

Decision trees

Data Mining course
Master in Information Technologies
Enginyeria Informàtica

Tomàs Aluja



UNIVERSITAT POLITÈCNICA
DE CATALUNYA

Decision trees in Data Mining

- The central task in DM is prediction (classification if response is categorical or regression if response is continuous)
- There are plenty of methods (models) to perform prediction: linear regression, logistic regression, discriminant analysis, Naive Bayes, Knn, Neural networks, SVM, ...
- The decision trees are a non parametric method to perform predictions. Its popularity comes from the simplicity of the results (everyone can understand them) and their direct operationality

Purpose

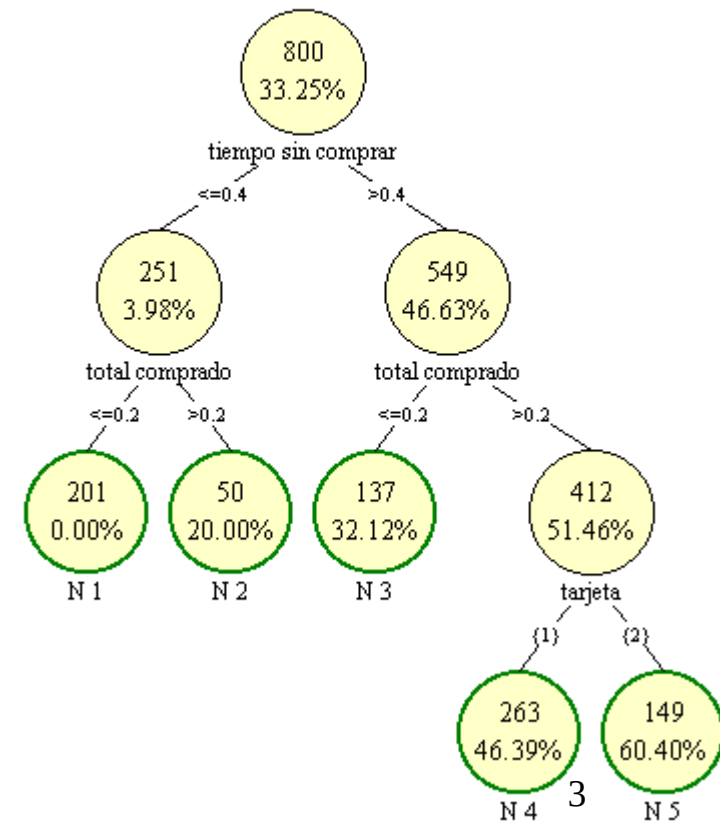
To segment the given data in order to find homogeneous groups *respect to* the response variable.

Visual output

Árbol: Árbol buenos compradores
Var.Resp: compradores

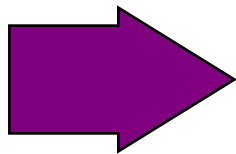
Trees may differ:

- Type of the response variable (con/cat)
- Type of explanatory variables (bin, nom, ord, con)
- Binary or multi-way
- Split criterion (information, gini, ...)
- Stop criterion (pre-pruning, post-pruning)



Algorithm (top-down)

1. To set all individuals in the root node
2. Evaluate all possible splits and find the optimal partition in children nodes.
3. For every child node: Decide if we stop the process or we go back to step 2.



We need

- ***a split criterion***
- ***a stop criterion (if prepruning)***

How many splits can we make in a node

- Splits are defined by the number and the type of the explanatory variables and the type of the tree

	<i>Multi way</i>	<i>Binary tree</i>
Binary	<i>1</i>	<i>1</i>
Nominal	<i>1</i>	$2^{q-1}-1$
Ordinal	<i>1</i>	$q-1$
Continuous	n_t-1	n_t-1

★ Attention

★ Attention

Antecedents

- AID (Sonquist y Morgan, 1964)
 - Response: Continuous
 - Binary tree, but can be extended to multiway
 - Split criterion: Fisher's F
 - Stop criterion: threshold upon the p.value of the split
- CHAID (Kass, 1980)
 - Response: Categorical
 - Binary tree, but can be extended to multiway
 - Split criterion: χ^2
 - Stop criterion: threshold upon the p.value of the split

AID Automatic Interaction Detection

(Sonquist & Morgan 1963)

AID split criterion based in the wellknown decomposition of variance, between the nodes and within the nodes. The interest is to maximize the between part.

$$\sum_{i=1}^n (y_i - \bar{y})^2 = \sum_{k=1}^q n_{t_k} (\bar{y}_{t_k} - \bar{y})^2 + \sum_{k=1}^q \sum_{i \in t_k} (y_{it_k} - \bar{y}_{t_k})^2$$

$$F = \frac{\sum_{k=1}^q n_{t_k} (\bar{y}_{t_k} - \bar{y})^2 / q - 1}{\sum_{k=1}^q \sum_{i \in t_k} (y_{it_k} - \bar{y}_{t_k})^2 / n - q} \sim F_{q-1, n-q}$$

The (categorical) variable giving the most significative F defines the optimal split

CHAID (Kass, 1980)

CHAID split criterion: based in the Chi-square statistic computed crossing the response variable with the tentative partition defined by a explanatory variable.

The most significant is the Chi-square, more different is the distribution of the response in the parent node respect to the children nodes

partition

response	$\vdots$	$\vdots$	$\vdots$	
	n_{k1}	n_{kj}	n_{kp}	$n_{k\cdot}$
	$\vdots$	$\vdots$	$\vdots$	
	$n_{\cdot j}$			n

$$\chi^2 = \sum_{k=1}^q \sum_{j=1}^m \frac{\left(n_{jk} - n_k \frac{n_j}{n} \right)^2}{n_k \frac{n_j}{n}}$$

q : Number of child nodes (=2 if binary tree)

m : number of response classes

The (categorical) variable giving the most significative χ^2 defines the optimal split



The CART solution



Leo Breiman,
American, 1928-2005

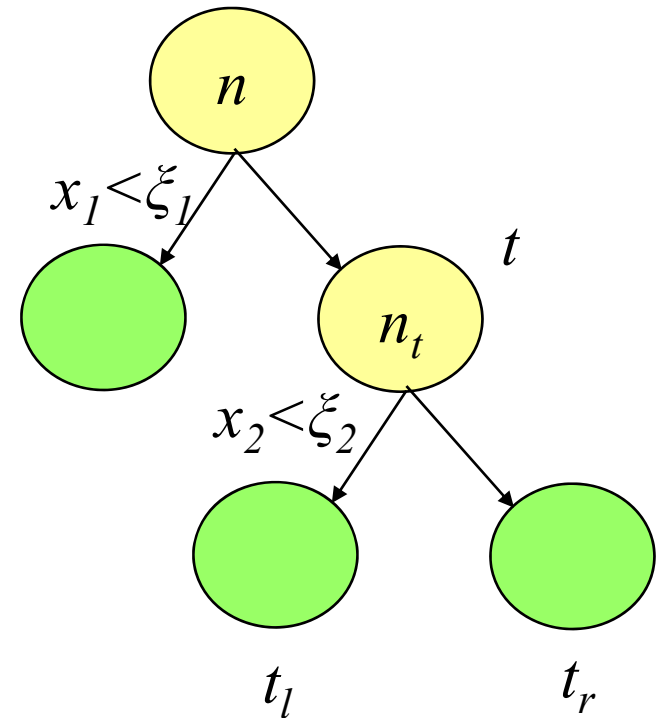
Jerome Friedman,
American

Developed 1974-1984 by 4 statistics professors: Leo Breiman (Berkeley), Jerry Friedman (Stanford), Charles Stone (Berkeley), Richard Olshen (Stanford). Distributed by Salford Systems <http://salford-systems.com/>

- Just perform Binary trees
- Unifies the categorical and continuous response under the same framework.
 - Classification tree
 - Regression tree
- Any kind of explanatory variable
- Split criterion: Impurity of the node
- Post pruning (without stop criterion)
- Delivers honest estimates of the quality of a tree

