Algorithms for Decision Tree Induction

Paulo Trigo Silva

The classification task

- To classify is to decide how to answer to the question:
  - "In which class (group) does this example better fits (belongs)?"
  - i.e., learn function $f: X \rightarrow \{ class\text{-}value_1, class\text{-}value_2, ..., class\text{-}value_n \}$
  - ... each $x \in X$ is an example we want to know the $class\text{-}value_i$ it better fits
- The input data
  - the set of class values to consider, e.g., $C = \{ cv_1, cv_2, ..., cv_n \}$
  - pre-classified examples, e.g., $<x_1, cv_3>$, $<x_2, cv_4>$, $<x_3, cv_3>$, $<x_4, cv_1>$, ...
- The output data
  - the function *f*, i.e., a "classification procedure",
  - ... the function *f* can be described in several formats,
  - e.g., statistical model, **decision tree**, set of rules, neural network, etc
  - ... *f* is used to associate a new object to its most likelihood class
  - ... to classify is to make such an association!

Paulo Trigo Silva

Classify – Induction (from data) Decision Trees

Approach:

"search for tree structures that classify the examples"

- ***Strategy***: *top down* "divide-and-conquer" recursion
- **[1]** select an attribute, *attr*, as the root node
  - create a descendent node (child) for each value of the attribute (*attr*)
- **[2]** split (separate) the dataset instances into subsets
  - for each descendent node, *r*, build a dataset subset, d_r
- **[3]** for the dataset of each node *r* (i.e., without *attr* and with data d_r)
  - repeat (recursively) from step **[1]**
- **[4]** terminate if all instances have the same class

Paulo Trigo Silva

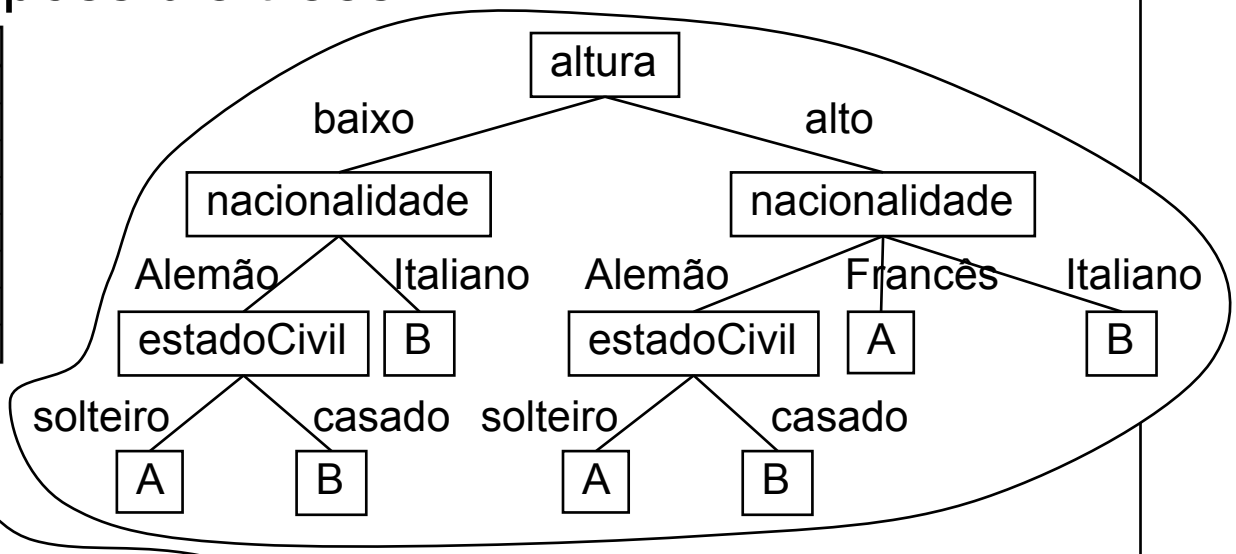
... an example (abstract)

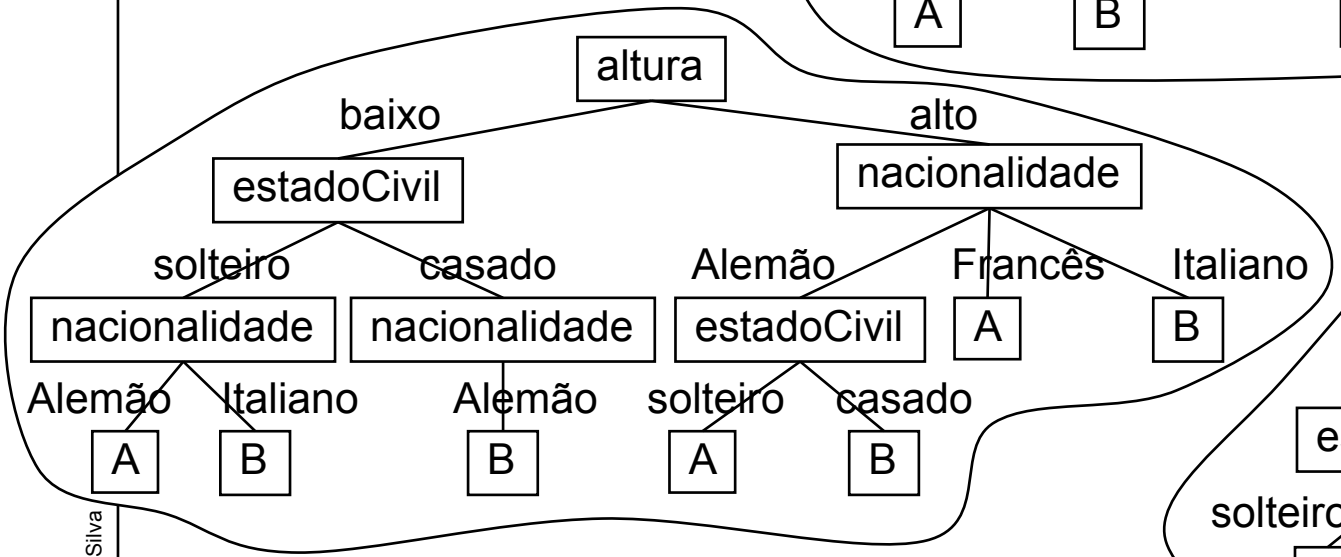
altura	nacionalidade	estadoCivil	classe
baixo	Alemão	solteiro	A
alto	Francês	solteiro	A
baixo	Italianao	solteiro	B
alto	Alemão	solteiro	A
alto	Alemão	casado	B
alto	Italiano	solteiro	B
alto	Italiano	casado	B
baixo	Alemão	casado	B

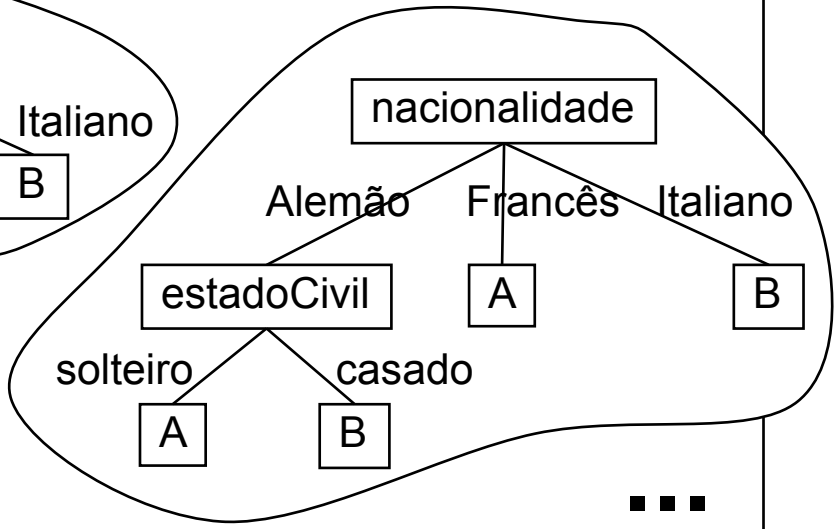
Build decision trees using the strategy previously described.

... example – some possible trees

altura	nacionalidade	estadoCivil	classe
baixo	Alemão	solteiro	A
alto	Francês	solteiro	A
baixo	Italianao	solteiro	B
alto	Alemão	solteiro	A
alto	Alemão	casado	B
alto	Italiano	solteiro	B
alto	Italiano	casado	B
baixo	Alemão	casado	B







Paulo Trigo Silva

The structure of a decision tree

- Decision Tree
  - a tree that we traverse to determine the class of an instance from the evaluation of conditions (tests) to the values of some of its attributes
- Decision Tree – evaluation of conditions (tests)
  - each test concerns the value of a single attribute
  - an internal node, *N*, rerepsents a test on the value of an attribute A
  - each arc, departing from node *N*, represents a possible value of A
  - each leaf node indicates a class value
- Decision Tree as a “classification procedure (or function)”
  - to classify an instance is to traverse the tree, starting at the root node
  - … branch traversal is guided by the values of the instance to classify

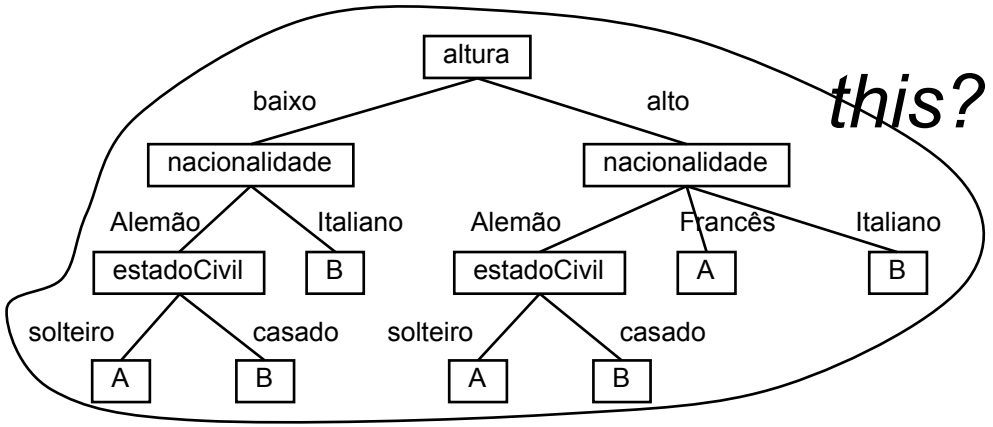
Paulo Trigo Silva

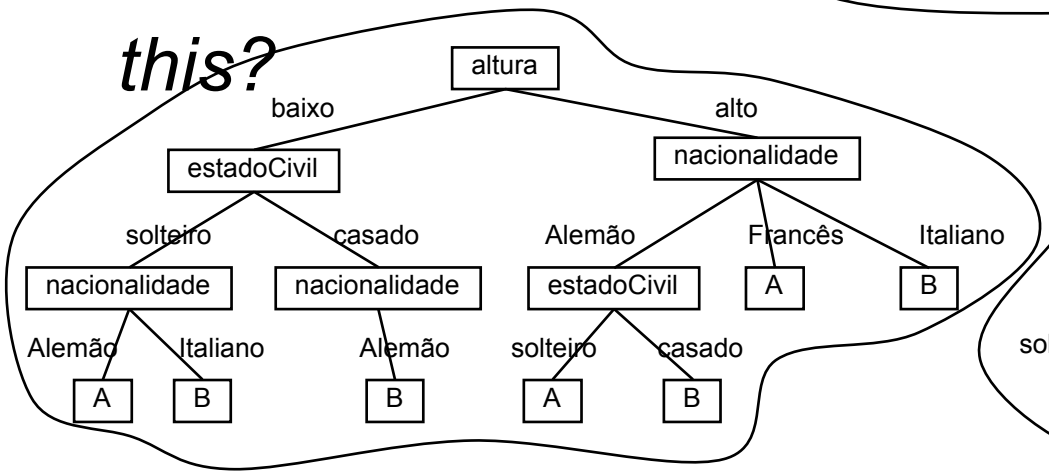
One dataset & Several (possible) Decision Trees

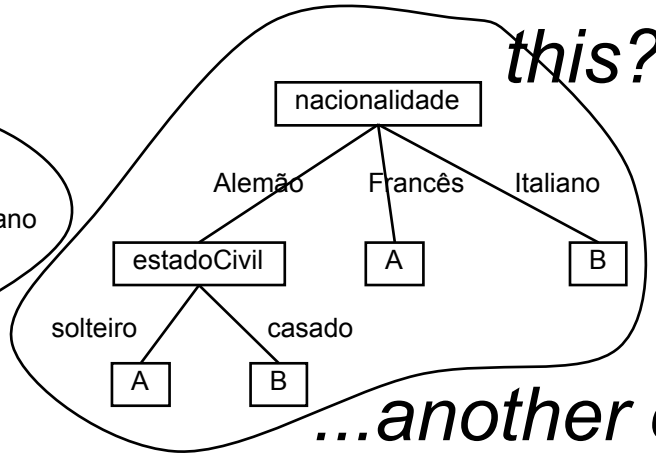
... but, which tree to choose?

