Algorithms – essential methods

Paulo Trigo Silva
ptrigo@isel.pt

Abstract

What is there to know about a *dataset*? How to formulate hypothesis about the available data? How to search for rules within a *dataset*? Which criteria to choose in order to decide one among several possible rules? What is the discretization of a numeric domain? How to deal with examples with missing values (for some of its attributes)? How to discretize a domain with missing values? What is the overfitting problem? How to reduce the (possible) overfitting that results from the dicretization? These pages presents some contributions to answer those questions.

"Read Data" and "See Patterns"

Which knowledge, e.g., rule(s), can we extract after reading this data (this *dataset*)?

age	prescription	astigmatic	tear_rate	lenses
young	myope	yes	normal	hard
young	myope	no	normal	soft
young	hypermetrope	yes	reduced	none
young	hypermetrope	no	normal	soft
young	hypermetrope	no	reduced	none
presbyopic	myope	yes	reduced	none
presbyopic	myope	yes	normal	hard
presbyopic	hypermetrope	yes	reduced	none
presbyopic	hypermetrope	yes	normal	none
presbyopic	hypermetrope	no	normal	soft
presbyopic	hypermetrope	no	reduced	none
pre-presbyopic	myope	yes	reduced	none
pre-presbyopic	myope	yes	normal	hard
pre-presbyopic	myope	no	normal	soft
pre-presbyopic	hypermetrope	yes	normal	none
pre-presbyopic	hypermetrope	no	normal	soft

we need to know the *dataset* and "be aware" of the "business"

What is the meaning of the data we are reading?

Witout being aware of the meaning, how do we know **what to search?**

Witout being aware of the meaning, how to **interpret the findings?**

. . . but, what is there to know about the data (about the *dataset*)?

- **Structure (format)**. Tabular representation? Nominal values (e.g., *string*), numeric (e.g., *date*) or both? Discrete domains, continuous or both? Missing values? Incorrect values?
- **Semantics (meaning)**. What does each column (of a tabular format) means? What represents each nominal value and numeric domain? Why are there missing or incorrect values? Does the *dataset* covers a representative space of examples?

Structure – characterize the data available in the *dataset*

age	prescription	astigmatic	tear_rate	lenses
young	myope	yes	normal	hard
young	myope	no	normal	soft
...	...	...	...	...

Know the **structure of data** in order to **choose and configure techniques**.

	Formato
Tabular	
"bitmap"	
outro	

	Valor
Nominal	
Numérico	
Ambos	

	Domínio
Discreto	
Contínuo	
Ambos	

	Omissões
Sim	
Não	

	Incorrecções
Sim	
Não	

✠ Characterize, regarding its structure, the *dataset* `contact-lenses`.

. . . characterize the data in the *dataset* (`contact-lenses`)

age	prescription	astigmatic	tear_rate	lenses
young	myope	yes	normal	hard
young	myope	no	normal	soft
...	...	...	...	...

This is the usual structure of a *dataset* that relates concepts (without metrics nor temporal nor geographical data).

	Formato
Tabular	X
"bitmap"	
outro	

	Valor
Nominal	X
Numérico	
Ambos	

	Domínio
Discreto	X
Contínuo	
Ambos	

	Omissões
Sim	
Não	X

	Incorrecções
Sim	
Não	X

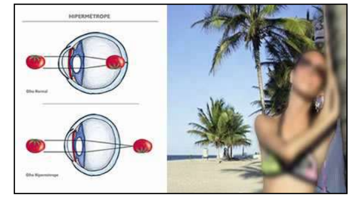
Semantics – grasp the meaning ("feeling") of problem's concepts

Knowing the **meaning of data** enables one to **draw hypothesis** and **interpret results**.

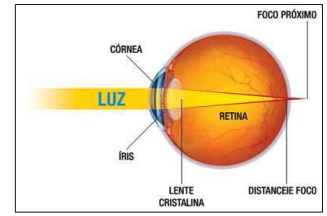
- **presbyopia**, literally means "aging eye" is an age-related eye condition that makes it more difficult, after the age of 40, to see very close,
- **hypermetropia**, vision better for distant objects (after 35 centimetres) than for near objects; eyeball too short, from front to back, so images are focused behind the retina,
- **myopia**, distant objects (more than de 35 centimetres) appear blurred because their images are focused in front of the retina rather than on it; close objects are focused,
- **astigmatism**, causes blurred vision due either to the irregular shape of the cornea, the clear front cover of the eye, or sometimes the curvature of the lens inside the eye,
- **contact lenses**, can be named (according to its material) as *hard* ou *soft* and the choice about the lenses type depends on the eye disease.