Notation

- n number of individuals.
- t a node of the T tree
- t_r the right child of t
- t_l the left child of t
- n_t individuals of node t
- n_{jt} individuals of node t belonging to the response class j
- $p(j/t)$ probability of class j in node t
- T_t subtree descendant of t

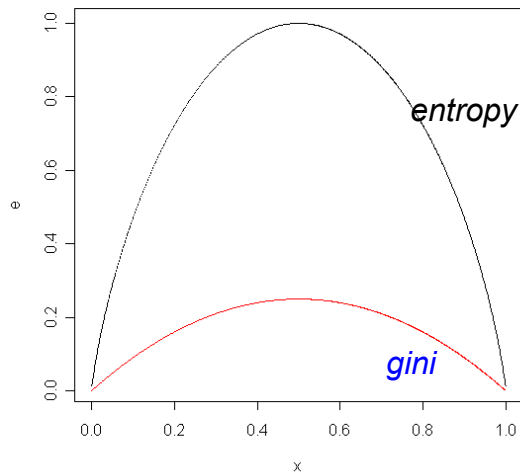


Split criterion

Measure of the quality of a node: ***the impurity***

$$i(t) = \begin{cases} \varphi(p_j / t) & j = 1, \dots, m \\ \psi(y_i / t) & i = 1, \dots, n_t \end{cases}$$

$i(t)$



$$i(t) = \max \quad \text{if} \quad p_j / t = cte.$$

$$i(t) = 0 \quad \text{if} \quad p_j / t = 0 \quad \forall j \neq k \quad p_k / t = 1$$

$$i(t) \geq \alpha i(t_r) + (1 - \alpha) i(t_l) \quad 0 \leq \alpha \leq 1$$

$p(j/t)$

Impurity of a node

For categorical responses:

- Gini

$$i(t) = \sum_{j \neq i} p(j/t)p(i/t) = 1 - \sum_j p_j^2$$

- Information (Entropy)

$$i(t) = -\sum_j p(j/t) \log_2 p(j/t)$$

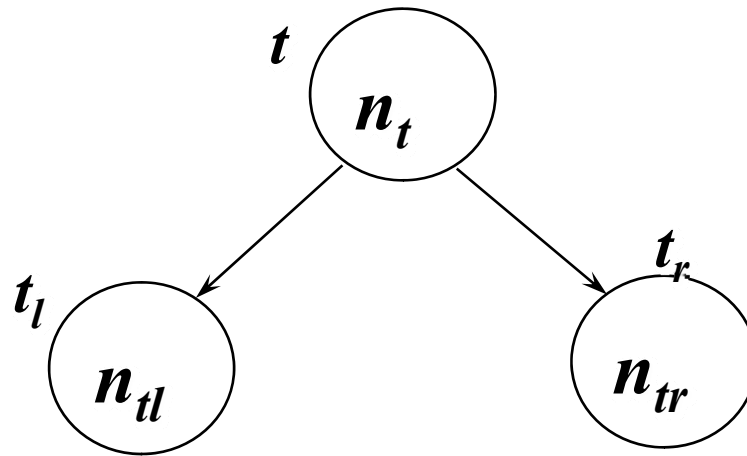
For continuous responses:

- Variance

$$i(t) = \frac{\sum_{i \in t} (y_i - \bar{y}_t)^2}{n_t}$$

Selection of the optimal partition

Maximize the decrement of impurity between the parent and its children



$$\Delta i(t) = i(t) - \frac{n_{tl}}{n_t} i(t_l) - \frac{n_{tr}}{n_t} i(t_r)$$

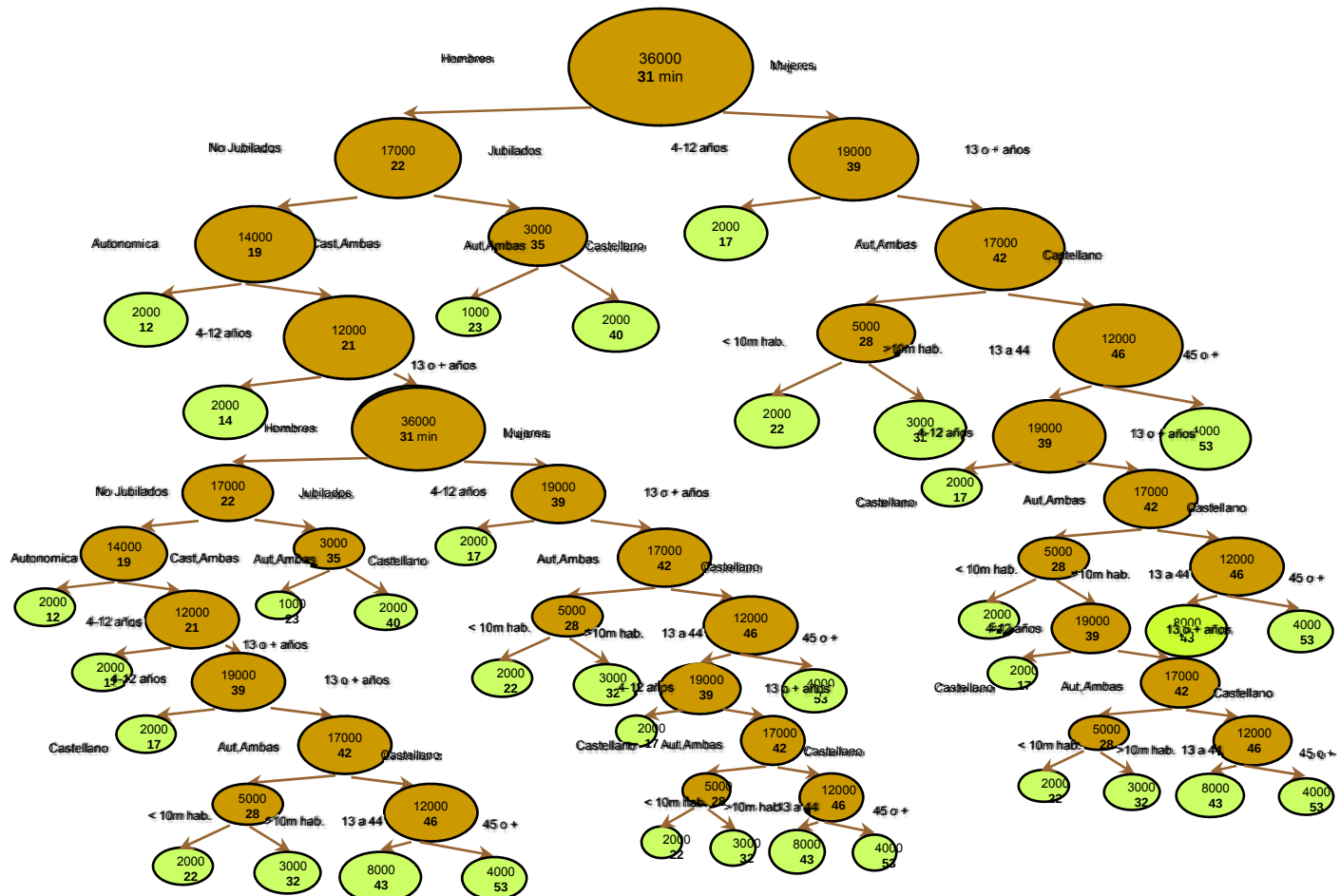
Importance of variables and dealing with missing data

- *Competing partitions:*
They are the other splits with similar decrement of impurity in a node. They can be used to rank the importance of a variables to predict the response in the whole tree.
- *Substitute partitions:*
They are splits most similar to the optimum respect to the assignment of individuals between the children. They are used as substitute in case that one individual has a missing value for the variable defining best split.

