Statistical Learning and applications

B. Ghattas & G. Oppenheim

Université d'Aix-Marseille - Université Paris-Sud

badihghattas@gmail.com, georges.oppenheim@gmail.com

Outline

1. Apprentissage supervisé. Introduction à R.
2. Sélection de variables.
3. Apprentissage non supervisé.
4. Modéles graphiques et Réseaux bayésiens.
5. Deep Learning 1. Exposé sur la reconnaissance de panneaux signalisation de train. CEA.
6. Deep Learning 2.
7. Deep Learning 3. Prévision de consommation. EDF.

But de la formation

1. J'ai une réponse, est-ce que quelqu'un a une question ?
2. Les problèmes auxquels des réponses sont données.
3. Savoir faire.
4. Pourquoi ça marche ou ça marche pas ? quand ç a marche.
5. Le Deep Learning est une couche de méthodes au dessus de l'Apprentissage.

1. Le Deep Learning est une couche de méthodes au dessus de l'Apprentissage.
2. La premiere partie est l'apprentissage, la seconde porte sur questions spécifiques au DEEP. Apprendre l'apprentissage supprime bon nombre
3. Ce sont des thèmes à la mode. On gagne sa vie avec une compétence sur la data-science. J'ai des anciens étudiants/ingénieurs débauchés. C-Discount.

Dans ces approches, il y a des

1. des données (organisées, accessibles), souvent beaucoup
2. des algorithmes programmés simples ou astucieux
3. de la théorie du signal et de l'image, des probabilités, des mathématiques
4. (et beaucoup de savoir faire)
5. Un peu de tout mais surtout comprendre et savoir qu'existent des solutions

Organisation

1. Matin. Présentation ou Cours
2. Cours
3. Apres midi TD et traitement de données
4. Apres midi TD et traitement de données

Validation

1. Uniquement par le rendu des rapports de TD=TraitementsDeDonnées
2. Rendu en fin de séance.
   Noté vrai mais la note est modifiable pendant 15 jours sautant de fois que vous le souhaitez.
3. Des questions ?

Outline

Sommaire

Airquality dataset

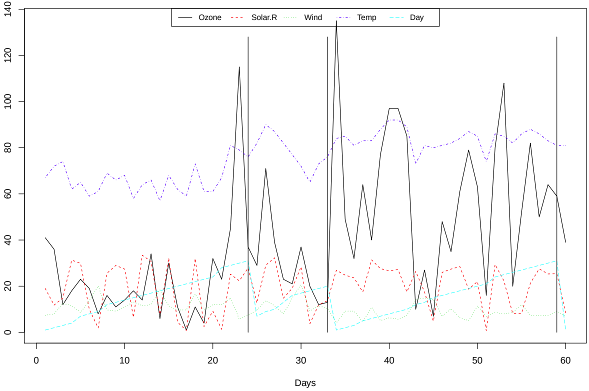
A data frame with 154 observations on 6 variables. Daily readings of the following air quality values for May 1, 1973 to September 30, 1973.

airquality.tex	Ozone	Solar.R	Wind	Temp	Month	Day
1	41	190	7.4	67	5	1
2	36	118	8.0	72	5	2
3	12	149	12.6	74	5	3
4	18	313	11.5	62	5	4
5			14.3	56	5	5
6	28		14.9	66	5	6

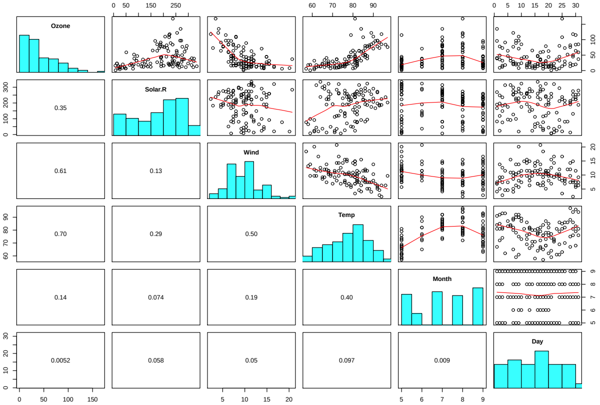
- *Ozone* Mean Ozone (ppb) from 1300 to 1500 hours at Roosevelt Island.
- *Solar.R* Solar radiation in Langleys from 0800 to 1200 hours at Central Park.
- *Wind* Average wind speed in miles per hour at 0700 and 1000 hours at La Guardia Airport.
- *Temp* Maximum daily temperature (degrees F) at La Guardia Airport.
- *Month* Month (1–12)
- *Day* Day of month (1–31)

Objective : Predict Ozone concentration (target variable) using the other variables.

