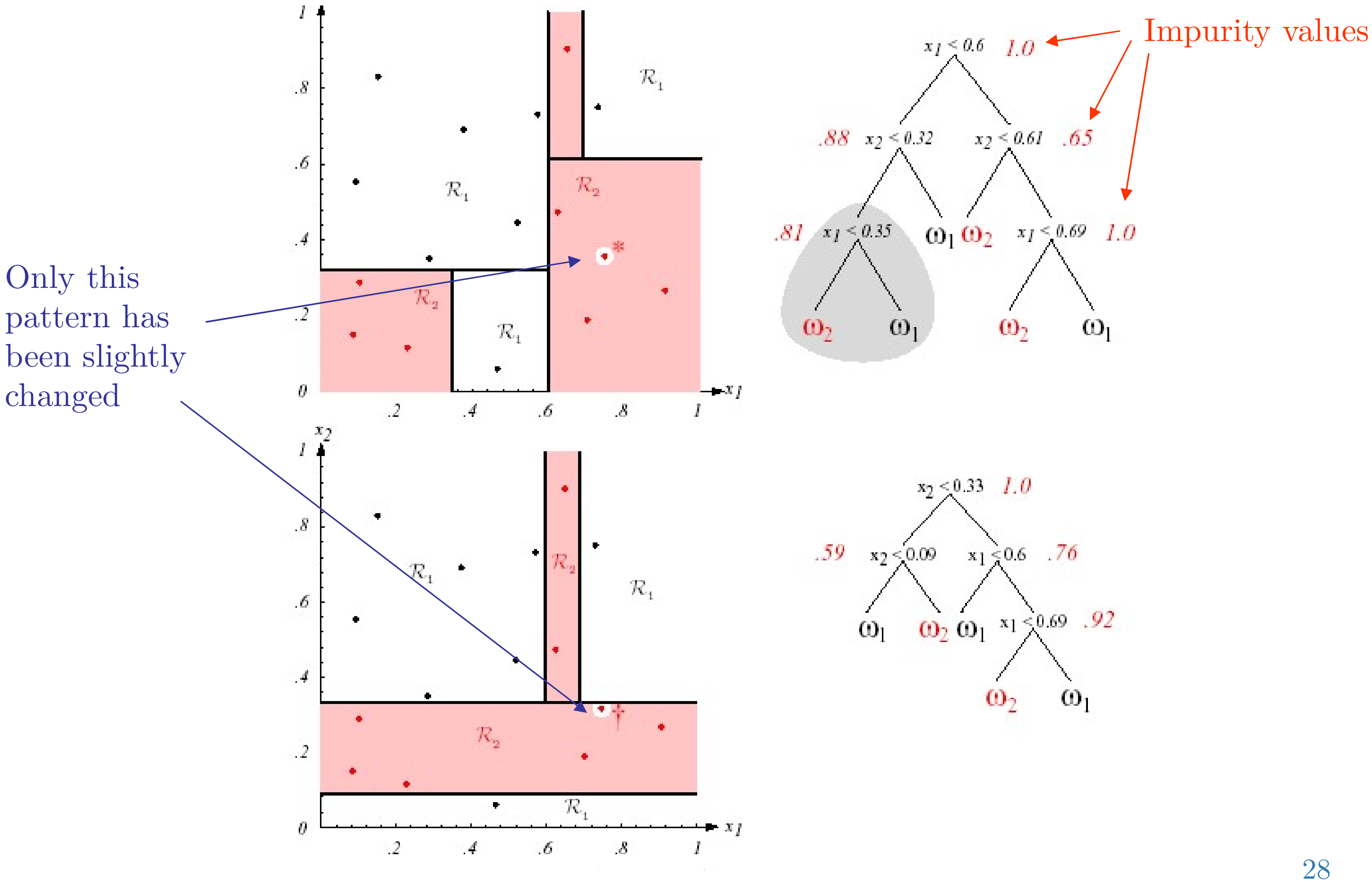
- It is a BOTTOM-UP approach:
  - It uses an additional validation set, called the “pruning set” unseen during the growing stage.
  - A simple error check is calculated for all *non-leaf* nodes.

- The node T_i is replaced by a leaf if the error is smaller than the error of the whole subtree using the pruning set.
  - The leaf is labeled to the majority class.
  - Danger: tendency towards *overpruning*.