Stop criterion

There is no stop criterion. CART uses post-pruning, it builds a maximum tree and prunes the non interesting branches

Absolute
maximum tree:
Tree with all
pure leaves



Selection of the optimal tree (from the maximum tree)

We need to define how good (or bad) is a tree:

Obvious criterion = *Probability of misclassification (= cost of the tree)*

How can we compute the Cost of the tree:

We assign every leave to the response class with maximum $p(j/t)$

$$p(j/t) = \frac{p(j,t)}{p(t)} = \frac{\pi_j^{n_{jt}/n_j}}{n_t/n}$$

If classes j autorepresentatives

$$\text{if } \pi_j = \frac{n_j}{n} \quad p(j/t) = \frac{n_{jt}}{n_t}$$

Cost of the tree

of a node: $r(t) = 1 - \max_j p(j / t)$

If there are missclassification costs

$$r(t) = \min_i \sum_j c(i / j) p(j / t)$$

Cost of the tree:

(decreasing with size)

$$R(T) = \frac{\sum_{t \in \tilde{T}} p(t) r(t)}{r(\text{root})} \times 100$$

Criterion to optimize:

$$\text{Min } R(T)$$

Cost of a regression tree

- In a regression tree, every individual is assigned with the mean value of the leaf.
- Cost of the tree = residual variance

$$r(t) = \frac{1}{n_t} \sum_{i=1}^{n_t} (y_{it} - \bar{y}_t)^2$$

Penalization for complexity

Criterion to optimize:

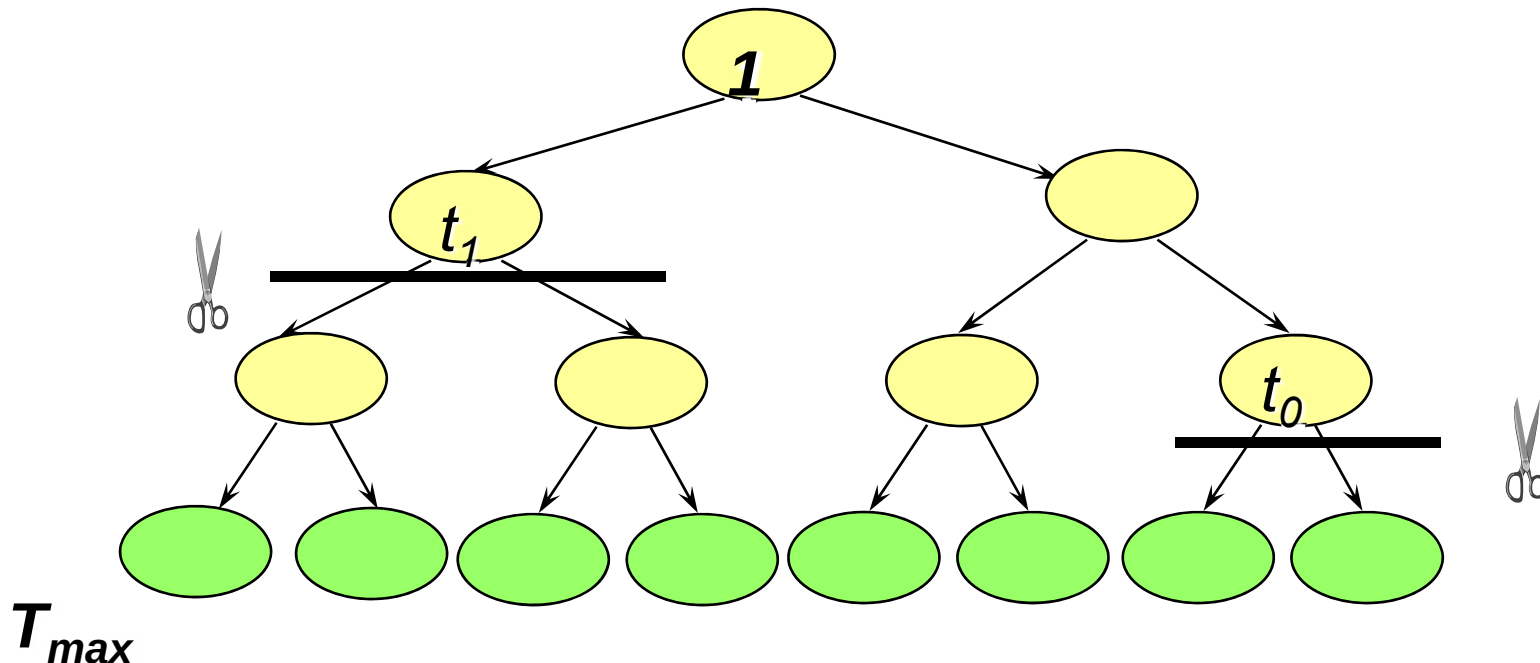
$$\text{Min} \left(R(T) + \alpha |T| \right)$$

Where α is the *complexity parameter*, expresses the penalization for building large trees

But, how can we determine α ?

We can build the trees for increasing values α , obtaining a sequence of optimal trees (of increasing size) (CART does this pruning from the maximal tree in a more intelligent way)

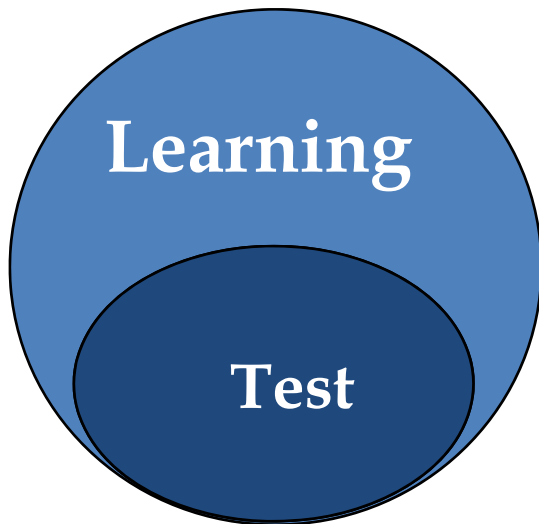
$$T_{\max}, T_{\max} - T_{t_0}, T_{\max} - T_{t_0} - T_{t_1}, \dots, 1$$



Each subtree is optimum ($\min R(T)$), within the subtrees of complexity $|T|$

Selection of the optimal tree

- To obtain a reliable tree, we need to test it with independent data from the one used in the learning phase. Hence, we divide the total data at random in one part (2/3) for training and the remaining for testing.
- We build the sequence of trees using the training data and we compute its goodness $R(T)$ using the test data



$$R^{ts}(T) = \frac{\sum_{t \in \tilde{T}} p(t) r^{ts}(t)}{r^{ts}(\text{root})} \times 100$$

Select the tree with optimal $R^{ts}(T)$ (in the test sample)

$R(T)$ train & test

Another way of putting the Cost relative to situation in the root (in this case we look for the maximum)

$$R^{ts}(T)_{rel} = \frac{r^{ts}(\text{root}) - \sum_{t \in \tilde{T}} p(t) r^{ts}(t)}{r^{ts}(\text{root})} \times 100$$