. . . grasp the meaning ("feeling") of problem's concepts


Visão com astigmatismo


Visão com hipermetropia

Visão com miopia



Visão com presbiopia

What to search in data? Which hypothesis (questions) to formulate?

age	prescription	astigmatic	tear_rate	lenses
young	myope	yes	normal	hard
young	myope	no	normal	soft
...	...	...	...	...

Several questions may be formulated. But, which is the one that makes most sense?

i. How is age influenced by the eye health and lenses' type?
ii. What is the expected tear rate given the age, eye pathology and the lenses' type?
iii. Which lenses' type to use given the age and the eye health aspects?
iv. Is astigmatism expected given my lenses' type, age and other eye health aspects?

✠ Choose the 1 question that seems to be the most appropriate. Justify your choice.

Question: *Which lenses' type to use?*

Atributos / Características (*Attributes* / *Features*) — Classe / Conceito / Alvo (*Class* / *Concept* / *Target*)

age	prescription	astigmatic	tear_rate	lenses
young	myope	yes	normal	hard
young	myope	no	normal	soft
...	...	...	...	...

We need to discover, in the data, **classification rules** to apply in future data.

We need to move from data into **rules that support a decision**.

An example of a decision: *Which lenses' type to use?*

✠ By looking only at the two tuples (above) is there any doubt on the decision? Why?

$0\mathcal{R}$ (*ZeroR*) – a very (excessively?) simple classification method

A decision process: Choose the **majority** option. When equal, choose randomly.

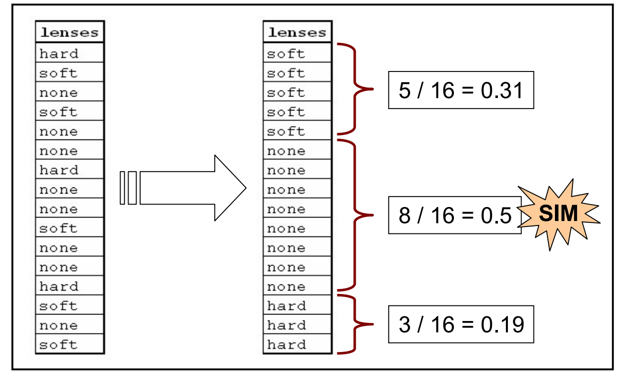
How to implement this decision process?

i. Sort the tuples given the concept (class).
ii. Identify each possible concept (class) option (value).
iii. Count the examples (instances) for each possible option (value).
iv. Choose the option with the most examples (instances).

✠ Apply the (above) decision process to the "contact lenses" *dataset*.

✠ What is the attributes' influence in this decision process?

$0\mathcal{R}$ (*ZeroR*) – the "contact lenses" *dataset*



$0\mathcal{R}$ *Rule (majority)*: **do not use contact lenses**.

$0\mathcal{R}$ – some characteristics

Ignores the knowledge about the domain.

Then, what is the purpose of knowledge acquisition?

Assumes that the knowledge from majority informs the individual choice.

But, if each individual follows the majority how to obtain such majority?
Through a "random approach"!

e.g., election with 4 candidates, each with 1/4 of the votes (from previous election).
Therefore, each voting person that follows the *ZeroR* principle makes a random choice.
If all voting people are *ZeroR* then the tendency if for the 1/4 tie to last forever.
If a person chooses on basis of other criteria, (s)he dictates the winner of a future election.

Choose using criteria (not just “majority rule”!)

Which attribute(s) (just 1, for now) best predicts the concept (each of its values)?

Atributos / Características (*Attributes* / *Features*) — Classe / Conceito / Alvo (*Class* / *Concept* / *Target*)

age	prescription	astigmatic	tear_rate	lenses
young	myope	yes	normal	hard
young	myope	no	normal	soft
...	...	...	...	...

e.g., consider just the 2 tuples (examples) below and the rules:
“`if age==young then hard`” — exhibits an error of 1/2
“`if astigmatic==yes then hard`” — exhibits an error of 0
Conclusion: usually, rules with **lowest error** provide **better prediction**.

$1\mathcal{R}$ (*OneR*) – search for rules involving 1 (a single) attribute

Idea: for each attribute build a rule that relates each value (of that attribute) with each value of the concept (class).