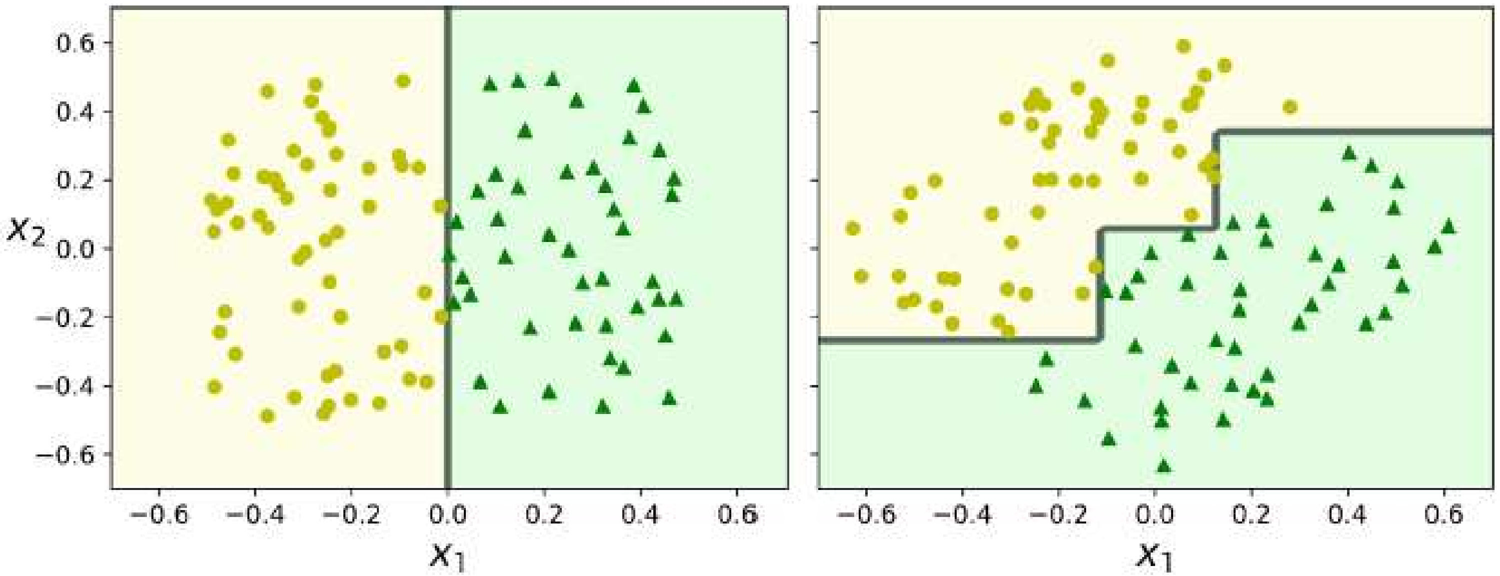
R. Kohavi, R. Quinlan, “Decision Tree Discovery”, 1999 http://ai.stanford.edu/~ronnyk/treesHB.pdf

4 Bias and variance of decision trees



Unstability

Decision trees are also sensitive to rotation of data.



The tree on the right is much more complex.

Shallow trees have less variance but higher bias and then will be better choice for sequential methods (boosting) that will be described in chapter 5.

Deep trees, on the other side, have low bias but high variance and, so, are relevant choices for bagging method that is mainly focused at reducing variance (chapter 5).

5 Missing attributes

When training:

Evaluation of impurities

- Assume vector $\boldsymbol{x}(n)$ has one of the attributes $x_j(n)$ missing. At each level of the tree, impurities for the splits involving all possible attributes are computed.
- When involving attribute x_j, use all vectors except $\boldsymbol{x}(n)$.

Surrogate splitting

- At node N, in addition to the primary split s_p involving feature x_k, each non-terminal node is given an ordered number of *surrogate splits* s_1, s_2,… involving attributes x_1, x_2,…, $x_{k\text{-}1}$, x_{k+1},…
- The first surrogate split s_1 maximizes the “predictive association” with the primary split s_p: the number of patterns sent to the left+right is the closest.