Elija el número de hojas que desea ver

Nombre del árbol: Árbol buenos compradores

Comparativa entre el árbol generado y el de test

Total: 11

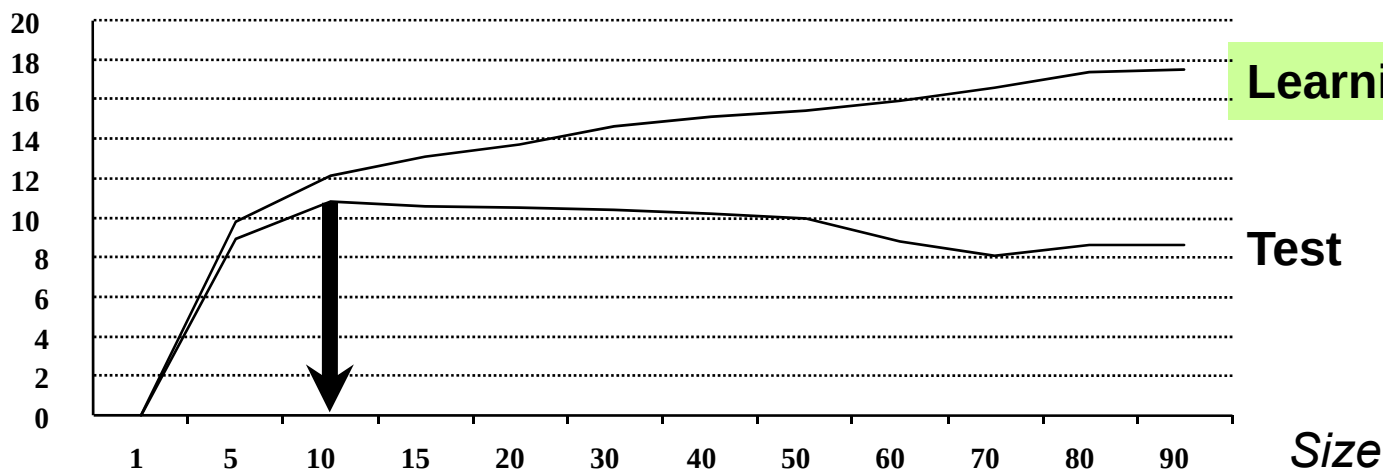
Nº Hojas	Bondad	Bondad árb...
16	50.60%	33.67%
15	50.60%	33.67%
14	50.60%	33.67%
12	50.60%	33.67%
11	46.43%	36.73%
10	42.26%	39.80%
9	38.10%	38.78%
6	33.33%	35.71%
5	31.55%	31.63%
3	18.45%	22.45%
1	0.00%	0.00%

Ver

Ponderar...

Salir

Cost



Selection of the optimal tree by crossvalidation

- We divide at random the data matrix in 10 parts (ten-fold cross validation). Lets call $n_1, n_2, \dots$ the groups formed.
- We compute the maximal tree for the following data sets formed with $n-n_1, n-n_2, \dots$ individuals.
- For a sequence of α values, we compute the cost of the corresponding trees. Then, the **cv** cost is:

$$R^{vc}(T_\alpha) = \sum_{l=1}^k \frac{\sum_{t \in \tilde{T}_\alpha} p_l(t) r_l^{vc}(t)}{r_l^{vc}(\text{root})} \times \frac{100}{k}$$

- **1se rule.** To have a more prudent tree, is possible to take, instead the minimum of $R^{cv}(T)$, the tree corresponding to the minimum value of $R^{cv}(T) + 1se$

$$\text{Min} \left(R^{cv}(T_\alpha) + 1 \times \text{desv tip}(R^{cv}(T_\alpha)) \right)$$

Presenting the results

Selected tree

Árbol: Árbol buenos compradores
Var.Resp: compradores

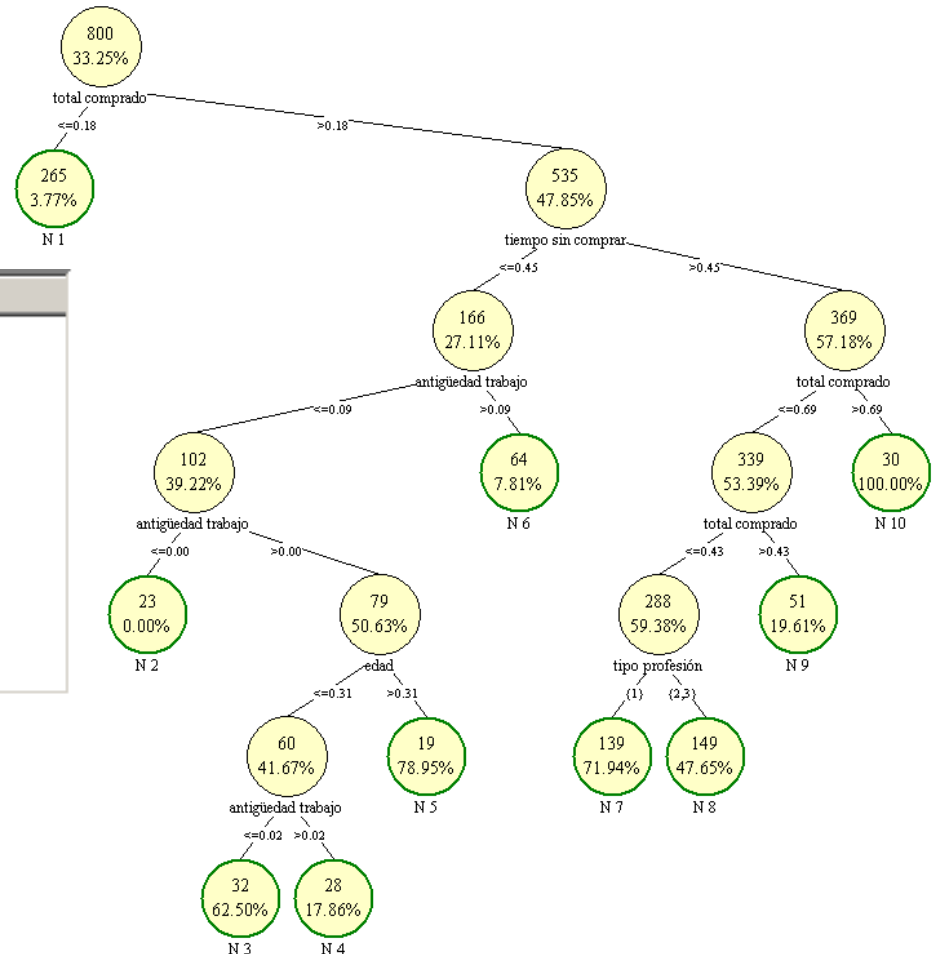


Table with ranked leaves

N	Prob.	Individuos	%Acum.	Éxitos	%Acum.	%Éxito
10	100.00	30	3.75	30	11.36	100.00
5	78.95	19	6.13	15	17.05	91.84
7	71.94	139	23.50	99	54.55	76.60
3	62.50	32	27.50	20	62.12	74.55
8	47.65	149	46.13	71	89.02	63.69
9	19.61	51	52.50	10	92.80	58.33
4	17.86	28	56.00	4	94.32	55.58
6	7.81	64	64.00	5	96.21	49.61
1	3.77	265	97.13	10	100.00	33.98
2	0.00	23	100.00	0	100.00	33.00