(among the several trees that classify the examples)

altura	nacionalidade	estadoCivil	classe
baixo	Alemão	solteiro	A
alto	Francês	solteiro	A
baixo	Italianao	solteiro	B
alto	Alemão	solteiro	A
alto	Alemão	casado	B
alto	Italiano	solteiro	B
alto	Italiano	casado	B
baixo	Alemão	casado	B







Paulo Trigo Silva

... a strategy to choose an attribute

- ***Question***:
  - which is the “best” attribute (i.e., which to choose in each step)?
- ***Heuristic***:
  - choose the attribute that produces the “most pure sets”
  - ... within a pure set all the instances belong to the same class
  - choose the attribute that "best” discriminates among class values
  - ... which is the “best”? one that produces “smallest” (lowest depth) tree
- (usual) “impurity” criterion of a node: information gain
  - information gain increases with the average “purity” of a (sub)set
  - … the more a set is “pure” the higher is the information gain value
- ***Goal***:
  - choose the attribute that provides the greatest of information gain

Paulo Trigo Silva

Attribute Selection – motivation

- "select attribute that minimizes the uncertainty of the class value"
- ***attributes***:
  - pele ∈ { clara, morena }
  - cabelo ∈ { preto, ruivo, louro }
  - cremeSolar ∈ { sim, não }
- ***class***:
  - queimaduraSolar ∈ { +, – }

pele	cabelo	cremeSolar	queimaduraSolar
clara	louro	não	+
morena	louro	sim	-
morena	ruivo	não	+
clara	preto	não	-
morena	preto	não	-
morena	louro	não	+
morena	preto	sim	-
clara	louro	sim	-

8 examples distributed between two classes:

[3+, 5-]

Build the 3 possible root nodes

Paulo Trigo Silva

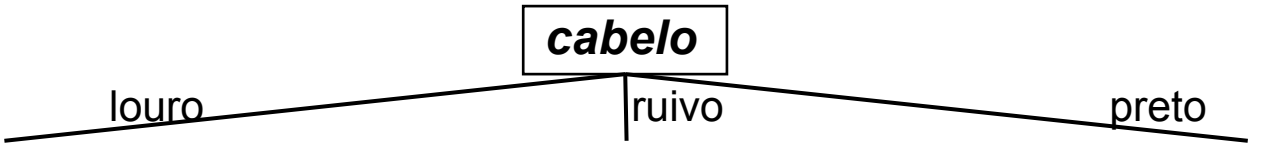
... the 3 root nodes and class distribution (for each node)

pele	cabelo	cremeSolar	queimaduraSolar
clara	louro	não	+
clara	preto	não	-
clara	louro	sim	-

[1+, 2-]

pele	cabelo	cremeSolar	queimaduraSolar
morena	louro	sim	-
morena	ruivo	não	+
morena	preto	não	-
morena	louro	não	+
morena	preto	sim	-

[2+, 3-]



pele	*cabelo*	cremeSolar	queimaduraSolar
clara	louro	não	+
morena	louro	sim	-
morena	louro	não	+
clara	louro	sim	-

[2+, 2-]

pele	*cabelo*	cremeSolar	queimaduraSolar
morena	ruivo	não	+

[1+, 0-]

pele	*cabelo*	cremeSolar	queimaduraSolar
clara	preto	não	-
morena	preto	não	-
morena	preto	sim	-

[0+, 3-]

pele	cabelo	*cremeSolar*	queimaduraSolar
clara	louro	não	+
morena	ruivo	não	+
clara	preto	não	-
morena	preto	não	-
morena	louro	não	+

[3+, 2-]

pele	cabelo	*cremeSolar*	queimaduraSolar
morena	louro	sim	-
morena	preto	sim	-
clara	louro	sim	-

[0+, 3-]

The decision tree as an information source

- The decision tree may be regarded as an “information source”
  - such that, given an instance, the tree generates a message
  - … with the class value for that instance
- ... the complexity of the decision tree is highly related with
  - the amount of information transmitted in each message it generates

- What amount of information is needed to identify each message?
  - let M be the universe of possible messages: M = { m_1, m_2, ..., m_n }
  - if all messages have the same probability, p, then p = 1/n
  - the information in the message can be coded using: $\log_2$ (1/n) bits
  - ... $\log_2(p) = \log_2(1/n) \Leftrightarrow \log_2(p) = \log_2(1) - \log_2(n) \Leftrightarrow$ **$-\log_2(p) = \log_2(n)$**
- i.e., if we have 16 messages, then $\log_2(16) = 4$, and therefore,
  - we need 4 bits to identify (distinguish) all possible messages

Paulo Trigo Silva

… example – “bits” to encode a message

You are watching a set of independent random samples of messages M

You see that M has four possible values (A, B, C, D) all with same probability

P(M = ‘A’) = 1/**4** | P(M = ‘B’) = 1/**4** | P(M = ‘C’) = 1/**4** | P(M = ‘D’) = 1/**4**

You transmit data over a binary serial link

You can encode each message with two bits:

e.g., A=00, B=01, C=10, D=11

So you want to transmit the information:

BAACBADC

just use the 2-bit encoding and send the information as:

0100001001001110

Paulo Trigo Silva

… example – “fewer bits” to encode a message

You are watching a set of independent random samples of messages M

Now, someone tells you that the <u>probabilities are not equal</u>

P(M = ‘A’) = 1/**2** | P(M = ‘B’) = 1/**4** | P(M = ‘C’) = 1/**8** | P(M = ‘D’) = 1/**8**

So you want to transmit the information: BAACBADC

Now, given that probability distribution it is possible…

… to invent a coding for your transmission

that only uses 1.75 bits, on average, per message (symbol)

How? “invent” such a coding

Paulo Trigo Silva

… example – “fewer bits” to encode a message (cont.)

You are watching a set of independent random samples of messages M

Now, someone tells you that the <u>probabilities are not equal</u>

P(M = ‘A’) = 1/**2** | P(M = ‘B’) = 1/**4** | P(M = ‘C’) = 1/**8** | P(M = ‘D’) = 1/**8**

invent a coding for your transmission

that only uses 1.75 bits, on average, per message (symbol)

e.g., A=0, B=10, C=110, D=111

(this is just one of several ways)

so, with this encoding, for BAACBADC transmit 1000110100111110

average-number-of-bits =

$= 1/2 * \log_2 2 + 1/4 * \log_2 4 + 1/8 * \log_2 8 + 1/8 * \log_2 8$

$= 1/2 * 1 + 1/4 * 2 + 1/8 * 3 + 1/8 * 3 =$ **1.75**

So, how much (expected) information (bits) in a message?

- For the general case,
  - the universe of messages is given by $M = \{ m_1, m_2, ..., m_n \}$
  - with a probability distribution $P = \{ p_1, p_2, ..., p_n \}$
  - ... where p_i represents the probability of m_i (and $\Sigma_{i=1..n}\ p_i = 1$)
- Hence, the information (Info) needed to identify m_i depends on p_i
  - i.e., $\mathbf{Info(\ m_i\ )\ =\ -\log_2\ p_i}$
- The expected information **of M** is given by,
  - $IE(\ \mathbf{M}\ )\ =\ \Sigma_{i=1..n}\ p_i\ Info(\ m_i\ )$
- Or, equivalently, the entropy, or uncertainty, **of P** is given by,
  - $IE(\ M\ )\ =\ entropy(\ \mathbf{P}\ )\ =\ \Sigma_{i=1..n}\ p_i\ Info(\ m_i\ )\ =\ \mathbf{\Sigma_{i=1..n}\ -p_i\ \log_2\ p_i}$

Paulo Trigo Silva

example – intuition on meaning of high, or low, entropy

- Given a X variable draw the histogram of its frequency distribution
  - what is the “shape” of the histogram if X has “**high** entropy”?
  - what is the “shape” of the histogram if X has “**low** entropy”?

variable V1 and its frequency distribution

freq.	2	2	2	2	2
value	A	B	C	D	E

variable V2 and its frequency distribution

freq.	1	6	1	1	1
value	A	B	C	D	E

Which variable, V1 or V2, exhibits the lowest entropy?

Intuition on the meaning of high, or low, entropy

- Given a X variable draw the histogram of its frequency distribution
  - what is the "shape" of the histogram if X has "**high** entropy"?
  - what is the "shape" of the histogram if X has "**low** entropy"?

- "**high** entropy" means that X is from a **uniform** (boring!) distribution
  - a histogram of the frequency distribution of values of X would be **flat**
- "**low** entropy" means that X is from a varied distribution
  - … from a distribution with **peaks and valleys**
  - a histogram of the frequency distribution of values of X would have **many lows and one or two highs**

Paulo Trigo Silva

… entropy example – in the “nature”!

lower entropy

… the values sampled from the location of water (or the location of grass) are entirely contained within the sea (or the soil)

higher entropy

… the values sampled from the location of water (or grass) are almost uniformed distributed throughout the whole space

Paulo Trigo Silva

Example (attribute *pele* – entropy from each value)

pele	cabelo	cremeSolar	queimaduraSolar
clara	louro	não	+
clara	preto	não	-
clara	louro	sim	-

[1+, 2-]

pele	cabelo	cremeSolar	queimaduraSolar
morena	louro	sim	-
morena	ruivo	não	+
morena	preto	não	-
morena	louro	não	+
morena	preto	sim	-

[2+, 3-]

$$\mathbf{IE(M) = entropy(P) = \Sigma_{i=1..n} - p_i \log_2 p_i}$$

pele = clara:

$IE([1; 2]) = entropy(1/3; 2/3) = -1/3 \log_2(1/3) - 2/3 \log_2(2/3) = 0{,}918$

pele = morena:

$IE([2; 3]) = entropy(2/5; 3/5) = -2/5 \log_2(2/5) - 3/5 \log_2(3/5) = 0{,}971$

Notice that for a class with 2 values the entropy range is [0 .. 1] *(to be detailed later)*

Example (attribute *pele* – entropy from the attribute)

pele	cabelo	cremeSolar	queimaduraSolar
clara	louro	não	+
clara	preto	não	-
clara	louro	sim	-

[1+, 2-]

pele	cabelo	cremeSolar	queimaduraSolar
morena	louro	sim	-
morena	ruivo	não	+
morena	preto	não	-
morena	louro	não	+
morena	preto	sim	-

[2+, 3-]

pele = clara: IE([1; 2]) = 0,918

pele = morena: IE([2; 3]) = 0,971

Now, compute the average of the entropy from each value of the attribute.

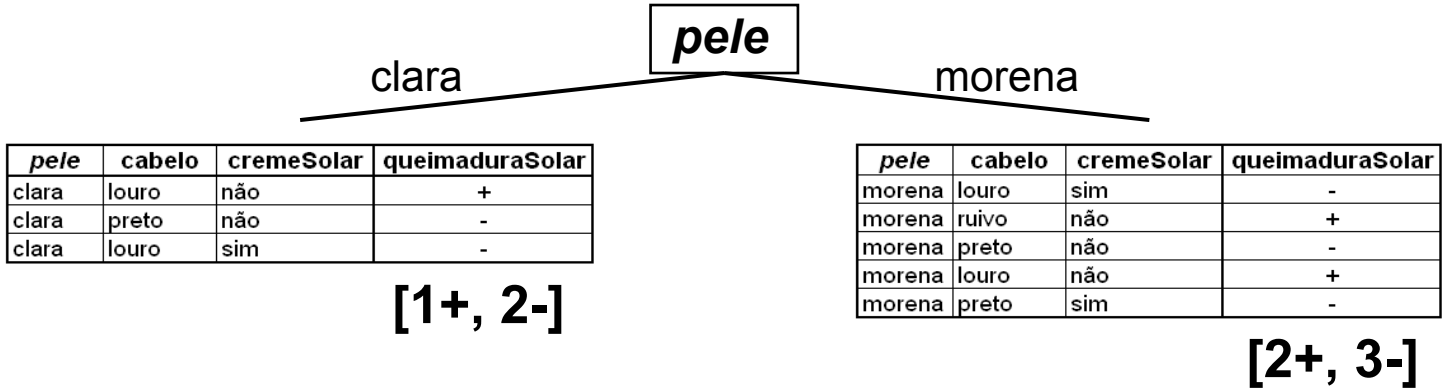
The average represents the expected amount of information that this tree needs "to generate a message" to specify the class for a new instance.

pela = clara v ***pele = morena*** (expected information for the **attribute**):

IE([1; 2], [2; 3]) = 3/8 IE([1; 2]) + 5/8 IE([2; 3]) = ***0,951***

Paulo Trigo Silva

Example with all the calculations (attribute *pele*)



pele	cabelo	cremeSolar	queimaduraSolar
clara	louro	não	+
clara	preto	não	-
clara	louro	sim	-

[1+, 2-]

pele	cabelo	cremeSolar	queimaduraSolar
morena	louro	sim	-
morena	ruivo	não	+
morena	preto	não	-
morena	louro	não	+
morena	preto	sim	-

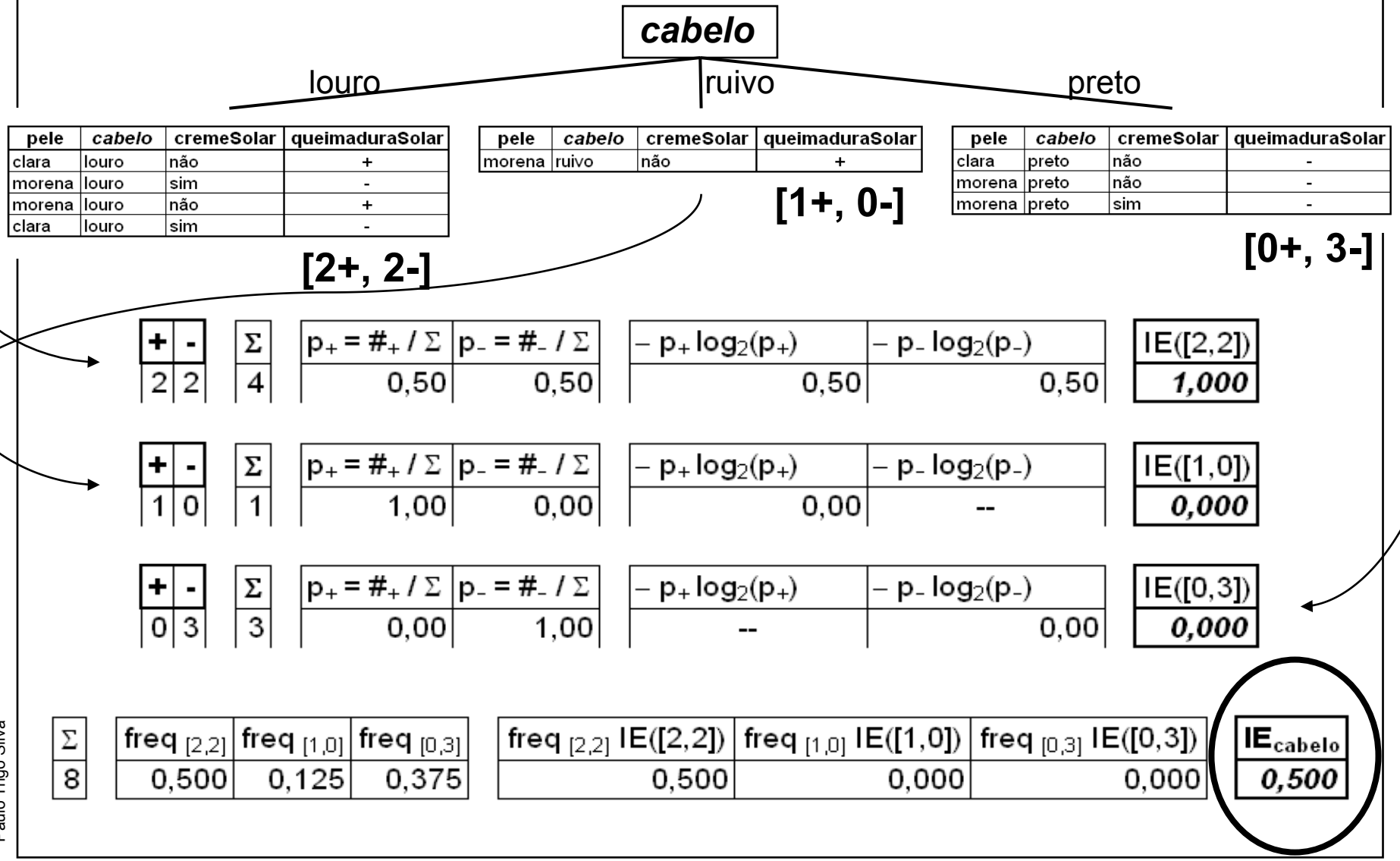
[2+, 3-]