A process that **repeats for each attribute**:

i. for each value get the frequency (*freq*) that it relates with each value of the concept.
ii. for each value calculate the error (*total-relations-of-that-value* − *freq*).
iii. choose the pairs (*attribute-value*, *concept-value*) with the lowest error; random if tie.
iv. calculate the attribute's error as the summation of the chosen pairs' errors.

After getting the rules and the error of each attribute (items above): **choose the attribute with minimum error** and adopt its **rules** as representative of the *dataset*; choose random among equal minimum errors.

$1\mathcal{R}$ (*OneR*) – an example (abstract)

Atr1	Atr2	Classe
a1	b1	x
a1	b2	x
a1	b3	y
a1	b4	x
a2	b1	x
a2	b2	y
a2	b3	z

✠ Build the rules associated with each attribute.

✠ Get the rule that represents this *dataset* (according to the $1\mathcal{R}$ method).

. . . the rules for the attribute “Atr1”

	freq	x	y	z	total
Atr1	a1	3	1	0	*4*
	a2	1	1	1	*3*

erro = (*total* - *freq*) / *total* = 1 - *freq* / *total*		
x	y	z
1 / 4 = 0.25	3 / 4 = 0.75	4 / 4 = 1
2 / 3 = 0.67	**2 / 3 = 0.67**	**2 / 3 = 0.67**

regra de menor erro		**Atr1**
	erro	**erro**
a1 → x	1 / 4	3 / 7
a2 → **z** *	2 / 3	

* *aleatório*

✠ Build, identically, the rules associated with attribute “Atr2”.

. . . the rules for the attribute "Atr2"

	freq	x	y	z	total
Atr2	b1	2	0	0	2
	b2	1	1	0	2
	b3	0	1	1	2
	b4	1	0	0	*1*

erro = (*total* - *freq*) / *total* = 1 - *freq* / *total*		
x	y	z
0 / 2 = 0	2 / 2 = 1	2 / 2 = 1
1 / 2 = 0.5	**1 / 2 = 0.5**	2 / 2 = 1
2 / 2 = 1	**1 / 2 = 0.5**	**1 / 2 = 0.5**
0 / 1 = 0	1 / 1 = 1	1 / 1 = 1

regra de menor erro		**Atr2**
	erro	**erro**
b1 → x	0 / 2	2 / 7
b2 → x *	1 / 2	
b3 → y *	1 / 2	
b4 → x	0 / 1	

* *aleatório*

✠ Which rule represents the $dataset$?

$1\mathcal{R}$ – the rule for this (abstract) example

regra de menor erro		**Atr1**
	erro	**erro**
a1 → x	1 / 4	3 / 7
a2 → **z** *	2 / 3	

* *aleatório*

regra de menor erro		**Atr2**
	erro	**erro**
b1 → x	0 / 2	2 / 7
b2 → x *	1 / 2	
b3 → y *	1 / 2	
b4 → x	0 / 1	

* *aleatório*

SIM

The $1\mathcal{R}$ result:

Atr2 : b1 → x
Atr2 : b2 → x
Atr2 : b3 → y
Atr2 : b4 → x

$1\mathcal{R}$: a description of the algorithm

i. for each attribute, atr, do:
ii. for each one of its values, $atr\text{-}v$, do:
iii. get the frequency, k, that $atr\text{-}v$ associates to each value, $c\text{-}v$, of the class
iv. get the pair $(atr\text{-}v,\ k)$ with maximum k value
v. build a rule that consists on the tuple $(atr,\ atr\text{-}v,\ c\text{-}v)$
vi. calculate #errors of each rule: $(\sum_{atr-v} k) - k$; the rule's error is $\frac{\#errors}{\sum_{atr-v} k}$.
vii. for each attribute choose the rule, i.e., tuple $(atr,\ atr\text{-}v,\ c\text{-}v)$, of minimum error
viii. calculate #errors for each attribute: sum #errors of the chosen rules (previous step)
ix. choose the attribute with minimum #errors and consider that attribute's rules

Remark that "choose minumum error", is random whenever among equal error values.