We rank the leaves according the predicted response

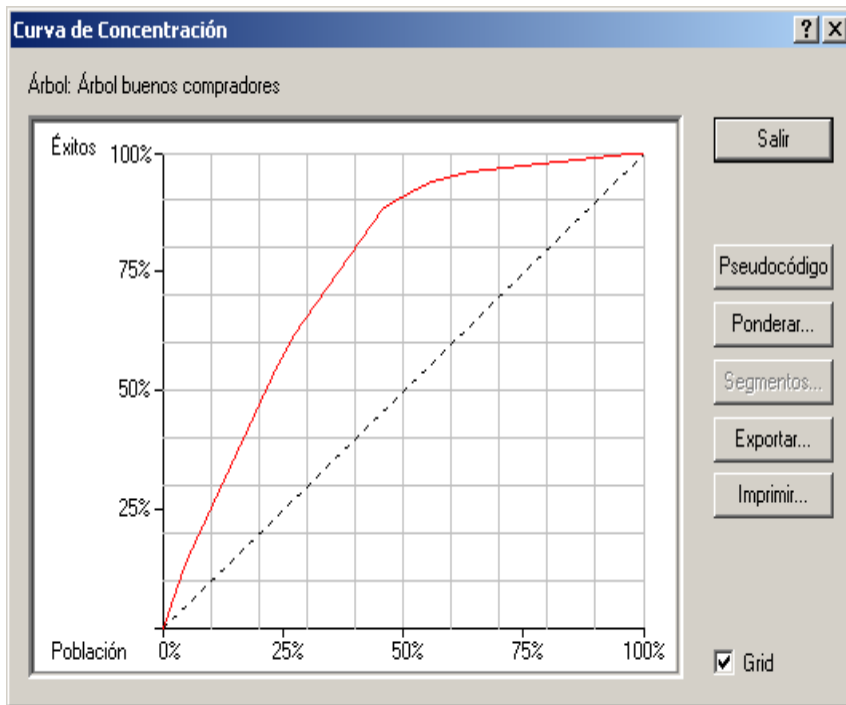
Quality assessment of the tree

Error rate

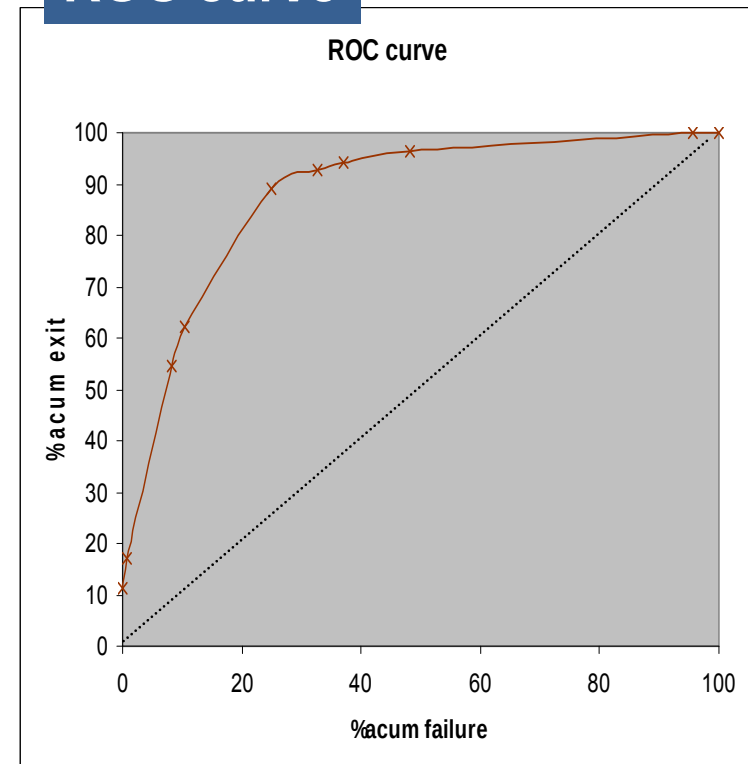
Given a threshold, compute the ratio of misclassified respect the total of instances

Global measures of quality:

Concentration curve



ROC curve



Error rate

We apply the model in the test sample, and we classify every individual according a threshold (usually = 0.5) on the outcome.

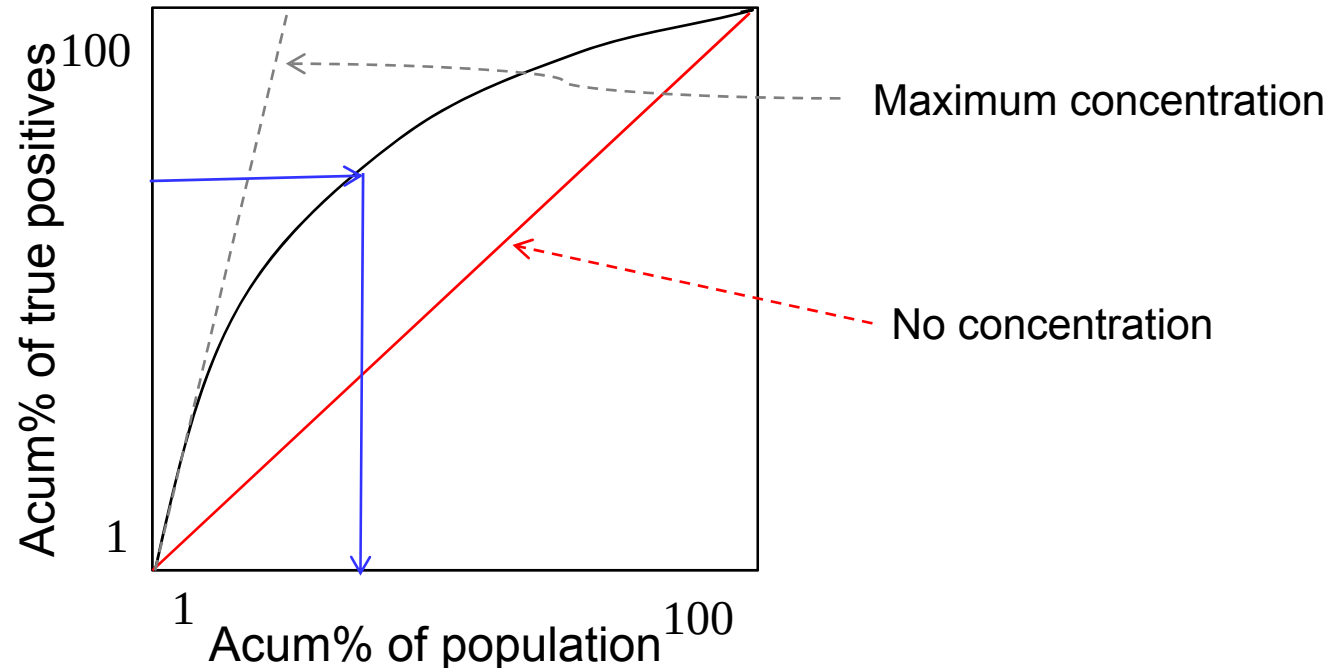
	Predicted class YES	Predicted class NO
Real class YES	n_{TP}	n_{FN}
Real class NO	n_{FP}	n_{TN}

$$Error\ rate = \frac{n_{FN} + n_{FP}}{n}$$

It depends on the chosen threshold

Concentration curves

- Represent how concentrated is the response variable in the data
 - Used in Economy
- Coordinates
 - y axis shows the accumulated percentage of true positives in the data
 - x axis shows the accumulated percentage of population



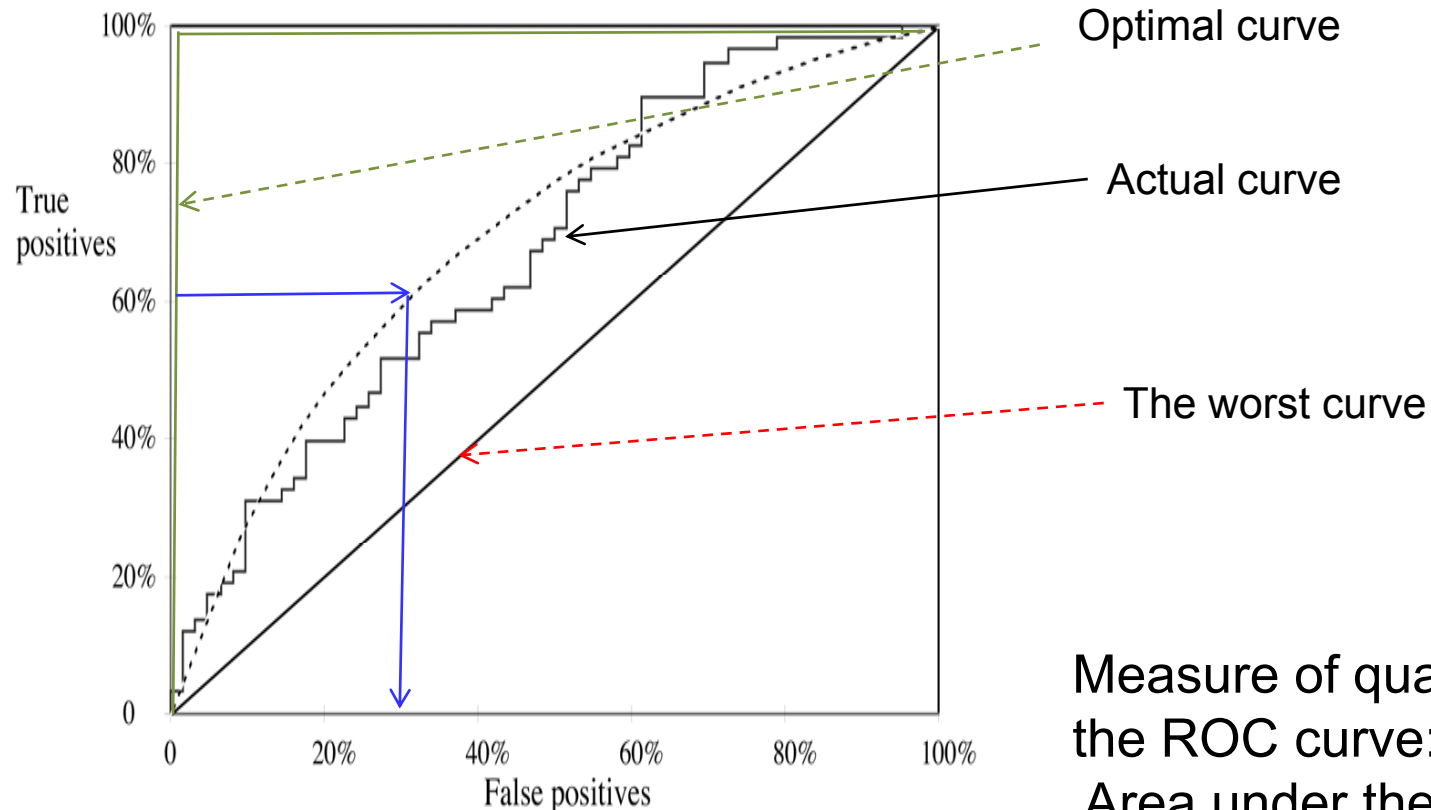
Measure of concentration:
Quotient of area under the curve and area of maximum concentration