+	-	Σ	$p_+ = \#_+ / \Sigma$	$p_- = \#_- / \Sigma$	$-p_+ \log_2(p_+)$	$-p_- \log_2(p_-)$	IE([1,2])
1	2	3	0,333	0,667	0,528	0,390	***0,918***

+	-	Σ	$p_+ = \#_+ / \Sigma$	$p_- = \#_- / \Sigma$	$-p_+ \log_2(p_+)$	$-p_- \log_2(p_-)$	IE([2,3])
2	3	5	0,400	0,600	0,529	0,442	***0,971***

Σ	$freq_{[1,2]}$	$freq_{[2,3]}$	$freq_{[1,2]}$ IE([1,2])	$freq_{[2,3]}$ IE([2,3])	IE_{pele}
8	0,375	0,625	0,344	0,607	***0,951***

Example with all the calculations (attribute *cabelo*)



pele	*cabelo*	cremeSolar	queimaduraSolar
clara	louro	não	+
morena	louro	sim	-
morena	louro	não	+
clara	louro	sim	-

pele	*cabelo*	cremeSolar	queimaduraSolar
morena	ruivo	não	+

pele	*cabelo*	cremeSolar	queimaduraSolar
clara	preto	não	-
morena	preto	não	-
morena	preto	sim	-

+	-	Σ	$p_+ = \#_+ / \Sigma$	$p_- = \#_- / \Sigma$	$-p_+ \log_2(p_+)$	$-p_- \log_2(p_-)$	IE([2,2])
2	2	4	0,50	0,50	0,50	0,50	*1,000*

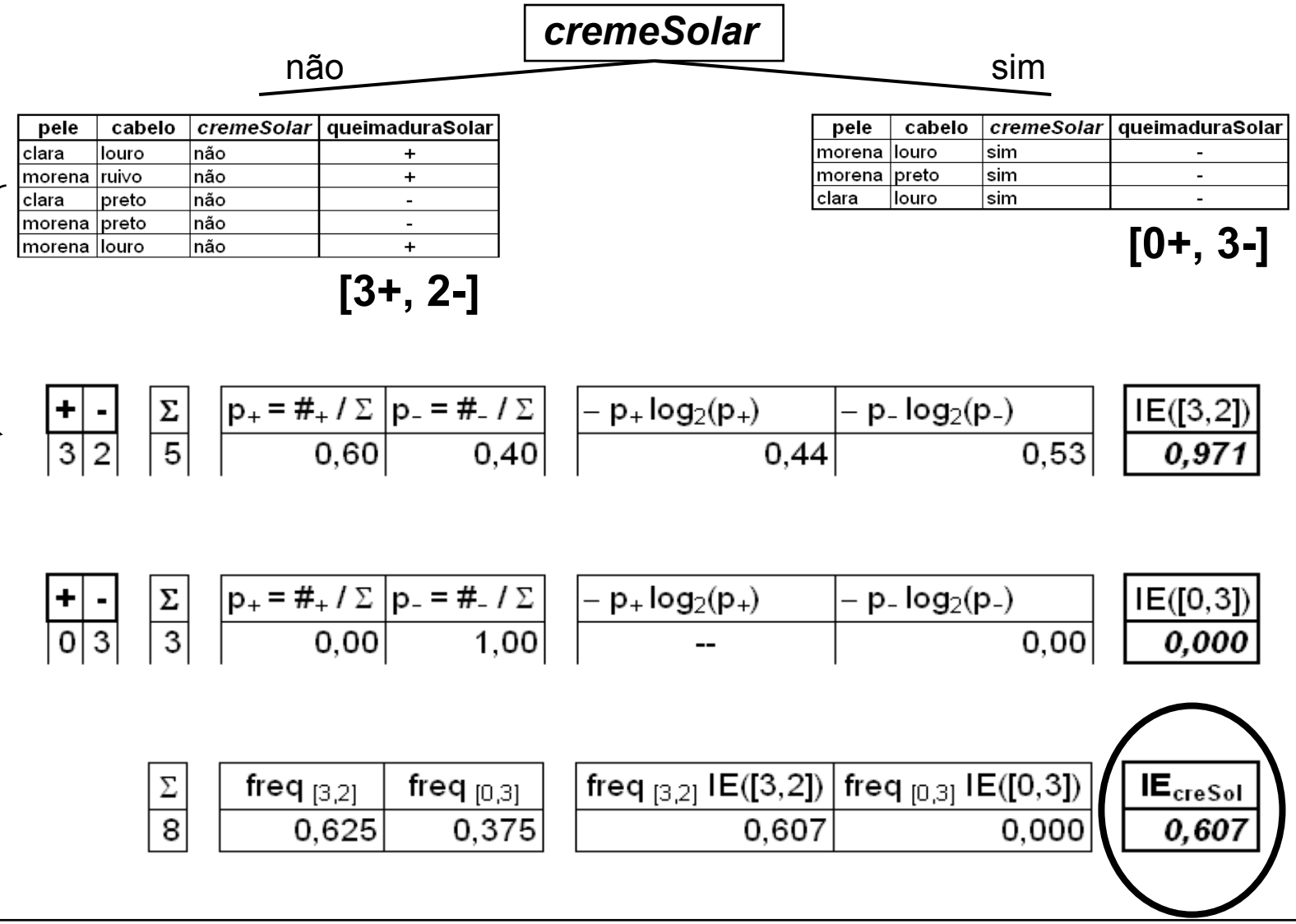
+	-	Σ	$p_+ = \#_+ / \Sigma$	$p_- = \#_- / \Sigma$	$-p_+ \log_2(p_+)$	$-p_- \log_2(p_-)$	IE([1,0])
1	0	1	1,00	0,00	0,00	--	*0,000*

+	-	Σ	$p_+ = \#_+ / \Sigma$	$p_- = \#_- / \Sigma$	$-p_+ \log_2(p_+)$	$-p_- \log_2(p_-)$	IE([0,3])
0	3	3	0,00	1,00	--	0,00	*0,000*

Σ	freq [2,2]	freq [1,0]	freq [0,3]	freq [2,2] IE([2,2])	freq [1,0] IE([1,0])	freq [0,3] IE([0,3])	IE_{cabelo}
8	0,500	0,125	0,375	0,500	0,000	0,000	*0,500*

Paulo Trigo Silva

Example with all the calculations (attribute *cremeSolar*)



... and now, which attribute to choose?

Heuristic: the one with the **highest** information gain.

Information Gain, g(C, A), the informational value of creating a branch for attribute A given a set of training instances C.

Given the attribute A with the possible values: $a_1, a_2, ..., a_m$

The choice of A originates the instances' partition C: $\{ C_{a1}, ..., C_{am} \}$

$$g(C, A) = IE(C) - \Sigma_{i=1..m} p(A = a_i) IE(C_{ai})$$

Recall: in C we have 8 examples distributed by 2 classes:

[3+; 5-]

$IE(C) = IE([3;5]) = entropy(3/8, 5/8) = - 3/8 \log_2(3/8) - 5/8 \log_2(5/8) =$ **0.954**

IE([3,5]) = 0,954	–	IE_{pele} = 0,951	
IE([3,5]) = 0,954	–	IE_{cabelo} = 0,500	we named this term as: IE_{ai}
IE([3,5]) = 0,954	–	IE_{creSol} = 0,607	

... and now, which attribute to choose? (cont.)

Given the attribute A with the possible values: a_1, a_2, ..., a_m

The choice of A originates the instances' partition C: { C_{a1}, ..., C_{am} }

$$g(C, A) = IE(C) - \Sigma_{i=1..m}\, p(A = a_i)\, IE(C_{ai})$$

Note: as the IE(C) is a constant value, the maximum gain is given by the attribute, A, with minimum average entropy (IE_A)

IE([3,5])		IE_{pele}
0,954	–	*0,951*

IE([3,5])		IE_{cabelo}
0,954	–	*0,500*

IE([3,5])		IE_{creSol}
0,954	–	*0,607*

Choose the attribute ***cabelo*** because it exhibits maximum "information gain"

g(C, ***pele***) = 0,954 – 0,951 = **0,003**

g(C, ***cabelo***) = 0,954 – 0,500 = **0,454**

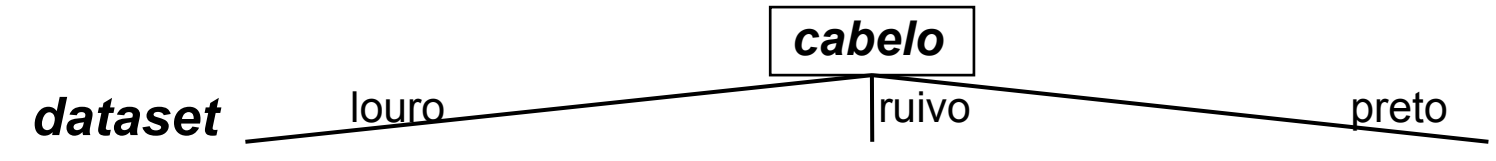
g(C, ***creSol***) = 0,954 – 0,607 = **0,348**

Paulo Trigo Silva

Construction of the decision tree

dataset

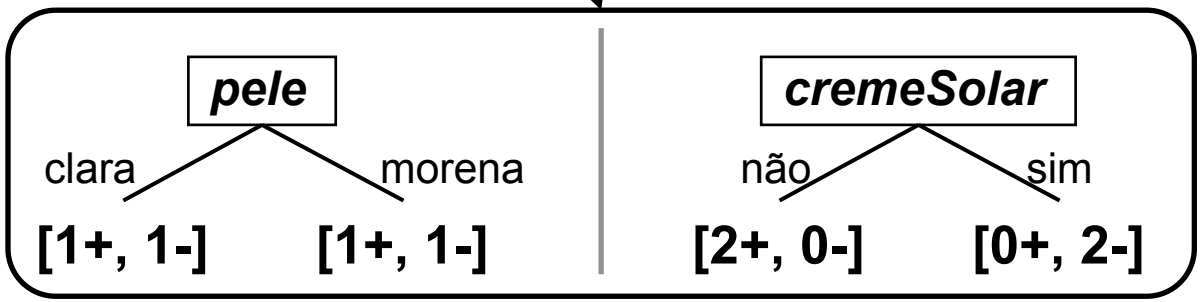
pele	cabelo	cremeSolar	queimaduraSolar
clara	louro	não	+
morena	louro	sim	-
morena	ruivo	não	+
clara	preto	não	-
morena	preto	não	-
morena	louro	não	+
morena	preto	sim	-
clara	louro	sim	-



pele	cremeSolar	queimaduraSolar
clara	não	+
morena	sim	-
morena	não	+
clara	sim	-

pele	cremeSolar	queimaduraSolar
morena	não	+

pele	cremeSolar	queimaduraSolar
clara	não	-
morena	não	-
morena	sim	-



Which attribute to choose in order to proceed with a new tree branch?

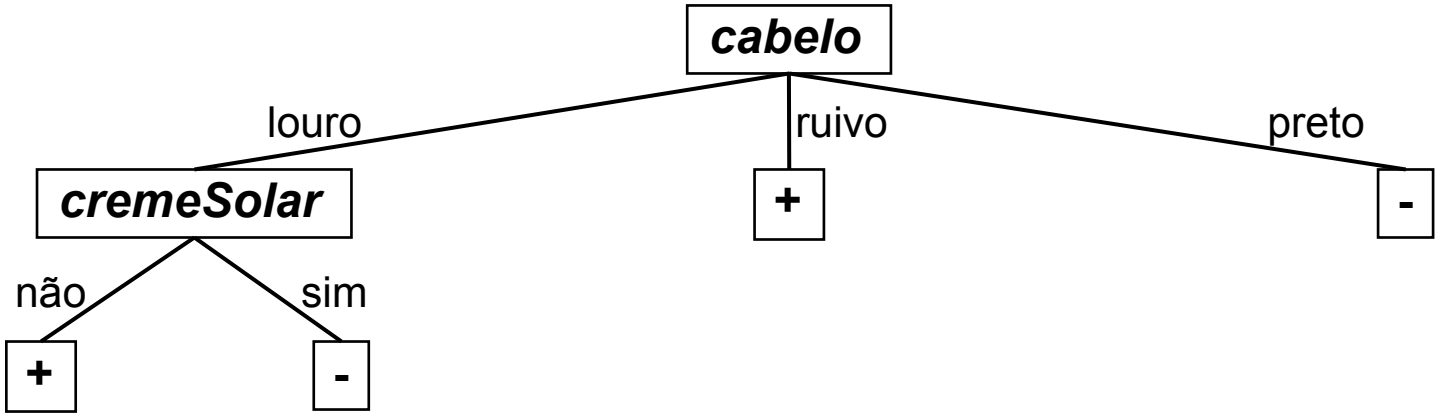
Why?

Paulo Trigo Silva

... and (at last!) the decision tree

training dataset

pele	cabelo	cremeSolar	queimaduraSolar
clara	louro	não	+
morena	louro	sim	-
morena	ruivo	não	+
clara	preto	não	-
morena	preto	não	-
morena	louro	não	+
morena	preto	sim	-
clara	louro	sim	-



What happens to someone with “cabelo louro” that does “not use creme solar”?
And to someone with “cabelo preto” that also “does not use creme solar”?
And happens to that one with “cabelo preto” that “uses creme solar”?

Paulo Trigo Silva

The variation of entropy – with two classes

It is interesting to analyze the entropy variation when the class only has 2 values(such as in the example we have been developing).