$1\mathcal{R}$ – the example of the "contact lenses"

age	prescription	astigmatic	tear_rate	lenses
young	myope	yes	normal	hard
young	myope	no	normal	soft
young	hypermetrope	yes	reduced	none
young	hypermetrope	no	normal	soft
young	hypermetrope	no	reduced	none
presbyopic	myope	yes	reduced	none
presbyopic	myope	yes	normal	hard
presbyopic	hypermetrope	yes	reduced	none
presbyopic	hypermetrope	yes	normal	none
presbyopic	hypermetrope	no	normal	soft
presbyopic	hypermetrope	no	reduced	none
pre-presbyopic	myope	yes	reduced	none
pre-presbyopic	myope	yes	normal	hard
pre-presbyopic	myope	no	normal	soft
pre-presbyopic	hypermetrope	yes	normal	none
pre-presbyopic	hypermetrope	no	normal	soft

✠ Which knowledge i.e., $1\mathcal{R}$ rule, can we extract after reading this $dataset$?

$1\mathcal{R}$ – execution of the algorithm with the “contact lenses” *dataset*

```
- HYPOTHESES
- ( attr, valueAttr, valueTarget )       : ( error, total )
('age', 'young', 'soft')                 : (3, 5)
('age', 'pre-presbyopic', 'none')        : (3, 5)
('age', 'presbyopic', 'none')            : (2, 6)
('prescription', 'myope', 'hard')        : (4, 7)
('prescription', 'hypermetrope', 'none') : (3, 9)
('astigmatic', 'no', 'soft')             : (2, 7)
('astigmatic', 'yes', 'none')            : (3, 9)
('tear_rate', 'normal', 'soft')          : (5, 10)
('tear_rate', 'reduced', 'none')         : (0, 6)
```

```
- attrACCURACY
- attr         : ( error, total ) # error / total
age            : (8, 16)          # 0.5
prescription : (7, 16)            # 0.4375
astigmatic   : (5, 16)            # 0.3125  SIM
tear_rate    : (5, 16)            # 0.3125  SIM
```

```
- One-R
- ( attr, valueAttr, valueTarget ) : (error, total)
('tear_rate', 'normal', 'soft') : (5, 10)
('tear_rate', 'reduced', 'none') : (0, 6)
```


$1\mathcal{R}$ – some characteristics

- The method learns a single level tree; the leaves hold the class values.
- Attribute chosen only from its own error; influence of remaining attributes is ignored
- Well fitted when attributes are independent and a single attribute determines the class.

Although $1\mathcal{R}$ has some (obvious) limitations it also has very interesting capabilities:

- Accommodates missing values.
- Accommodates numeric domains (as well as nominals).
- Very fast to execute thus favouring the application to large volumes of data.
- Although simple the results can be compared with the most sophisticated (e.g., C4.5).

$1\mathcal{R}$ – an example that explores an extreme scenario

What is expected to happen after adding a new attribute with a single value (the same for all instances)?

i. The *dataset* cannot have an attribute with a single value [T? | F?]
ii. The $1\mathcal{R}$ assigns maximum error to this attribute [T? | F?]
iii. The $1\mathcal{R}$ chooses, for certain, this attribute [T? | F?]
iv. The error associated with this attribute is zero [T? | F?]
v. The attribute is chosen depending on the error associated with missing values [T? | F?]

✠ Assign the truth value of each (above) statement [T $\equiv$ true | F $\equiv$ false]

✠ Justify your choices.

$1\mathcal{R}$ – accommodates missing values

“$1\mathcal{R}$ handles missing values by treating “missing” as a legitimate value”

i.e., each missing value is filled with a new value, e.g., “?”, “missing”, “NA”.

Atr1	Atr2	Atr3	Classe
a1	b1	c1	x
a1	b2		x
	b3		
	b4		
a2	b1		x
a2			y
a2	b3		z

✠ Build this *dataset* rules, for each attribute, and get the attribute with minimum error.

$1\mathcal{R}$ – rules for the *dataset* with missing values

```
- HYPOTHESES
- ( attr, valueAttr, valueTarget ) : ( error, total )
('Atr1', 'a1', 'x')              : (0, 2)
('Atr1', '?', '?')               : (0, 2)
('Atr1', 'a2', 'y')              : (2, 3)
('Atr2', 'b1', 'x')              : (0, 2)
('Atr2', 'b2', 'x')              : (0, 1)
('Atr2', 'b3', 'z')              : (1, 2)
('Atr2', 'b4', '?')              : (0, 1)
('Atr2', '?', 'y')               : (0, 1)
('Atr3', 'c1', 'x')              : (0, 1)
('Atr3', '?', '?')               : (4, 6)
```

```
 attrACCURACY
- attr : ( error, total ) # error / total
Atr1 : (2, 7) # 0.285714285714
Atr2 : (1, 7) # 0.142857142857 SIM
Atr3 : (4, 7) # 0.571428571429
```

```
- One-R
- ( attr, valueAttr, valueTarget ) : (error, total)
('Atr2', 'b1', 'x') : (0, 2)
('Atr2', 'b4', '?') : (0, 1)
('Atr2', 'b2', 'x') : (0, 1)
('Atr2', 'b3', 'z') : (1, 2)
('Atr2', '?', 'y')  : (0, 1)
```