ROC curves

- Similar to the concentration curve
 - Stands for “receiver operating characteristic”, used in signal detection to show trade off between hit rate and false alarm rate over noisy channel
- Coordinates:
 - y axis shows the accumulated percentage of true positives in the data
 - x axis shows the accumulated percentage of false positives in the data

Predicted class			
		Yes	No
Actual class	Yes	TP: True positive	FN: False negative
	No	FP: False positive	TN: True negative

a ROC curve

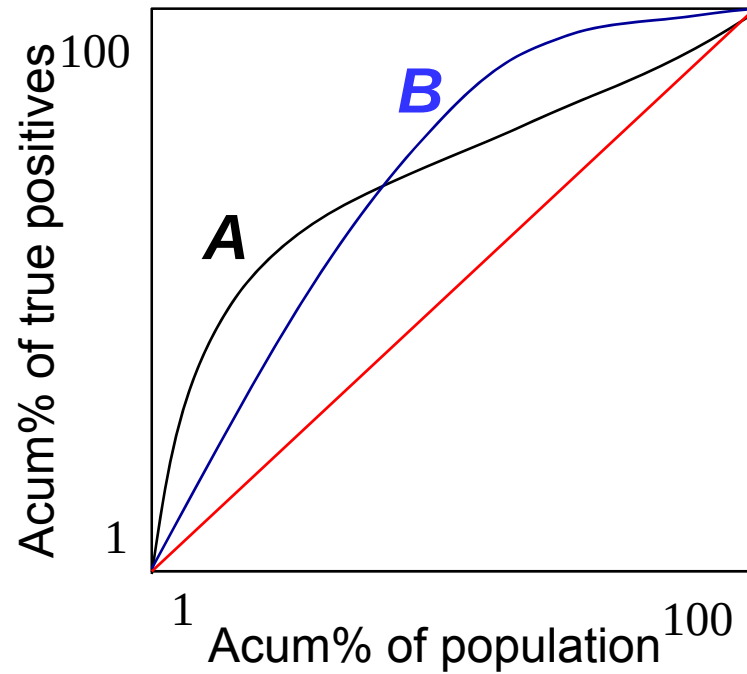


Measure of quality of
the ROC curve:
Area under the curve

The area under the ROC is equivalent to the area under the Concentration curve

$$Area\ Concentration = 2Area\ ROC - 1$$

but be aware !



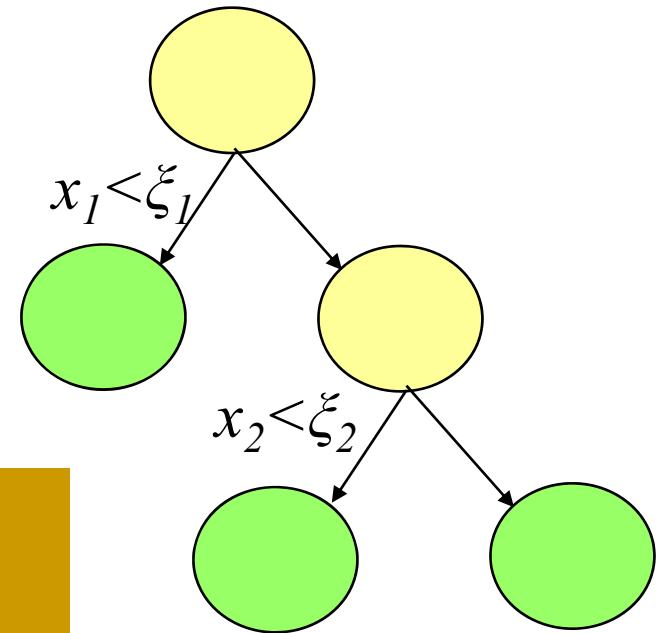
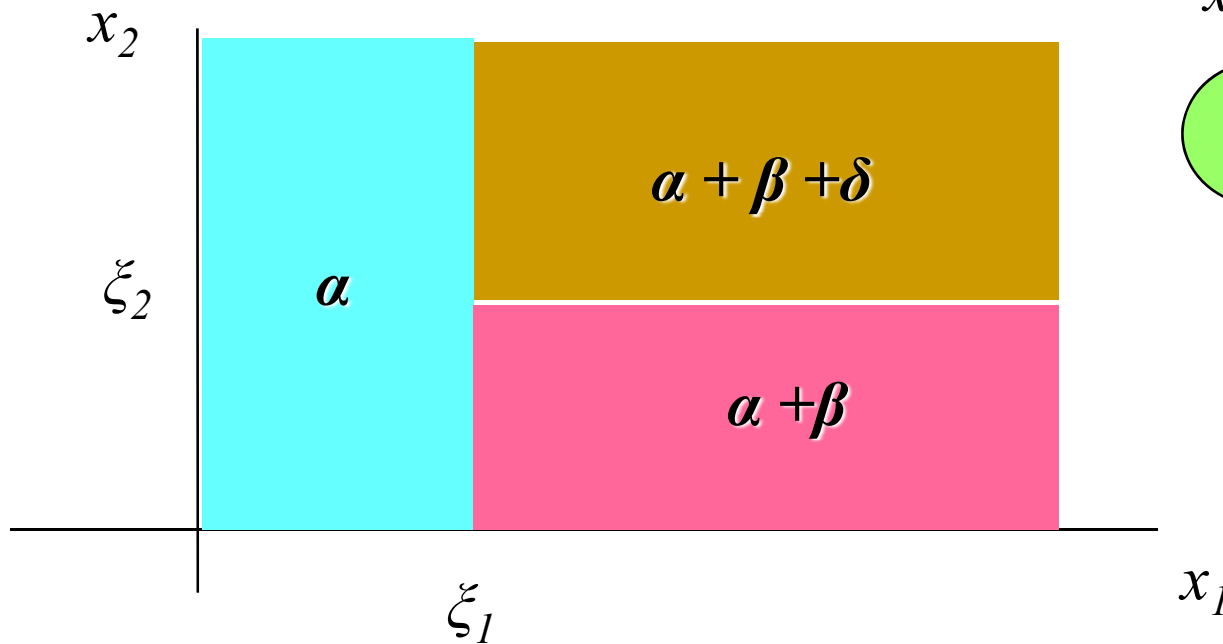
For small sample, model A is better, but for large sample model B is better

Advantages of decision trees

- Very easy to interpret .
- The branches define the assignment rules. Results are directly operational.
- The branches simulate fairly well the human process for taking decisions.
- Minimizing the preprocess, they can work up to certain level of error and missing data.
- Detect automatically complex interactions among variables (no need to specify them).
- Computationally efficient (scalability guaranteed for ...)
- But rough approximation to the response, since it approximates the response by leaps (discontinuities) (worse prediction error than other methods)

Decision trees as simple additive models

$$f(x) = \alpha + \beta I(x_1 > \xi_1) + \delta I(x_2 > \xi_2)$$



How to improve the precision

Performing predictions from consensus bootstrap resampling.