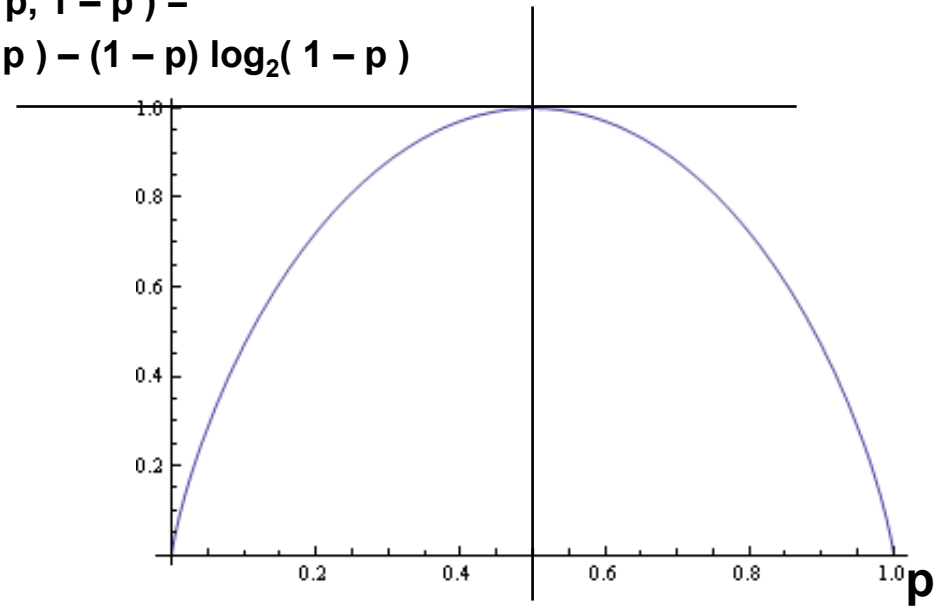
Let p be the probability of a class value; the other value's probability is 1 – p

If p=0 (1–p=1); entropy=0, so, maximum information gain; the same for p=1

If p=0.5; (half the examples for each class value); entropy=1 (maximum), so, minimum information gain

entropy(p, 1 – p) =

$-p \log_2(p) - (1-p) \log_2(1-p)$



Algorithm ID3 (*Iterative Dichotomiser 3*)

A synthetic description:

[»] ***assumption***: attributes with finite domain

[»] C ≡ current set of training examples

if C ≠ ∅, then

if all the examples in C have the same value, v, for the class,

then, the tree associated with C is a leaf with the value v from the class

else, ***choose-attribute-A*** for the root of the tree associated with C

the root A has ***n*** sub-trees, one for value of A: a_1, ..., a_n

split C into ***n*** sub-sets C_j = { x | A(x) = a_j }, for i = 1..***n***, and

apply, recursively, this construction process for each sub-tree using C_j as the current set of training examples

else the tree is a leaf with an undetermined class

choose-attribute-A (regarding the training set C):

[1] for each attribute consider each value and calculate its entropy (in C)

[2] for each attribute calculate the average entropy (on the entropy of its values)

[3] return the attribute with lowest average entropy (or maximum information gain)

Paulo Trigo Silva

Highly branching attributes

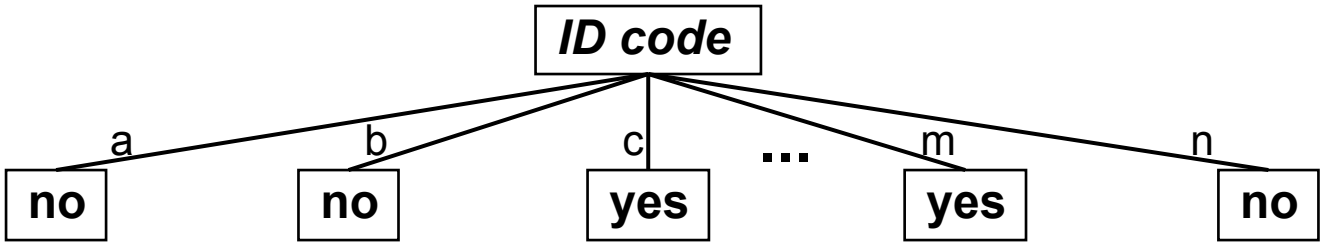
- Problem with attributes with a large number of possible values
  - the extreme case is an identification code (e.g., primary key)
- The increase on the number of values of an attribute
  - also increases its capability to generate pure (sub)sets
- The information gain measure tends (is biased) to prefer
  - attributes with large number of possible values
- ... this can contribute to the overfitting problem
  - i.e., select an attribute that is not the best for the prediction task
  - ... it is optimal at describing the training set (e.g., an identifier) but it can not realize a prediction

... a training dataset with an “identifier code”

ID code	Outlook	Temperature	Humidity	Windy	Play
a	sunny	hot	high	false	no
b	sunny	hot	high	true	no
c	overcast	hot	high	false	yes
d	rainy	mild	high	false	yes
e	rainy	cool	normal	false	yes
f	rainy	cool	normal	true	no
g	overcast	cool	normal	true	yes
h	sunny	mild	high	false	no
i	sunny	cool	normal	false	yes
j	rainy	mild	normal	false	yes
k	sunny	mild	normal	true	yes
l	overcast	mild	high	true	yes
m	overcast	hot	normal	false	yes
n	rainy	mild	high	true	no

Which decision tree has the highest information gain?

... the decision tree!



The expected information to specify the class value given the attribute value is:

IE([0; 1]) + IE([0; 1]) + IE([1; 0]) + ... + IE([1; 0]) + IE([0; 1])

which is zero because each of the 14 terms is always zero.

“ID code” identifies each instance and determines the class without ambiguity.

Therefore, the information gain of this attribute is just the information at the root,

g(C, “ID code”) = IE([9; 5]) – 0 = entropy(9/14; 5/14) = **0,940**

and this value is greater than the information gain of any other attribute.

But branching on the “ID code” is no good for predicting the class of unknown instances and tells nothing about the structure of the decision, which after all are the main goals of machine learning!

Paulo Trigo Silva

Gain Ratio – reduces the “ID code” effect

- The Gain Ratio (GR) compensates the Information Gain (IG)
  - reduces IG tendency to choose attributes with large number of values
- Gain Ratio (GR) accounts for the number and dimension of branches
  - takes into account the number and size of daughter nodes into which an attribute splits the dataset, disregarding any information about the class
  - “corrects” IG with the “intrinsic information”, “SplitInfo”, of each partition
  - ... **GR(attribute) = IG(attribute) / SplitInfo(attribute)**
- The “intrinsic information” also named as “SplitInfo” consists of
  - the entropy of the distribution of the instances through the branches
  - ... without considering any information about the class
  - i.e., “how much information to tell which is the branch of an instance?”

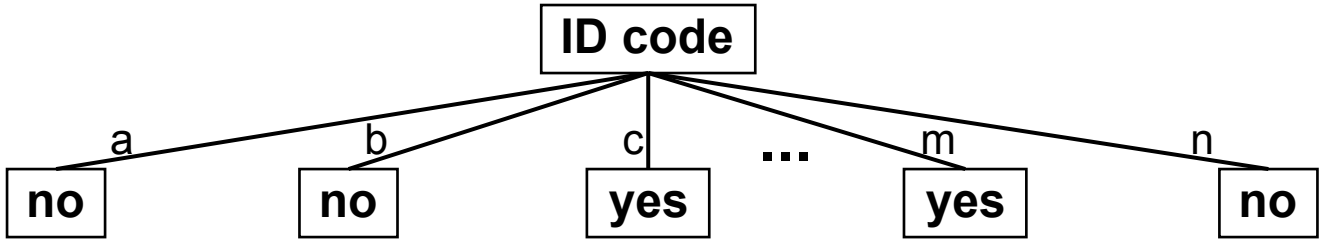
Paulo Trigo Silva

Gain Ratio (and "intrinsic information" – SplitInfo)

- Consider the attribute A and the training dataset C, where
  - A has the values a_1, a_2, ..., a_n
  - $C = C_1 \cup C_2 \cup ... \cup C_n$
  - C_i is the subset of C induced by the value a_i of A
- The intrinsic information of C, SplitInfo(C), is given by:
  - $\mathbf{SpiltInfo(A) = -\Sigma_{i=1..n} |C_i| / |C| \log_2 (|C_i| / |C|)}$
  - just looks at the dimension, $|C_i|$ and $|C|$, of partitions
  - does not pay any attention to the classes involved in the subsets
- The intrinsic information represents the potential (of information)
  - generated by splitting C in ***n*** subsets
  - it is higher for a more highly branching attribute
- Therefore the, Gain Ratio
  - is given by: **GR(A) = IG(A) / SplitInfo(A)**

Paulo Trigo Silva

Example – Gain Ratio



SplitInfo("ID code") = IE([1; 1; ...; 1]) = $- 1/14 \log_2 1/14 - \ ... \ - 1/14 \log_2 1/14$ =

= $14 \times (- 1/14) \times \log_2 (1/14)$ = $\log_2 (14)$ = **3,807**

g(C, "ID code") = IE([9; 5]) – 0 = entropy(9/14; 5/14) = **0,940**

GR("ID code") = IG("ID code") / SplitInfo("ID code") =

= g(C, "ID code") / SplitInfo("ID code") = 0,940 / 3,807 = **0,247**

SplitInfo("Outlook") = ?

GR("Outlook") = ?

What is the "Gain Ratio" of "Outlook"?

... example – Gain Ratio

ID code	Outlook	Temperature	Humidity	Windy	Play
a	sunny	hot	high	false	no
b	sunny	hot	high	true	no
c	overcast	hot	high	false	yes
d	rainy	mild	high	false	yes
e	rainy	cool	normal	false	yes
f	rainy	cool	normal	true	no
g	overcast	cool	normal	true	yes
h	sunny	mild	high	false	no
i	sunny	cool	normal	false	yes
j	rainy	mild	normal	false	yes
k	sunny	mild	normal	true	yes
l	overcast	mild	high	true	yes
m	overcast	hot	normal	false	yes
n	rainy	mild	high	true	no

SplitInfo("Outlook") = IE([5; 4; 5]) =

$= -5/14 \log_2 5/14 - 4/14 \log_2 4/14 - 5/14 \log_2 5/14 =$ **1,577**

GR("Outlook") = g(C, "Outlook") / 1,577 = 0,247 / 1,577 = **0,157**

Paulo Trigo Silva

Example – Gain Ratio for the remaining attributes

GR("ID code") = 0,247

and the GR for each one of the remaining attributes is given by:

Outlook		Temperature		Humidity		Windy	
info:	0.693	info:	0.911	info:	0.788	info:	0.892
gain: 0.940–0.693	0.247	gain: 0.940–0.911	0.029	gain: 0.940–0.788	0.152	gain: 0.940–0.892	0.048
split info: info([5,4,5])	1.577	split info: info([4,6,4])	1.557	split info: info ([7,7])	1.000	split info: info([8,6])	0.985
gain ratio: 0.247/1.577	0.157	gain ratio: 0.029/1.557	0.019	gain ratio: 0.152/1	0.152	gain ratio: 0.048/0.985	0.049

Note that, in this example, the "**ID code**" is still the one to be chosen;

however its **"advantage" is greatly reduced**.

In a practical implementation we can use an 'ad-hoc' constraint (test) to guard against splitting on such an useless attribute (such as "ID code)

Paulo Trigo Silva

Issues with Gain Ratio

- In some situation the Gain Ration can overcompensate
  - i.e., choose an attribute just because it “does not separate” the C set
  - an extreme scenario is the attribute, A, with 1 single value
  - ... where SplitInfo(A) = IE([|A|]) = $-$ |C| / |C| $\log_2$ (|C| / |C|) = 0
  - ... and so GR(A) = $+\infty$
- i.e., when we tend to the case of the attribute with 1 single value
  - it increases the possibility of choosing the attribute just because of that!

A way to “fix” such anomaly is:

choose the attribute that maximizes the gain ratio, provided that the information gain for that attribute is at least as great as the average information gain for all the attributes examined

Paulo Trigo Silva

Another “impurity” measure of a set

- **Gini Index**: impurity measure (used in IntelligentMiner – IBM)
  - pure set: iff all members belong to the same class
- Consider the training dataset C with examples from m class values
  - **Gini(C)** = $1 - \Sigma_{j=1..m}\ p_j^2$
  - where p_j is the relative frequency of class value j in C
- If the set C is separated into 2 subsets C_1 e C_2
  - with dimensions, respectively, N_1 e N_2
- ... then Gini Index has examples from the m classes and is given by
  - $\mathbf{Gini_{split}(C)} = (N_1 / m)\ Gini(C_1) + (N_2 / m)\ Gini(C_2)$
- The **attribute with lowest $Gini_{split}(C)$ is the chosen**
  - it is necessary to enumerate all possible “split points” for each attribute

Paulo Trigo Silva

ID3 – some characteristics

- The space of hypothesis being considered
  - is the set of univariate decision trees
- Search method
  - from the simplest to the more complex trees
- Search strategy
  - hill-climbing with the information gain heuristic
- ... depth search
  - there is just one alternative which is the current tree
- ... there is no backtracking in the search process
  - possible extension: “post-pruning”
- Does not guarantee to find the optimal solution, i.e., minimum tree

Paulo Trigo Silva

... Algorithms for the induction of Decision Trees

- Family of TDIDT machine learning algorithms
  - Top-Down Induction of Decision Trees
- Some algorithms
  - ID3 (Iterative Dichotomiser 3) [Quinlan, 1979]
  - CART (Classification and Regression Trees) [Brieman et al, 1984]
    - ◊ similar to ID3, other criterion for attribute selection
  - C4.5 [Quinlan, 1993] – extension to ID3 (*details later*)
  - See5, or C5.0 (commercial version of C4.5) [Quinlan, 1997]
- C4.5 extends ID3, essentially with respect to,
  - deal with numeric and missing attributes
  - revised impurity measure
  - generation of rules from the tree
  - evaluation of performance / tree pruning

Paulo Trigo Silva

C4.5 – numeric attributes

- ID3 creates as many descendants of node A as the values of A
  - hence it can only deal with attributes with a finite domain (discrete)
- Given A an attribute with a continuous domain, C4.5 does
  - binary tests about A with different, z, “threshold values”

Test “A > z” (new virtual binary attribute; with values “yes”, “no”)

- **[1]** sort the values of A in the training dataset C
  - $< a_1, \ldots a_k >$, is a vector with finite dimension because C is finite
- **[2]** there are k – 1 possible threshold values
  - which are the mean point of the intervals $] a_i, a_{i-1} [$
- **[3]** for each threshold z_i, calculate the information gain
  - considering test $A > z_i$ and choose the threshold with highest gain

Paulo Trigo Silva

Example

Consider the following training dataset:

Outlook	Temperature	Humidity	Windy	Play
sunny	85	85	false	no
sunny	80	90	true	no
overcast	83	86	false	yes
rainy	70	96	false	yes
rainy	68	80	false	yes
rainy	65	70	true	no
overcast	64	65	true	yes
sunny	72	95	false	no
sunny	69	70	false	yes
rainy	75	80	false	yes
sunny	75	70	true	yes
overcast	72	90	true	yes
overcast	81	75	false	yes
rainy	71	91	true	no

Build the "A > z" test on the attribute "Temperature"

Paulo Trigo Silva

Example – sort and thresholds (*Temperature*)

- **[1]** sort the values of A in the training dataset C
  - $< a_1, \ldots a_k >$, is a vector with finite dimension because C is finite

64	65	68	69	70	71	72	75	80	81	83	85
yes	no	yes	yes	yes	no	no yes	yes yes	no	yes	yes	no

- **[2]** there are k – 1 possible threshold values
  - which are the mean point of the intervals $] a_i, a_{i-1} [$

There are 11 thresholds; or 8 if we do not split values from same class.