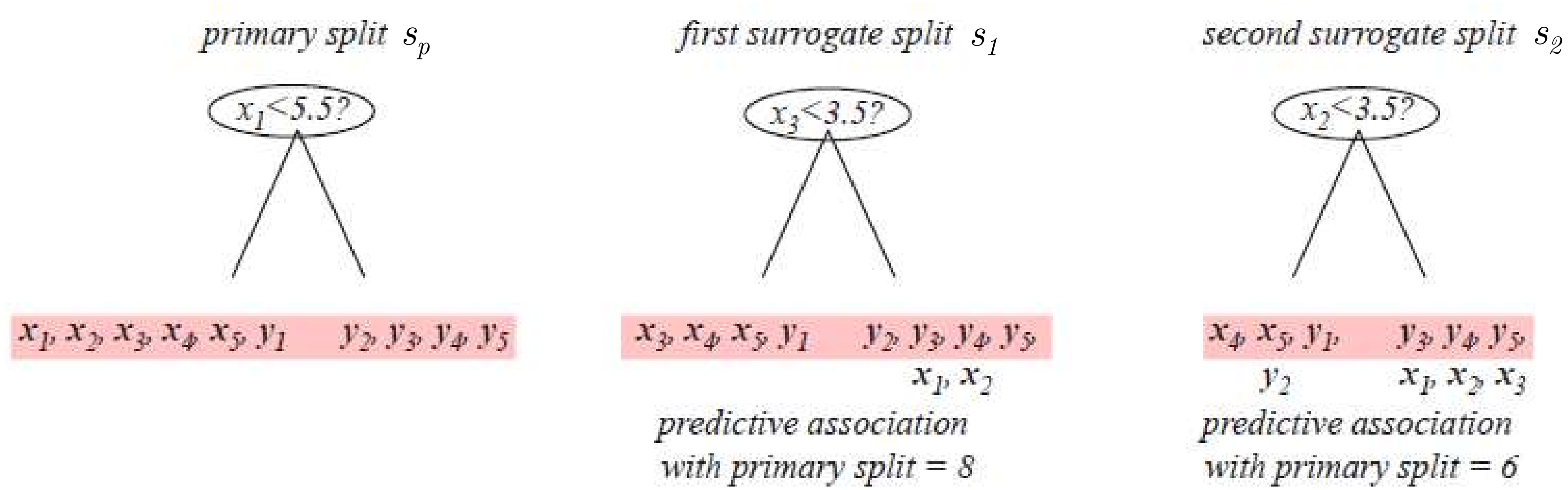
When classifying:

- When a pattern missing attribute x_k arrives at node N, the first *surrogate split* is used s_1 instead of s_p.

Example 6: Surrogate splits and missing attributes

$$\omega_1: \quad \begin{array}{ccccc} \mathbf{x}_1 & \mathbf{x}_2 & \mathbf{x}_3 & \mathbf{x}_4 & \mathbf{x}_5 \\ \begin{pmatrix} 0 \\ 7 \\ 8 \end{pmatrix}, & \begin{pmatrix} 1 \\ 8 \\ 9 \end{pmatrix}, & \begin{pmatrix} 2 \\ 9 \\ 0 \end{pmatrix}, & \begin{pmatrix} 4 \\ 1 \\ 1 \end{pmatrix}, & \begin{pmatrix} 5 \\ 2 \\ 2 \end{pmatrix} \end{array}$$

$$\omega_2: \quad \begin{array}{ccccc} \mathbf{y}_1 & \mathbf{y}_2 & \mathbf{y}_3 & \mathbf{y}_4 & \mathbf{y}_5 \\ \begin{pmatrix} 3 \\ 3 \\ 3 \end{pmatrix}, & \begin{pmatrix} 6 \\ 0 \\ 4 \end{pmatrix}, & \begin{pmatrix} 7 \\ 4 \\ 5 \end{pmatrix}, & \begin{pmatrix} 8 \\ 5 \\ 6 \end{pmatrix}, & \begin{pmatrix} 9 \\ 6 \\ 7 \end{pmatrix} \end{array}.$$



6 Other Methods

ID3: Interactive dichotomizer 3 (Non Binary)

- It is designed for **nominal** data.
- Discrete Feature Vectors (or discretized)
- Categories are treated as unordered
- Each node has as many children nodes as the number of categories of the (nominal) feature at that node.
- The maximum number of levels is equal to the number of features.
- The algorithm continues until all nodes are pure or there are no more variables to split on.

C4.5

- It is an extension of ID3 that accounts for unavailable values, continuous attribute value ranges, pruning of decision trees, rule derivation, and so on. Last Version C5.0 [Quinlan, 1993]
- It uses continuous valued variables as in CARTS and the nominal variables as in ID3.

7 Conclusions

- Entropy **impurity** measure is the most acceptable in most cases.
- **Pruning** (post-pruning) is preferred over stopped splitting (pre-pruning) but computationally worse.
- Large training set size can produce overfitted trees.

- Advantages of decision trees
  - Natural handling of mixed data types
  - Handling missing features
  - Robustness to outliers
  - Computational scalability (large data bases)
  - Insensitiveness to monotone transformation of inputs
  - Ability to deal with irrelevant features

- Limitations of decision trees
  - Poor bias-variance tradeoff
  - Low prediction accuracy: only practical if non-metric data, and/or combined with bagging, boosting or *random forest* classifiers (see next chapter)