Bagging:

- Extract M bootstrap samples.

- Obtain the optimal tree for each bootstrap resample

- Predict every individual by the mean of the M trees if continuous response or by the majoritary vote if categorical response

Boosting

- Is equal to the previous, but with bootstrap resampling with probabilities not uniform, but proportional to the previous missclassification error (to force the classifier to focus on the difficult cases).

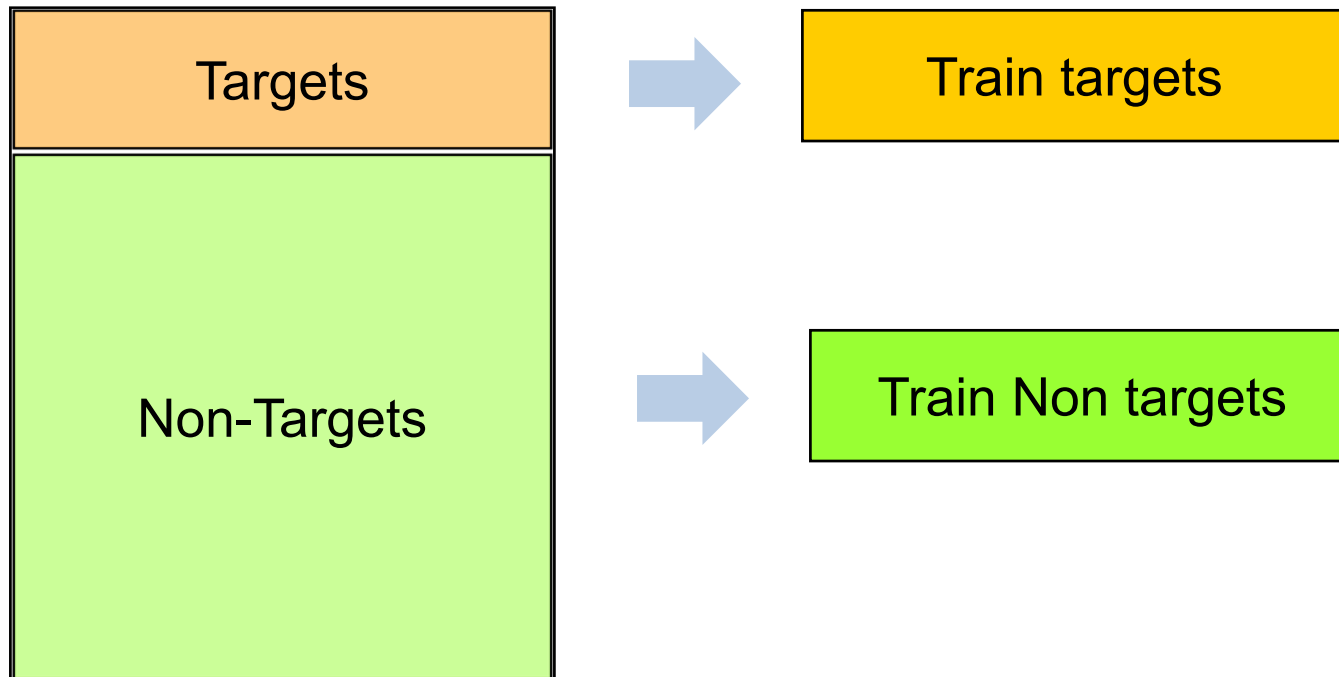
Práctica en árboles de decisión

- Inadecuación del criterio de calidad de los árboles. La asignación de un nodo a una clase de respuesta depende de las probabilidades iniciales de cada clase de respuesta.
- El poder de discriminación de un árbol debe medirse relativa a la situación inicial de partida.
- El interés del árbol es la ordenación de la población por grupos según valores crecientes de la variable de respuesta. Estos grupos somos capaces de entenderlos y localizarlos en la base de datos.

Handling unbalanced data

- Sometimes, response classes have very unequal frequency
 - Attrition prediction: 97% stay, 3% attrite (in a month)
 - Medical diagnosis: 90% healthy, 10% disease
 - eCommerce: 99% don't buy, 1% buy
 - Security: >99.99% of Catalans are not terrorists
- Similar situation with multiple classes
- Majority class classifier can be 97% correct, but useless
- CART cost is intended for balanced classes (it is hard to a minority class to become majority in a node).
- **Need to balance response classes in the training data**

Balancing the training data



Or if possible; weighing the non targets, in such a way that:

$$\sum_{\text{targets}} \text{weights} = \sum_{\text{non targets}} \text{weights}$$

Decision trees in R - CART

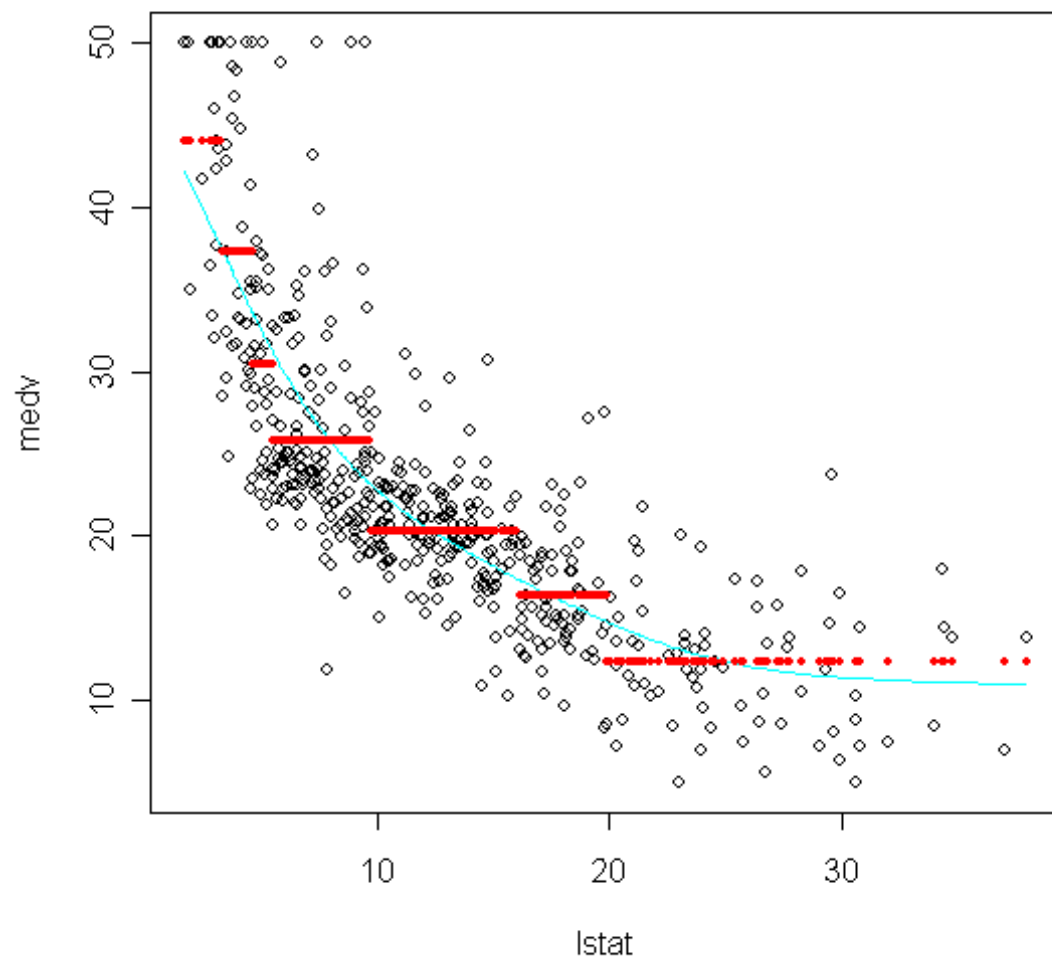
Librery *rpart*

```
p2 = rpart(dict~edat + ratfin + tiptreb, data=dd,parms=list(prior=c(.50,.50), split='gini'))
```

```
> l6=rpart(medv~lstat)
> l6p=predict(l6)
> points(lstat,l6p,col="violet",pch=20)
> l6
n= 506

node), split, n, deviance, yval
  * denotes terminal node

1) root 506 42716.3000 22.53281
 2) lstat>=9.725 294 7006.2830 17.34354
    4) lstat>=16.085 144 2699.2200 14.26181
      8) lstat>=19.9 75 1214.0820 12.32533 *
      9) lstat< 19.9 69 898.1933 16.36667 *
    5) lstat< 16.085 150 1626.6090 20.30200 *
 3) lstat< 9.725 212 16813.8200 29.72925
    6) lstat>=4.65 162 6924.4230 26.64630
      12) lstat>=5.495 134 5379.1940 25.84701 *
      13) lstat< 5.495 28 1049.9370 30.47143 *
    7) lstat< 4.65 50 3360.8940 39.71800
      14) lstat>=3.325 32 2109.5620 37.31562 *
      15) lstat< 3.325 18 738.3178 43.98889 *
```



```

> p1 = rpart(dict~edat + ratfin + tiptreb,
data=dd[learn,])
> p1
n= 406
node), split, n, deviance, yval
* denotes terminal node
1) root 406 101.490100 0.4950739
 2) ratfin>=70.5 264 62.359850 0.3825758
   4) tiptreb=altr,temp 46 4.456522 0.1086957 *
   5) tiptreb=auton,fixe 218 53.724770 0.4403670
      10) edat< 50.5 178 42.252810 0.3876404
         20) tiptreb=auton 55 10.909090 0.2727273 *
         21) tiptreb=fixe 123 30.292680 0.4390244
            42) edat< 25.5 21 3.238095 0.1904762 *
            43) edat>=25.5 102 25.490200 0.4901961 *
            11) edat>=50.5 40 8.775000 0.6750000 *
      3) ratfin< 70.5 142 29.577460 0.7042254
         6) tiptreb=altr,auton,temp 61 15.049180
            0.5573770 *
         7) tiptreb=fixe 81 12.222220 0.8148148 *
> plot(p1)
> text(p1, use.n=T)
> p1l=predict(p1,data=dd[learn,])
> p1lp=NULL
> p1lp[p1l<0.5]=0
> p1lp[p1l>=0.5]=1
> table(dict[learn],p1lp)
  p1lp
    0    1
0 150   55
1  74 127