Paulo Trigo Silva

Exemplo – limiares e ganho de informação (*Temperature*)

- **[3]** for each threshold z_i, calculate the information gain
  - considering test A > z_i and choose the threshold with highest gain

64	65	68	69	70	71	72	75	80	81	83	85
yes	no	yes	yes	yes	no	no yes	yes yes	no	yes	yes	no

Example – calculate the information gain for the threshold 71,5

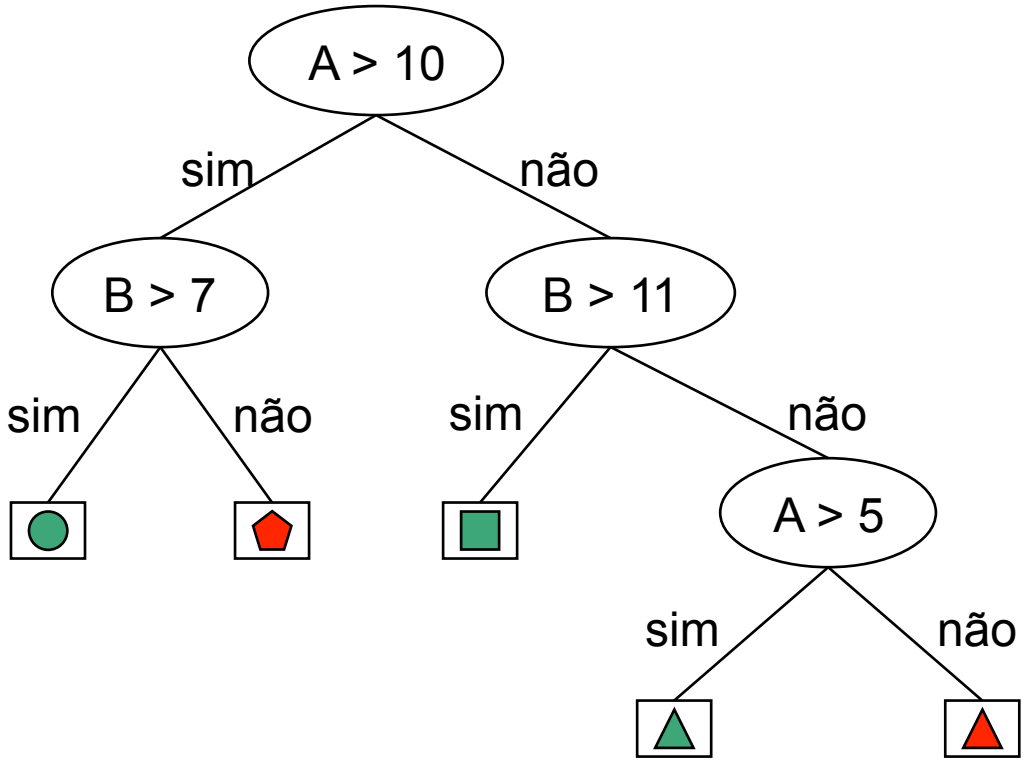
The information gain is calculated as previously seen.

e.g., for the test:
Temperature < 71,5 we have 4 ‘yes’ and 2 ‘no’
Temperature > 71,5 we have 5 ‘yes’ and 3 ‘no’

therefore,
IE([4; 2], [5; 3]) = (6/14) IE([4; 2]) + (8/14) IE([5; 3]) = **0,939**

C4.5 – the space of instances and the decision regions

With an univariate decision tree, the boundaries that separate the decision regions are parallel to the axes.



Represent this tree in the space (2D).

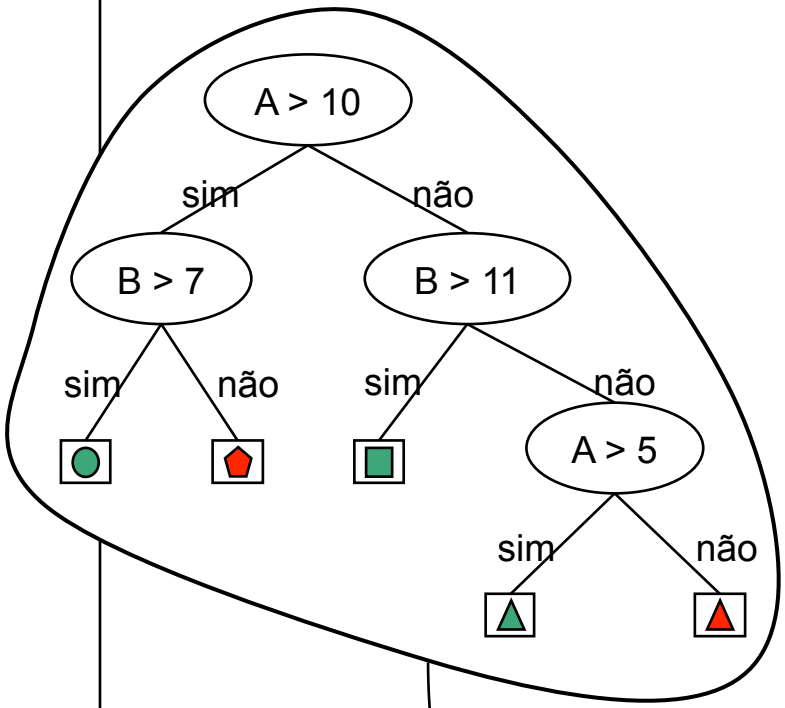
The attribute A in one axes (e.g., horizontal) and attribute B in the other axes (e.g., vertical).

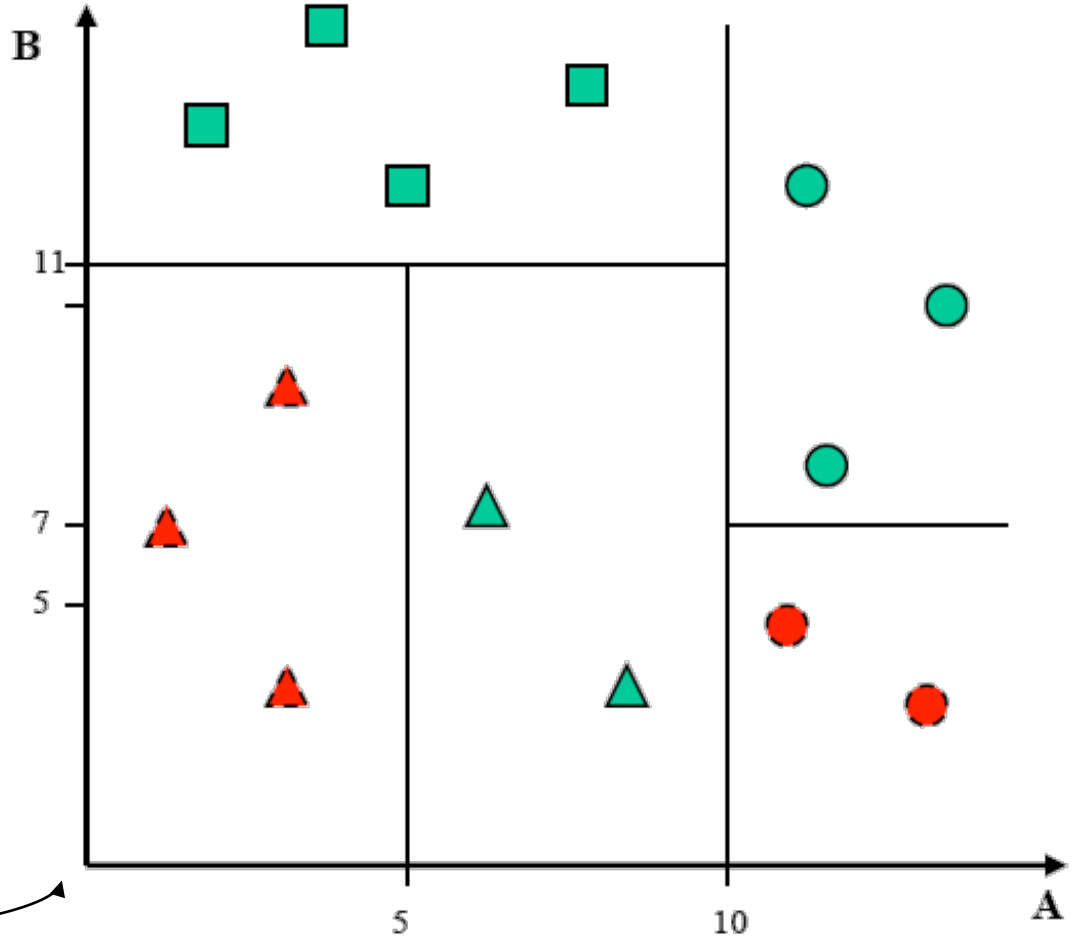
Axes are orthogonal.

Paulo Trigo Silva

C4.5 ... the decision region (numeric attributes)

... the boundaries that separate the decision regions are parallel to the axes.





C4.5 – missing values

- Treat them as just another possible value for the attribute
  - appropriate if the fact that attribute is missing is significant in some way
  - ... and, in that case no further action need to be taken!
- ... but, if there is no particular significance in the fact that
  - … a certain instance has a missing attribute value
  - then, a more subtle solution is needed!
- A simple (tempting) but not much viable solution, consists of,
  - ignore all instances in which some of the values are missing
  - ... not much viable, because instances with missing values often provide a good deal of information
  - … also, sometimes the attributes whose values are missing play no part in decision; in this case those instances are as good as any other

Paulo Trigo Silva

C4.5 – another approach for missing: *pseudo-examples*

- Assume that an instance with a missing value (of A)
  - is (notionally) split into pieces, one piece for each branch (i.e., a_i value)
  - ... in the same proportion as the known instances go down the branches
  - … so, we have an additional pseudo-example for each branch from A
- ... and, each pseudo-example is associated with a weight
  - the weight (of a_i) is given by the relative frequency of the training instances going down that branch (of a_i)
  - ... the sum of all weights equals 1
- The pseudo-examples contribute to decisions
  - in the usual way through the information gain calculation (or gain ratio)
  - Instead of having integer counts, the weights are used to compute gain
  - ... they may be further split at lower nodes if the values of other attributes as unknown as well

C4.5 – given a tree classify instance with missing values

- Given a tree how to classify an instance with missing values?
  - process identical to the tree construction with missing values
- ... build the pseudo-example
  - weight of a_i is assigned considering the proportion of the test instances that follow through that branch
  - ... the sum of all weights equals 1
- If we reach a leaf node with different classifications
  - we have to consider a composed classification
  - ... as a function of the weights assigned to the pseudo-examples

Decision Tree – an important characteristic

- The described method splits the set of training examples
  - until all partitions achieve “pure” subset
  - i.e., none of the tests produces a positive entropy
- ... may originate a complex tree and overfitted to training dataset
  - ideal classifier of the training dataset (with zero or minimum error)
- When used to classify another data (apart from the training)
  - may originate a higher error than the one produced by a simpler tree

It is desirable to have forms of:

- characterizing the notion of overfitting to training dataset
- reduce the overfitting produced by an algorithm

Paulo Trigo Silva

C4.5 – training dataset overfitting

- Notion of overfitting to the training data
  - a general notion that applies whenever a model is induced from data
  - ... i.e., it is not specific to the models that are represented as a tree
- We say that a model is overfitted to the training data
  - if another model worse adapted to those training data, i.e., with worse performance on those training data
  - has a better performance in the global distribution of instances, i.e., the performance “out-of training” compensates that of the training
- Possible performance measure (of a classification model)
  - error rate calculated from a set of pre-classified examples
- The overfitting can be produced by
  - noise (incorrectness) in the training data
  - small number or inadequate selection of training examples

Paulo Trigo Silva

C4.5 –Tree simplification with Pruning

- Approaches to reduce the decision tree overfitting problem
  - pre-pruning, i.e., stop expanding the tree before achieving pure nodes
  - post-pruning, i.e., reduce the tree after terminating its full construction
- ... for the pre-pruning (or forward pruning, or stopping) do:
  - determine the best partition of the current set of examples
  - evaluate the partition in the perspective of its statistical relevance (e.g., with χ^2 test), information gain, error reduction or other metric
  - if the evaluation value is lower than a predefined threshold then reject the tree expansion and the node is considered as a leaf with its majority class
- ... for the post-pruning (or backward pruning) do:
  - substitute a sub-tree by a leaf (sub-tree replacement)
  - substitute a node by a sub-tree descending from node (sub-tree raising)

Paulo Trigo Silva

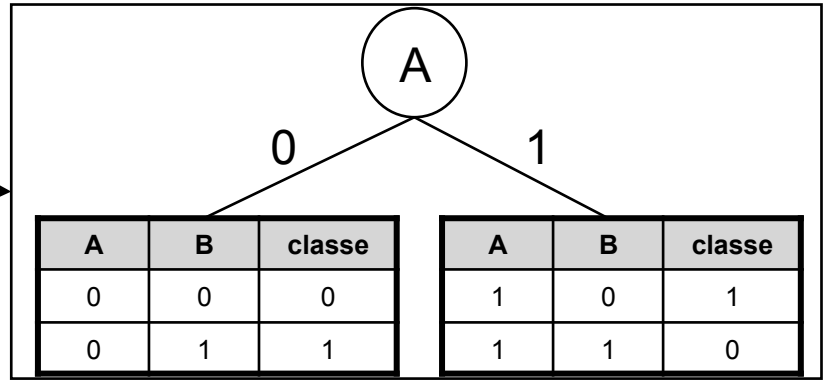
... pre-pruning – the early stopping problem

- The pre-pruning can prematurely stop the tree from growing
  - i.e., early stopping
- Classical example – the parity problem (or XOR)
  - the structure is only visible from the complete tree
  - none of the attributes has, by himself, significant association to class
  - ... so, in the XOR example pre-pruning does not expand the root node!
- But, in practice the XOR-type problems <u>are rarely found</u>
  - and the pre-pruning is much faster than the post-pruning