$1\mathcal{R}$ – accommodates numeric domains (apart from nominals)

Convert numeric attributes into nominal ones uses a simple discretization method:

i. Sort the *dataset* according to the values of the numeric attribute.
ii. Discretization involves partitioning the generated sequence (after the previous step).

How do define the breakpoints that constitute the boundaries of each partition?

i. If two consecutive values, of the class, are different, then break in the middle point of the corresponding values of the numeric attribute.
ii. If two consecutive values, of the class, are different, but the corresponding numeric values are equal, then move forward to the next numeric value.
iii. In the previous case (item ii) the partition contains more than one value of the class; therefore, assign the value (of the class) that corresponds to the majority; in case there is no majority move forward until the next change in the class value (cf., item i).

$1\mathcal{R}$ – an example of the discretization process

EstadoTempo	Temperatura	Humidade	Ventoso	Jogar?
soalheiro	29	85	falso	não
soalheiro	26	90	verdade	não
nublado	28	86	falso	sim
chuvoso	21	96	falso	sim
chuvoso	20	80	falso	sim
chuvoso	18	70	verdade	não
nublado	17	65	verdade	sim
soalheiro	22	95	falso	não
soalheiro	20	70	falso	sim
chuvoso	24	80	falso	sim
soalheiro	24	70	verdade	sim
nublado	22	90	verdade	sim
nublado	27	75	falso	sim
chuvoso	21	91	falso	não

✠ In *dataset* consider the class "`Jogar`" and make the discretization of "`Temperatura`".

. . . example – sort by "`Temperatura`"

EstadoTempo	**Temperatura**	Humidade	Ventoso	Jogar?
nublado	17	65	verdade	sim
chuvoso	18	70	verdade	não
chuvoso	20	80	falso	sim
soalheiro	20	70	falso	sim
chuvoso	21	96	falso	sim
chuvoso	21	91	falso	não
soalheiro	22	95	falso	não
nublado	22	90	verdade	sim
chuvoso	24	80	falso	sim
soalheiro	24	70	verdade	sim
soalheiro	26	90	verdade	não
nublado	27	75	falso	sim
nublado	28	86	falso	sim
soalheiro	29	85	falso	não

✠ Build partitions of "`Temperatura`" according to the values of the class "`Jogar`".

. . . example – build partitions of “Temperatura”

Apply the rule:

“If two consecutive values, of the class, are different, then break in the middle point of the corresponding values of the numeric attribute”.

Pontos de Partição: ***17,5   19   ?   ?   25   26,5   28,5***

Temperatura	17	18	20	20	21	21	22	22	24	24	26	27	28	29
Jogar?	sim	não	sim	sim	sim	não	não	sim	sim	sim	não	sim	sim	não
Temperatura	C1	C2	C3			C4		C5			C6	C7		C8

But, which breakpoint separates the partitions C3 and C4; and partitions C4 and C5?

The “breakpoint” cannot be located in a place with a change in the value of the class.

The “breakpoint” moves forward into the next position; and then, analyse again.

. . . example – build partitions of “Temperatura” (cont.)

Pontos de Partição: ***17,5   19   ?   ?   25   26,5   28,5***

Temperatura	17	18	20	20	21	21	22	22	24	24	26	27	28	29
Jogar?	sim	não	sim	sim	sim	não	não	sim	sim	sim	não	sim	sim	não
Temperatura	C1	C2	C3			C4		C5			C6	C7		C8

Pontos de Partição: *17,5   19   25   26,5   28,5*

Temperatura	17	18	20	20	21	21	22	22	24	24	26	27	28	29
Jogar?	sim	não	sim	sim	sim	não	não	sim	sim	sim	não	sim	sim	não
Temperatura	C1	C2	C3								C4	C5		C6

Nas instâncias de valor 21 o “ponto de partição” avança 1 posição.

For the value 22 the “breakpoint” moves forward while the values of the class do not change; it stops when the class value changes as well as the attribute values.