```

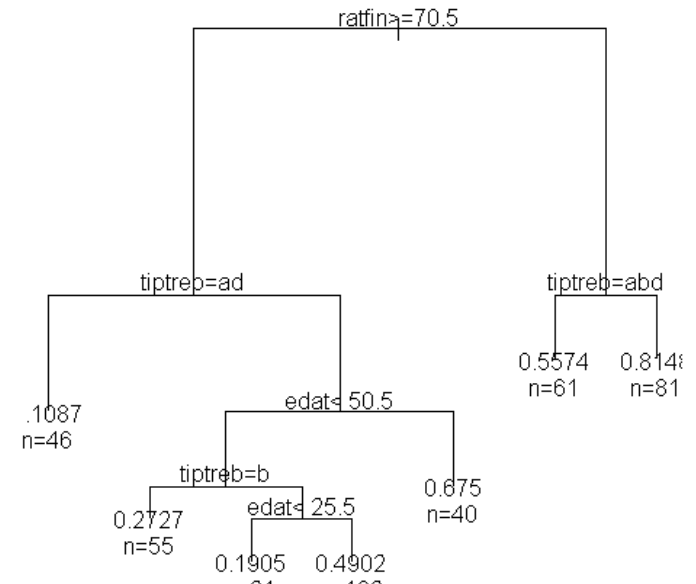
$P_{acierto}=68.2\%$

```

> p1t = predict(p1,newdata=dd[-learn,])
> p1tp=NULL
> p1tp[p1t<0.5]=0
> p1tp[p1t>=0.5]=1
> table(dict[-learn],p1tp)
  p1tp
    0    1
0  63   35
1  44   59

```

$P_{acierto}=60.7\%$



rpart *package:rpart*

Recursive Partitioning and Regression Trees

rpart(formula, data, weights na.action = na.rpart, method, parms, control, cost, ...)

formula: a formula, as in the 'lm' function.

data: an optional data frame in which to interpret the variables named in the formula

weights: optional case weights.

na.action: The default action deletes all observations for which 'y' is missing, but keeps those in which one or more predictors are missing.

method: one of "anova", "poisson", "class" or "exp". If 'method' is missing then the routine tries to make an intelligent guess.

parms=list(prior=c(.85,.15), split='information')

control=rpart.control(minsplit=20, minbucket=round(minsplit/3), cp=0.01,maxcompete=4, maxsurrogate=5, usesurrogate=2, xval=10, maxdepth=30, ...)

minsplit: the minimum number of observations that must exist in a node, in order for a split to be attempted.

minbucket: the minimum number of observations in any terminal '<leaf>' node.

cp: complexity parameter.

maxcompete: the number of competitor splits retained in the output.

maxsurrogate: the number of surrogate splits retained in the output.

usesurrogate: 1= use surrogates, in order, to split subjects missing the primary variable; if all surrogates are missing the observation is not split. 2= if all surrogates are missing, then send the observation in the majority direction.

xval: number of cross-validations

maxdepth: Set the maximum depth of any node of the final tree, with the root node counted as depth 0

C4.5 (ID3, Quinlan 1983)



Ross Quinlan
American

**** Multi-way trees ****

Node impurity: Entropy (Information)

Number of bits to predict an event of a distribution.

$$I_t = \sum_{k=1}^q p_k \log_2 p_k \quad p_k = \frac{n_{tk}}{n_t}$$

Split criterion:

Gain of information

Maximize GAIN

$$GAIN = I_{parent} - \sum_{j=1}^{m_j} \frac{n_{tj}}{n_t} I_j$$

Introduction to the Theory of Information

A random event provides information, the more when the event is less probable. We can measure this amount of information by the number of bits needed to locate the event in its distribution.

Examples.

Location of a cell in a square of 16 cells: 4 (Why?)

Transmission of a character of the alphabet (26 letters suppose + 6 punctuation)

: How much is it

The amount of information needed to locate an event is:

$$I_w = \log_2 n$$

Lets suppose that the events are grouped according a categorical variable A , with levels A_k . If we know that an event belong to A_k . Then the amount of information is:

$$I_{w \in A_k} = \log_2 n_k$$

Hence, the information of a modality A_k is:

$$I_{A_k} = I_w - I_{w \in A_k} = -\log_2 \frac{n_k}{n}$$

and on average:

$$I_A = -\sum_{k=1}^q P(A_k) \log_2 P(A_k)$$

C4.5 (ID3, Quinlan 1983)

The GAIN depends of the number of madilities of the splitting variable, thus the results depend largely of the actual coding of varaibles.

Remedy. Relativize respect the intrinsic information of the splitting varriable 8which depends on its number of modalities):

$$GAIN_{rel} = \frac{I_{parent} - \sum_{j=1}^{m_j} \frac{n_{tj}}{n_t} I_j}{Intr.Info(t)}$$

Intr.Info(t) = entropy of the splitting variable

If the variable is continuous, the algorithm detects the cutoff point where the gain is highest. If the variable is categorical, there is a partition in as many classes as children nodes have the explanatory variable.

C4.5

Criterion stop: It stops when the relative gain is below a certain threshold.

Allocation of each node to the response modality with maximum probability

Pruning the branches if the probability of misclassification in descending nodes does not improve the probability of misclassification in the current node in a significant way.