XOR

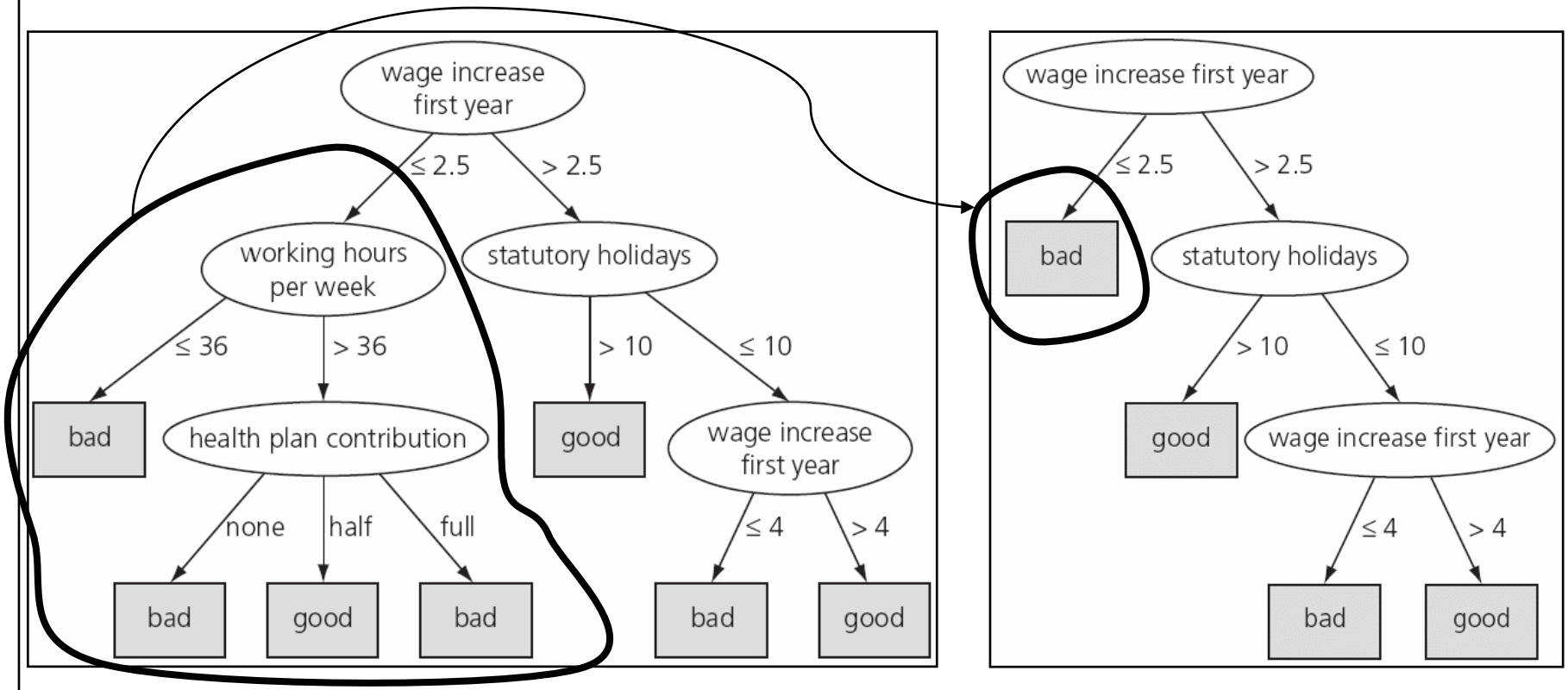
A	B	classe
0	0	0
0	1	1
1	0	1
1	1	0

uma árvore do XOR com pre-pruning



Paulo Trigo Silva

... post-pruning – the “subtree replacement” technique



1º. decide substitute the 3 children of “health plan contribution” by 1 single leaf

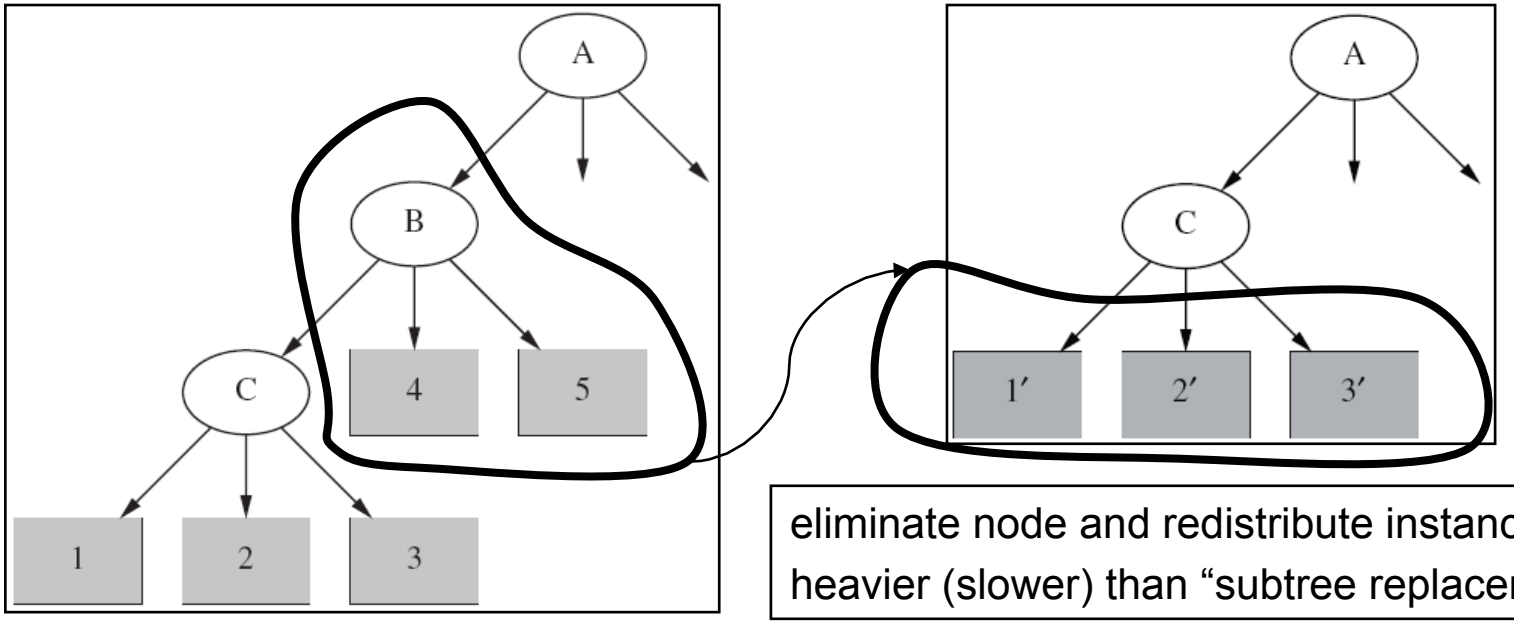
2º. decide substitute the 2 children of “work. hours per day” (now 2 leaves) by 1 leaf

3º. decide substitute the 2 children of “wage inc. first year” by 1 leaf [NO]

Question: which metric to use to make each one of those decisions?

Paulo Trigo Silva

... post-pruning – the “subtree raising” technique



eliminate node and redistribute instances
heavier (slower) than “subtree replacement”

Note that although the B and C descendants are leaves they could be entire trees

All the tree descending from C was “raised” to substitute the B sub-tree
In this case it is necessary to re-classify the examples from nodes 4 and 5;
that is why the nodes 1, 2, 3 are labeled as 1’, 2’, 3’; i.e., 1’, 2’, 3’ differ from 1, 2, 3, in that they include the instances coming from nodes 4 and 5

Question: which metric to use to make each one of those decisions?

Paulo Trigo Silva

Decision – “to do, or not to do, each substitution?”

- In the “sub-tree replacement” how to decide
  - to substitute an internal node by a leaf?
- In the “sub-tree raising” how to decide
  - to substitute an internal node by one of its descendant trees?

- Substitute only if it does not increase the estimated error
  - consider a set of testing instances
  - keep the original tree and construct the tree with the substitution
  - for each tree calculate the classification error of the testing instances
- ... compare the error of the trees before and after the substitution
  - if the tree with the substitution increases the error, then do not substitute

Paulo Trigo Silva

Which set of instances use to estimate the error?

- It is not a good idea to use the training dataset to estimate the error
  - it would not originate substitutions (pruning)
  - ... because the tree was built to classify that same dataset!
- A better idea is to split the training dataset into:
  - training examples, and
  - validation examples,
  - ... and this is named as *reduced-error pruning*
- The problem of the *reduced-error pruning*
  - training dataset with a small dimension
  - … the construction of the tree can loose too much relevant information

... the C4.5 method – training dataset to estimate error

- The C4.5 uses a heuristic method with statistical support
  - and estimates the error from the training dataset
- *IDEA*: consider the set of instances in each node
  - and imagine that the “majority rule ” is used to represent the node
- *EFFECT OF THE IDEA*: a certain quantity of “errors”, *E*
  - relative to the total number of instances, *N*
- *STATISTICAL SUPPORT*: “the truth probability of the error”
  - assume that *q* represents that (true) probability within a node
  - and that the *N* instances (from which *E* are errors) are generated by a Bernoulli process with parameter *q*
- In statistics, a sequence of independent events that
  - either have success or failure is described as a “Bernoulli process”

Paulo Trigo Silva

Bernoulli Process – motivation

- The classical example is the “coin tossing”
  - each toss is an independent event
- Assume that the prevision is to “always get heads”
  - so, rather than “heads” or “tails” each toss is either “success” or “failure”
- Now assume that the coin is “biased”
  - but we do not know what is the probability of heads
- Given a sequence of ***N* tosses**, from which ***S*** are **success**
  - then, the **observed success rate is *f* = *S* / *N***
  - ... but what can we say about the **true success probability, *p***?
- ... *p*, belongs to an interval with a certain specified confidence
  - e.g., if N=1000 and S=750, the true success rate is “around” 75%
  - ... but “how close” to 75% is the true success rate?

Paulo Trigo Silva

Bernoulli Process (BP) – confidence interval

- e.g., if N=1000 and S=750, true success rate is “around” 75%
  - ... but “how close” to 75% is this “true success rate”?
  - with **80%** of confidence it belongs to the interval [**73,2**% .. **76,7**%]
- e.g., if N=100 and S=75, true success rate is “around” 75%
  - ... but the experiment is smaller and therefore the interval is larger
  - i.e., with **80%** of confidence it belongs to the interval [**69,1**% .. **80,1**%]
- How to get the quantitative evaluation of the “confidence interval”?
  - in a single Bernoulli trial with success rate *p* we know that,
  - ... mean is *p* and the variance is *p* (1 – *p*)
- ... if N trials are taken from a Bernoulli process, then
  - the expected rate success *f* = *S* / *N* is a random variable
  - with same mean *p* and variance reduced by a factor *N*,
  - ... i.e., with variance *p* (1 – *p*) / N

We will see soon how to bet to these intervals!

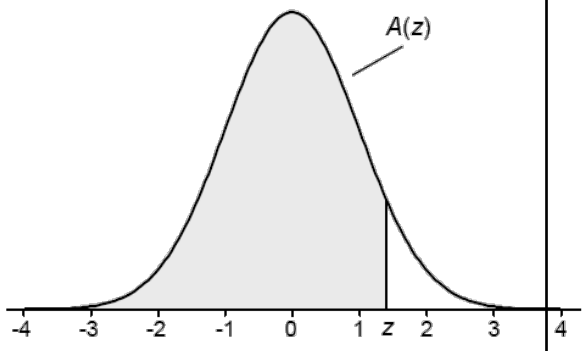
... Bernoulli – approaches the Gauss distribuition

- For N sufficiently large the random variable, $f = S / N$,
  - approaches the normal (Gauss) distribution
  - ... these are facts of statistics (we will not demonstrate them here)
- The probability that a random variable Z, with mean 0 (variance 1),
  - lies within a confidence range of width $2z$ is given by: $\Pr[-z \leq Z \leq z] = c$
- ... for a normal distribution (Gauss), the values of c and z
  - are given by tables with the values of
  - … the “Standard Normal Cumulative Distribution Function”

Paulo Trigo Silva

… see Standard Normal Cumulative Distribution $N(0, 1)$

z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.7	0.7580	0.7611	0.7642	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7852
0.8	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.9	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389
1.0	0.8413	0.8438	0.8461	0.8485	0.8508	0.8531	0.8554	0.8577	0.8599	0.8621
1.1	0.8643	0.8665	0.8686	0.8708	0.8729	0.8749	0.8770	0.8790	0.8810	0.8830
1.2	0.8849	0.8869	0.8888	0.8907	0.8925	0.8944	0.8962	0.8980	0.8997	0.9015
1.3	0.9032	0.9049	0.9066	0.9082	0.9099	0.9115	0.9131	0.9147	0.9162	0.9177
1.4	0.9192	0.9207	0.9222	0.9236	0.9251	0.9265	0.9279	0.9292	0.9306	0.9319
1.5	0.9332	0.9345	0.9357	0.9370	0.9382	0.9394	0.9406	0.9418	0.9429	0.9441
1.6	0.9452	0.9463	0.9474	0.9484	0.9495	0.9505	0.9515	0.9525	0.9535	0.9545
1.7	0.9554	0.9564	0.9573	0.9582	0.9591	0.9599	0.9608	0.9616	0.9625	0.9633
1.8	0.9641	0.9649	0.9656	0.9664	0.9671	0.9678	0.9686	0.9693	0.9699	0.9706
1.9	0.9713	0.9719	0.9726	0.9732	0.9738	0.9744	0.9750	0.9756	0.9761	0.9767
2.0	0.9772	0.9778	0.9783	0.9788	0.9793	0.9798	0.9803	0.9808	0.9812	0.9817
2.1	0.9821	0.9826	0.9830	0.9834	0.9838	0.9842	0.9846	0.9850	0.9854	0.9857
2.2	0.9861	0.9864	0.9868	0.9871	0.9875	0.9878	0.9881	0.9884	0.9887	0.9890
2.3	0.9893	0.9896	0.9898	0.9901	0.9904	0.9906	0.9909	0.9911	0.9913	0.9916
2.4	0.9918	0.9920	0.9922	0.9925	0.9927	0.9929	0.9931	0.9932	0.9934	0.9936
2.5	0.9938	0.9940	0.9941	0.9943	0.9945	0.9946	0.9948	0.9949	0.9951	0.9952
2.6	0.9953	0.9955	0.9956	0.9957	0.9959	0.9960	0.9961	0.9962	0.9963	0.9964
2.7	0.9965	0.9966	0.9967	0.9968	0.9969	0.9970	0.9971	0.9972	0.9973	0.9974
2.8	0.9974	0.9975	0.9976	0.9977	0.9977	0.9978	0.9979	0.9979	0.9980	0.9981
2.9	0.9981	0.9982	0.9982	0.9983	0.9984	0.9984	0.9985	0.9985	0.9986	0.9986
3.0	0.9987	0.9987	0.9987	0.9988	0.9988	0.9989	0.9989	0.9989	0.9990	0.9990
3.1	0.9990	0.9991	0.9991	0.9991	0.9992	0.9992	0.9992	0.9992	0.9993	0.9993
3.2	0.9993	0.9993	0.9994	0.9994	0.9994	0.9994	0.9994	0.9995	0.9995	0.9995
3.3	0.9995	0.9995	0.9995	0.9996	0.9996	0.9996	0.9996	0.9996	0.9996	0.9997
3.4	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9998
3.5	0.9998	0.9998	0.9998	0.9998	0.9998	0.9998	0.9998	0.9998	0.9998	0.9998
3.6	0.9998	0.9998	0.9999							



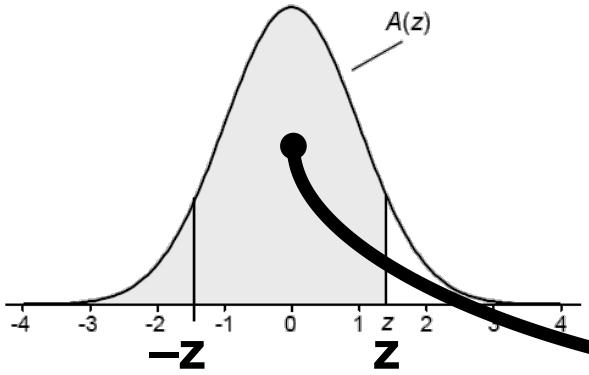
Pr[$Z \leq 1.65$] = 0.9505

What is the value of:

Pr[$-1.65 \leq Z \leq 1.65$]

?