Characteristics of this discretization method

This method has the tendency to **form a high number pf partitions**.

The $1\mathcal{R}$ **algorithm tends to choose an attribute that originates a high number of partitions**; because increasing the number of partitions also makes it more likely that an instance belongs to the same class of the majority within its partition.

The extreme scenario is that of an attribute with a different value for each instance.

Such an **attribute** is designated by **identification code attribute**.

The **identification code attribute** has **zero error** because each partition has 1 instance.

The identification code attribute **cannot predict** new examples (outside the $dataset$).

The problem of “overfitting”

The identification code attribute **cannot predict** new examples (outside the $dataset$).

The **overfitting** consists of:

the $dataset$ fitting a model “too well”

the model describes data nearly perfectly, but the model is too rigid to fit any other sample; it is not loose enough to serve the predictive needs.

$Consequences$: **difficult** to obtain good results **outside** the training $dataset$.

$1\mathcal{R}$ has (imminent) overfitting when an attribute has a high number of possible values.

. . . overfitting is a common problem among classification algorithms (not just $1\mathcal{R}$)

$1\mathcal{R}$ - reduce the overfitting that results from discretization

$1\mathcal{R}$ has (imminent) overfitting when an attribute has a high number of possible values.

Idea to reduce the possibility of overfitting that results from discretization:
the **majority**, of class values in each partition, with a **minimum** number of examples.
Effect: increases the dimension of each partition; thus, reduces the number of partitions.

Example – again the attribute "`Temperatura`" and the class "`Jogar`":

Temperatura	17	18	20	20	21	21	22	22	24	24	26	27	28	29
Jogar?	sim	não	sim	sim	sim	não	não	sim	sim	sim	não	sim	sim	não

Temperatura	

✠ Build partitions of "`Temperatura`" with a minimum of 3 examples of class "`Jogar`".

. . . the steps to build the partitions

Pontos de Partição: *25*

Temperatura	17	18	20	20	21	21	22	22	24	24	26	27	28	29
Jogar?	sim	não	sim	sim	sim	não	não	sim	sim	sim	não	sim	sim	não
Temperatura	C1										C2			

- minimum of 3 examples: `sim não sim sim | sim` . . .
- break when change: `sim não sim sim sim | não` . . .
- assign breakpoint (21.5): `sim não sim sim sim não | não` . . .
- break when change: `sim não sim sim sim não não | sim` . . .
- assign breakpoint (23): `sim não sim sim sim não não sim | sim` . . .
- break when change: `sim não sim sim sim não não sim sim sim | não` . . .

About last partition: may not satisfy the constraint and may originate a random choice.

Discretization with missing values

Idea for discretization of numeric domain with missing values:

i. consider that all missing values are represented by a single class value, and
ii. execute the discretization process for the instances for which the value is defined.

Example: discretization of 3 instances with missing value of "`Temperatura`":

Pontos de Partição: *21.5* *28.5*

Temperatura	18	20	20	21	22	22	24	26	27	28	29			
Jogar?	não	sim	sim	sim	não	sim	sim	não	sim	sim	não	não	sim	sim
Temperatura	C1				C2						C3	C4		

Discretization originated 4 class values; `C4` represents the attribute missing values.

$1\mathcal{R}$ – "simple but ranked among the best"

About benchmarking results for $1\mathcal{R}$ (cf., details in [1]:

A study is published (using 16 *dataset* explored by researchers to evaluate their algorithms) where, despite its simplicity, the $1\mathcal{R}$ learns rules just few points less accurate than the decision trees produced by C4.5 (to see later).

[1] Robert C. Holte; "Very simple classification rules perform on most commonly used datasets"; Machine Learning Journal 11:63–91; 1993.

A very important lesson (cf., details in [2]):

"Simple ideas often work very well; we recommend a **simplicity-first methodology** when analyzing practical datasets". $1\mathcal{R}$ often gives a useful first impression of a *dataset*.

[2] Ian Witten and Eibe Frank; Elsevier; "Data Mining - Practical Machine Learning Tools and Techniques"; Morgan Kaufmann; 2nd Ed; 2005; (cf. pág 83).

$1\mathcal{R}$ – implementation and integration in the “Orange” framework

An implementation (Python) of $1\mathcal{R}$ available in: `oneR_classifier.pyc`

The “contact lenses” *dataset* is available in: `lentes.tab`

Integration of $1\mathcal{R}$ in the “Orange” framework (`http://www.ailab.si/orange/`):