... probability within an interval using the $N(0, 1)$



Pr[$Z \leq 1.65$] = 0.9505

... which is the value of:

Pr[$-1.65 \leq Z \leq 1.65$] ?

this area, between –z e z, is the one we are interested in!

Pr[$-1.65 \leq Z \leq 1.65$] =

Pr[$Z \leq 1.65$] – Pr[$Z \leq -1.65$] =

0.9505 – Pr[$Z \geq 1.65$] =

0.9505 – (1 – Pr[$Z \leq 1.65$]) =

0.9505 – (1 – 0.9505) =

$2 \times 0.9505 - 1 =$ **0.90**

we do not have the table for negative values because Z is symmetric, i.e.,

Pr[$Z \leq -z$] = Pr[$Z \geq z$]

This means that the probability of Z to occur more than 1.65 standard deviations from the average (above or bellow) is 90%

Paulo Trigo Silva

And now, back again to the Bernoulli Process

Recall: If X is $N(\mu, \sigma)$, then $Z = (X - \mu) / \sigma$ is $N(0, 1)$; where μ is the average and σ is the standard deviation.

- Remember, from the Bernoulli process, that variable $f = S / N$ has
  - average p and standard deviation $p(1 – p) / N$,
  - … i.e., standard deviation $(p(1 – p) / N)^{½}$
- Hence, for $\Pr[-z \leq Z \leq z] = c$, with Z normal distribution $N(\mu, \sigma)$,
  - doing $Z = (f – \mu) / \sigma$, we get,

$$\Pr\left[-z < \frac{f-p}{\sqrt{p(1-p)/N}} < z\right] = c$$

Now, rewrite that inequality as an equality and solving it for the probability ***p*** we have:

$$p = \left(f + \frac{z^2}{2N} \pm z\sqrt{\frac{f}{N} - \frac{f^2}{N} + \frac{z^2}{4N^2}} \right) \Big/ \left(1 + \frac{z^2}{N} \right)$$

Paulo Trigo Silva

... what is the purpose of the expression we have arrived?

The expression of the probability ***p***:

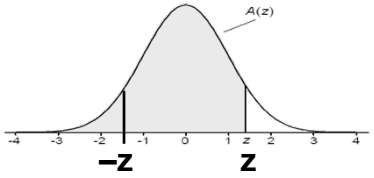
$$p = \left(f + \frac{z^2}{2N} \pm z\sqrt{\frac{f}{N} - \frac{f^2}{N} + \frac{z^2}{4N^2}} \right) \Big/ \left(1 + \frac{z^2}{N} \right)$$

- Note that the ± in expression gives two values to *p*, representing,
  - the inferior and the superior limits of the confidence interval
- i.e., $f = S / N$ doing $Z = (f - \mu) / \sigma$ we get the $N(0, 1)$, and
  - Pr[$-z \leq Z \leq z$] = *c*, we get $\Pr\left[-z < \frac{f - p}{\sqrt{p(1-p)/N}} < z \right] = c$
  - hence, **given *f*, *N* and *c* calculate the confidence interval *p* with**

Paulo Trigo Silva

Example – confidence level of the "true success rate"

- if N=1000 e S=750, the observed success rate is $f = S / N$ = 75% = 0,75
  - ... but "how close" to 75% is this "true success rate", p?
  - i.e., which is the interval that $\boldsymbol{p}$ belongs with the confidence level of c = **80%**?



c	Pr[Z ≤ z] = (c + 1)/2
0,8	0,9

de tabela Z (0, 1)			
z	S	N	f=S/N
1,28	750	1000	0,75

$$\text{Pr}[-z \le Z \le z] = 0.8 \Leftrightarrow$$

$$\text{Pr}[Z \le z] - \text{Pr}[Z \le -z] = 0.8 \Leftrightarrow$$

$$\text{Pr}[Z \le z] - \text{Pr}[Z \ge z] = 0.8 \Leftrightarrow$$

$$\text{Pr}[Z \le z] - (1 - \text{Pr}[Z \le z]) = 0.8 \Leftrightarrow$$

$$\mathbf{2 \times Pr[Z \le z] - 1} = 0.8 \Leftrightarrow$$

$$\text{Pr}[Z \le z] = (0.8 + 1)/2 \Leftrightarrow$$

$$\text{Pr}[Z \le z] = \mathbf{0.9} \Leftrightarrow \mathbf{z = 1.28}$$

symmetry of the normal

see the table from $N(0, 1)$

$$p = \left(f + \frac{z^2}{2N} \pm z\sqrt{\frac{f}{N} - \frac{f^2}{N} + \frac{z^2}{4N^2}} \right) \Big/ \left(1 + \frac{z^2}{N} \right)$$

$(f+z^2/2N \pm z(f/N - f^2/N + z^2/4N^2)^{1/2}) / (1+z^2/N)$	
p (cálculo com -)	p (cálculo com +)
0,732	0,767

In synthesis,

$\boldsymbol{p} \in$ [**0,732** .. **0,767**]

with **80%** of confidence

... the same example, with a lower dimension sampling

- if N=**100** and S=750, the observed success rate is $f = S / N$ = 75% = 0,75
  - ... this sampling (experiment) is lower than the previous, i.e., now **N=100**, so,
  - what happens, now, to the interval dimension to which p belongs with c = **80%**?
  - i.e., which is the interval that ***p*** now belongs with the confidence level c = **80%**?

i.e., which are, limInf, limSup, such that,

$\boldsymbol{p} \in [$ **limInf** .. **limSup** $]$

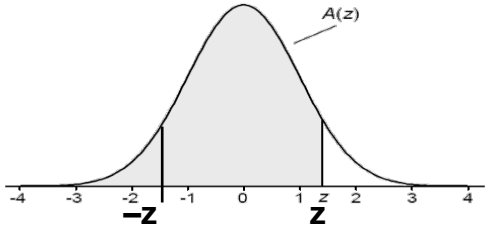
with **80%** of confidence

?

Paulo Trigo Silva

... the example with lower sampling (calculation detail)

- if *N*=**100** and *S*=750, the observed success rate is *f* = *S* / *N* = 75% = 0,75
  - ... this sampling (experiment) is lower than the previous, i.e., now **N=100**, so,
  - what happens, now, to the interval dimension to which *p* belongs with *c* = **80%**?
  - i.e., which is the interval that ***p*** now belongs with the confidence level *c* = **80%**?



c	Pr[Z ≤ z] = (c + 1)/2
0,8	0,9

de tabela Z (0, 1)			
z	S	N	f=S/N
1,28	75	100	0,75

$(f+z^2/2N \pm z\,(f/N - f^2/N + z^2/4N^2)^{1/2})\,/\,(1+z^2/N)$	
p (cálculo com -)	p (cálculo com +)
0,691	0,801

$\Pr[-z \le Z \le z] = 0.8 \Leftrightarrow$

$\mathbf{2 \times Pr[Z \le z] - 1} = 0.8 \Leftrightarrow$

$\Pr[Z \le z] = (0.8 + 1)/2 \Leftrightarrow$

$\Pr[Z \le z] = \mathbf{0.9} \Leftrightarrow \mathbf{z = 1.28}$

In synthesis, for N=100, S=75, we have

$p \in [\ \mathbf{0,691}\ ..\ \mathbf{0,801}\ ]$

with **80%** of confidence

i.e., the interval increases as the sampling dimension decreases (e.g., from *N*=1000 to *N*=100)

Paulo Trigo Silva

Now, back again, to the C4.5

Recall: the C4.5 estimates the error based on the training dataset and with some statistical assumptions

- *IDEA*: consider the set of instances in each node
  - and imagine that the “majority rule ” is used to represent the node
- *EFFECT OF THE IDEA*: a certain quantity of “errors”, *E*
  - relative to the total number of instances, *N*
  - … note that E = N – S (where S is the number of successes)
- *STATISTICAL SUPPORT*: “the truth probability of the error”
  - assume that *q* represents that (true) probability within a node
  - and that the *N* instances (from which *E* are errors) are generated by a Bernoulli process with parameter *q*

Paulo Trigo Silva

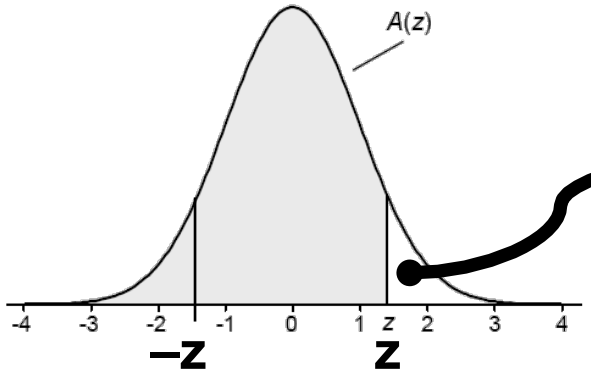
Bernoulli Process (Bp) & estimate error in the C4.5

- The Bernoulli process (Bp) is a general model
  - in C4.5 the Bp is used to provide a “statistical rationality”
  - ... and therefore the calculations for the error estimation;
  - but, applying the Bp to the C4.5 to estimate the error
  - ... implies analyzing two aspects: “failure and the origin of the data”.
- **[1]** in the Bernoulli process *p* represents the true success rate
  - within C4.5 the error estimation considers, *q*, the failure rate
  - therefore, given that $p + q = 1$, we need to make, $\boldsymbol{p} = 1 - \boldsymbol{q}$
- **[2]** the *N* and the *E* (*S* – *N*) are measured from the training dataset
  - which, in his turn, was used to build the tree!
- ... so, rather than estimate *p* as a confidence interval, we
  - make a **pessimistic** estimate of the error rate (over training data)
  - … by using just the **upper confidence limit**, i.e., **[limSup .. + ∞[**

Paulo Trigo Silva

C4.5 – estimate error being "pessimistic" (with Bp)

- Given a confidence level, *c*, find threshold *z*,
  - such that: Pr[$Z > z$] = *c*
  - ... which is a pessimistic estimate of Pr[$-z \leq Z \leq z$] = *c*
  - *i.e.*, $\Pr\left[\frac{f-q}{\sqrt{q(1-q)/N}} > z\right] = c$ where $f=E/N$ is the observed error rate



Z is a Bernoulli process (Bp) where *q* represents the "true" error rate.

so, for Pr[$Z > z$] = *c* we will get a confidence interval where the "true" error rate, *q*, is certainly higher than the one which would be comprised between –*z* and *z*; i.e., we get a **pessimistic** perspective of the "true" error rate.

C4.5 – pessimistic error estimation (with training dataset)

Pr[$Z > z$] = c, therefore,

$$\Pr\left[\frac{f-q}{\sqrt{q(1-q)/N}} > z\right] = c$$ where $f=E/N$ is the observed error rate

$$e = \frac{f + \frac{z^2}{2N} + z\sqrt{\frac{f}{N} - \frac{f^2}{N} + \frac{z^2}{4N^2}}}{1 + \frac{z^2}{N}}$$

solving in order of q and just for superior limit, i.e., only the + branch of expression

(recall both branches, ±, in the expression)

Intuition about the meaning of the value of z:

"number of standard deviations" corresponding to the confidence c.

C4.5 uses, by default, c = 25% = 0.25 i.e.,

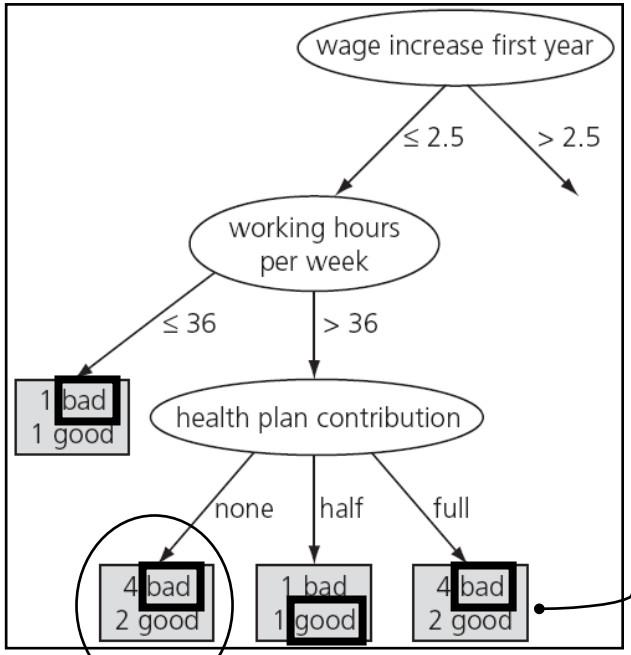
Pr[$Z > z$] = 0.25 ⇔ 1 – Pr[$Z \leq z$] = 0.25 ⇔ Pr[$Z \leq z$] = 0.75 ⇔ **z = 0.68**

consult the table from $N(0, 1)$

Paulo Trigo Silva

Example C4.5 – estimate error using the training dataset

Let us consider that c = 25%, therefore, z = 0.68



c	Pr[Z ≤ z] = 1 - c
0,25	0,75

consult the table from $N(0, 1)$

≡ class of the leaf

classification of the training examples

de tabela Z (0, 1)			
z	E	N	f=E/N
0,68	2	6	0,333

$$(f+z^2/2N+z\sqrt{(f/N-f^2/N+z^2/4N^2)})/(1+z^2/N)$$

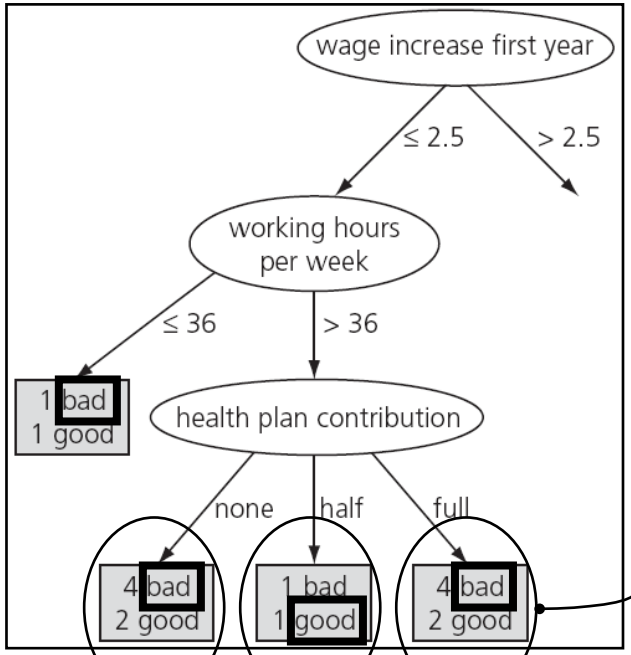
q
0,472

This means that: instead of using the training dataset error for this leaf, which is 2/6 = 33%, we use the pessimistic error estimate which is 47.2%

Paulo Trigo Silva

... example C4.5 – estimated error of each leaf

Let us consider that c = 25%, therefore, z = 0.68



≡ class of the leaf

classification of the training examples

c	Pr[Z ≤ z] = 1 - c
0,25	0,75

de tabela Z (0, 1)			
z	E	N	f=E/N
0,68	2	6	0,333

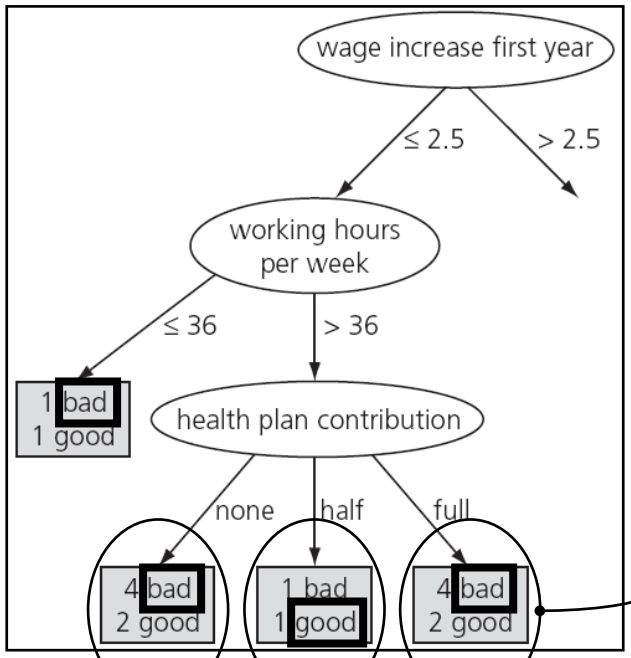
$(f+z^2/2N+z\sqrt{(f/N-f^2/N+z^2/4N^2)})/(1+z^2/N)$
q
0,472

de tabela Z (0, 1)			
z	E	N	f=E/N
0,68	1	2	0,5

$(f+z^2/2N+z\sqrt{(f/N-f^2/N+z^2/4N^2)})/(1+z^2/N)$
q
0,717

... example C4.5 – weighted mean of the estimated error

Let us consider that c = 25%, therefore, z = 0.68



≡ class of the leaf

classification of the training examples

q = 0.472 | q = 0.717 | q = 0.472

Weighted mean of the estimated error:

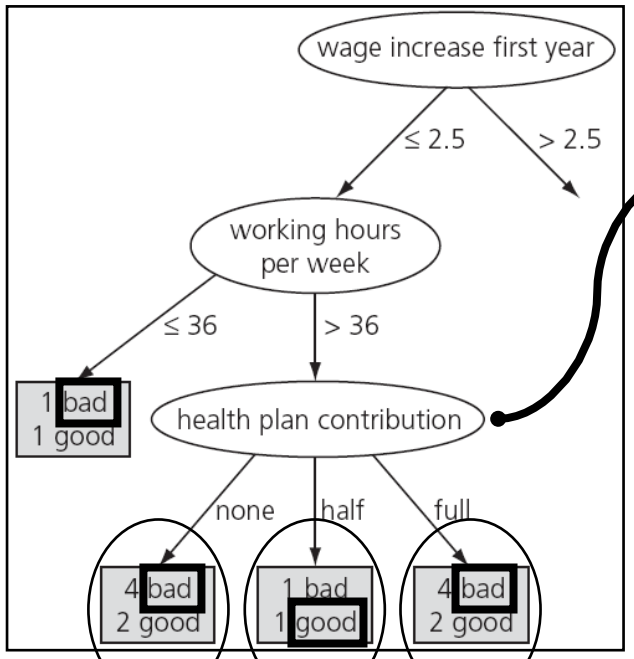
6/14 * 0.472 + 2/14 * 0.717 + 6/14 * 0.472 = **0.507**

Now we must see if the estimated error of the parent node is higher than the weighted mean of the estimated error of leaves ...

Paulo Trigo Silva

... example C4.5 – parent error 'versus' descendant error

Let us consider that c = 25%, therefore, z = 0.68



The parent node ("health plan contribution") has:

9 examples as "bad" and 5 as "good",

therefore, f=5/14

de tabela Z (0, 1)			
z	E	N	f=E/N
0,68	5	14	0,357

$$(f+z^2/2N+z\sqrt{(f/N-f^2/N+z^2/4N^2)})/(1+z^2/N)$$

q

0,448

q = 0.472 | q = 0.717 | q = 0.472

Weighted mean of the estimated error:

6/14 * 0.472 + 2/14 * 0.717 + 6/14 * 0.472 = **0.507**

Parent node error is lower than the error from descendants, so **prune**;

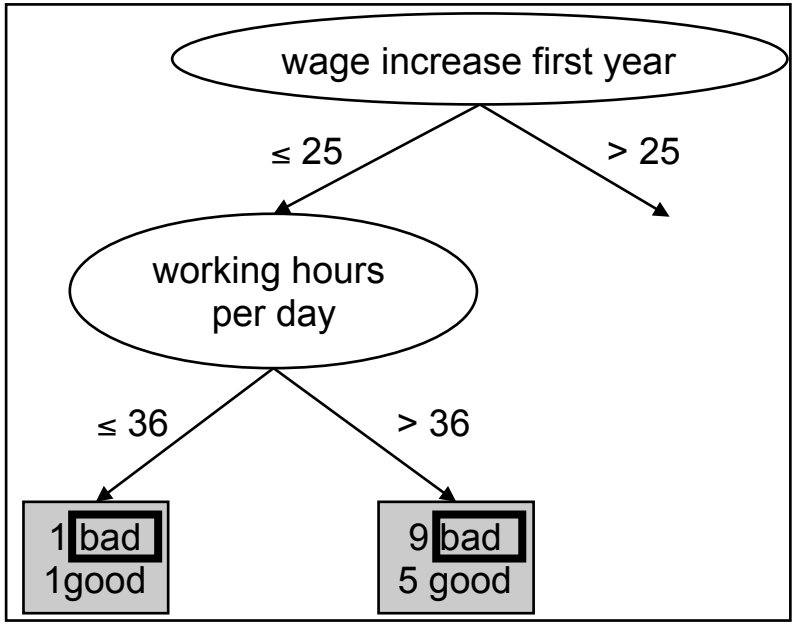
i.e., parent node becomes a leaf and classifies according to the "majority rule";

which, in this case is "bad"

Paulo Trigo Silva

... example C4.5 – keep pruning?

Let us consider that c = 25%, therefore, z = 0.68

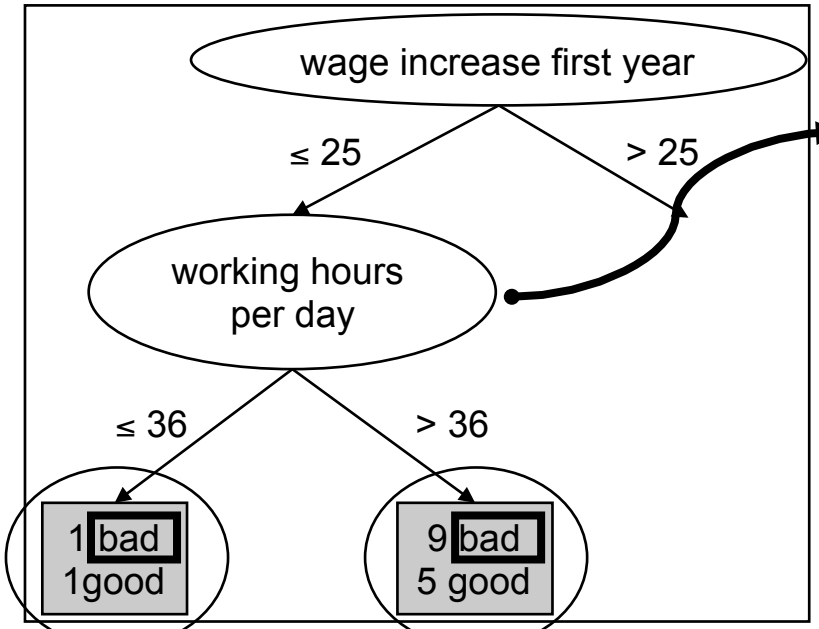


Should we prune the node "*working hours per day*"?

Paulo Trigo Silva

... example C4.5 – keep pruning? (calculations)

Let us consider that c = 25%, therefore, z = 0.68



The parent node ("working hours per day") covers: 10 examples as "bad" and 6 as "good", therefore, f=6/16

de tabela Z (0, 1)			
z	E	N	f=E/N
0,68	6	16	0,375

$(f+z^2/2N+z(f/N-f^2/N+z^2/4N^2)^{1/2})/(1+z^2/N)$
q
0,460

Parent node error is lower than the error from descendants, so **prune**

0.448
(*já visto atrás*)

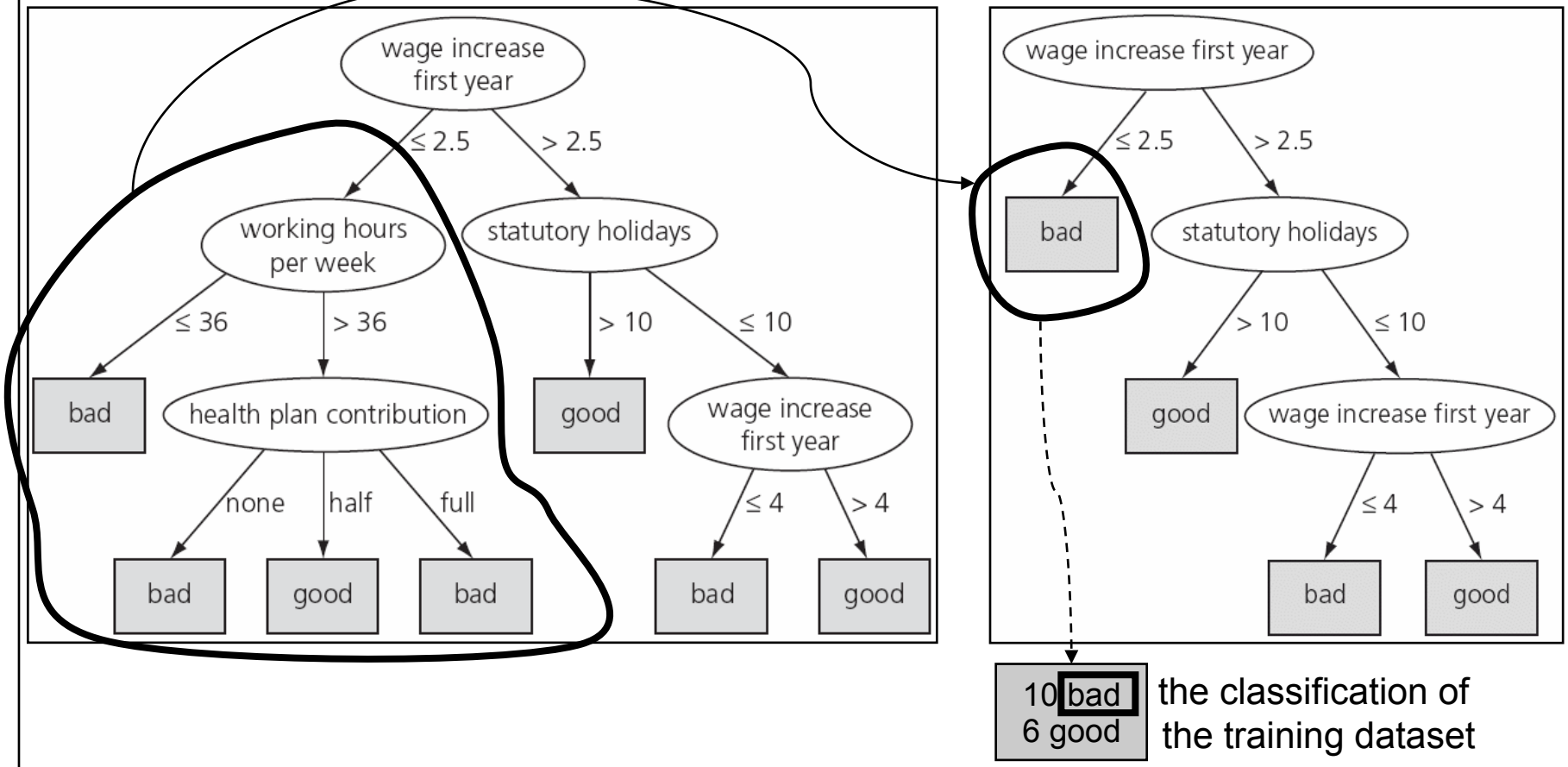
de tabela Z (0, 1)			
z	E	N	f=E/N
0,68	1	2	0,5

$(f+z^2/2N+z(f/N-f^2/N+z^2/4N^2)^{1/2})/(1+z^2/N)$
q
0,717

Weighted mean of the estimated error:
2/16 * 0.717 + 14/16 * 0.448 = **0.482**

Paulo Trigo Silva

C4.5 – synthesis of the example



1º. decide substitute the 3 children of "health plan contribution" by 1 leaf: "bad"

2º. decide substitute the 2 children of "work. hours per day" (now 2 leaves) by 1 leaf

3º. decide substitute the 2 children of "wage inc. first year" by 1 leaf [NO]

Paulo Trigo Silva

C4.5 – some considerations

- Assumption in the error estimate based on the training dataset
  - approximate to the normal distribution
  - estimate error as the upper limit of the confidence interval
  - use statistics taken from the training dataset
- ... despite those assumptions
  - the qualitative behavior of the error is correct
  - this method presents good practical results
- C4.5 has two configuration parameters
  - [1] confidence level, which is, by default, 25%
  - ... decrease the confidence level increases the amount of pruning
  - [2] minimum number of instances in the two branches with more instances
  - ... by default the minimum of instances is 2

Paulo Trigo Silva

C4.5 – about adjusting the confidence level

- C4.5 has a confidence level, ***c***, which, by default, is 25%
  - i.e., **Pr[$Z > z$] = c** $\Leftrightarrow$ $1 - \Pr[Z \leq z] = c$ $\Leftrightarrow$ **Pr[$Z \leq z$] = 1 – c**
- What is the effect of **decreasing the confidence level *c*** ?
  - i.e., decreases the area Pr[$Z > z$], so the **value of z increases**
- … as we **increase the value of *z***
  - **we also increase the value, *e*, of the error**
  - ... the ***e*** represents the pessimistic estimative of the “true” error rate
- ... i.e., as we increase the value of the error (***e***)
  - the estimative becomes “more pessimistic”;
  - i.e., the mean error in the leaves increases; so, iit also increases the possibility of that error being higher than the error of the parent node
- ... so, **decreasing *c* can increase the quantity of pruning**

Paulo Trigo Silva