Airquality Series



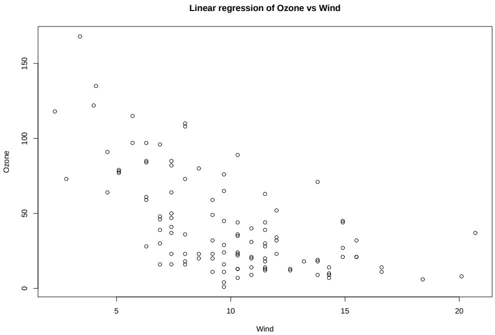
Airquality pairs



Airquality - summary statistics

Ozone	Solar.R	Wind	Temp	Month	Day
Min. : 1.00	Min. : 7.0	Min. : 1.700	Min. :56.00	Min. :5.000	Min. : 1.0
1st Qu. : 18.00	1st Qu. :115.8	1st Qu. : 7.400	1st Qu. :72.00	1st Qu. :6.000	1st Qu. : 8.0
Median : 31.50	Median :205.0	Median : 9.700	Median :79.00	Median :7.000	Median :16.0
Mean : 42.13	Mean :185.9	Mean : 9.958	Mean :77.88	Mean :6.993	Mean :15.8
3rd Qu. : 63.25	3rd Qu. :258.8	3rd Qu. :11.500	3rd Qu. :85.00	3rd Qu. :8.000	3rd Qu. :23.0
Max. :168.00	Max. :334.0	Max. :20.700	Max. :97.00	Max. :9.000	Max. :31.0
NA's :37	NA's :7				

Airquality : Linear Regression



We assume that the relation between *Wind* and *Ozone* has the following form :

$$Ozone_i = \beta_0 + \beta_1 * Wind_i + \epsilon_i$$

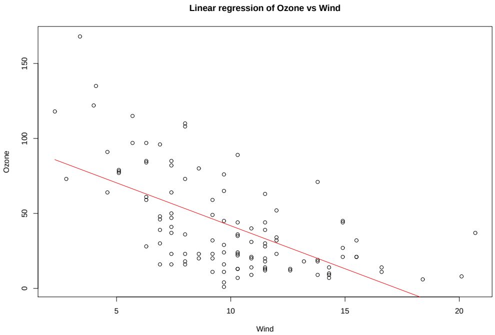
The values of the coefficients may be obtained by minimizing the Mean Squared Error (MSE) :

$$\sum_{i=1}^{n} (Ozone_i - \beta_0 + \beta_1 * Wind_i)^2$$

We get :

$$Ozone_i = 99.041 - 5.729 * Wind_i + \epsilon_i$$

Airquality : Linear Regression



We assume that the relation between *Wind* and *Ozone* has the following form :

$$Ozone_i = \beta_0 + \beta_1 * Wind_i + \epsilon_i$$

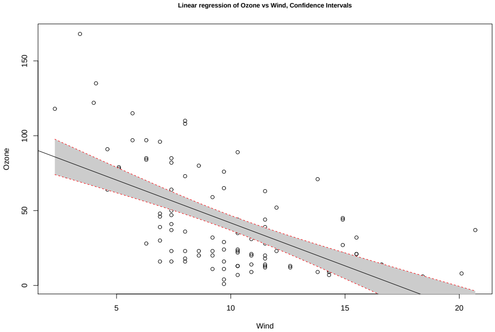
The values of the coefficients may be obtained by minimizing the Mean Squared Error (MSE) :

$$\sum_{i=1}^{n} (Ozone_i - \beta_0 + \beta_1 * Wind_i)^2$$

We get :

$$Ozone_i = 99.041 - 5.729 * Wind_i + \epsilon_i$$

Airquality : Linear Regression



We assume that the relation between *Wind* and *Ozone* has the following form :

$$Ozone_i = \beta_0 + \beta_1 * Wind_i + \epsilon_i$$

The values of the coefficients may be obtained by minimizing the Mean Squared Error (MSE) :

$$\sum_{i=1}^{n} (Ozone_i - (\beta_0 + \beta_1 * Wind_i))^2$$

We get :

$$Ozone_i = 99.041 - 5.729 * Wind_i + \epsilon_i$$

Linear Model, multiple input

$$\widehat{Y} = \widehat{\beta}_0 + \sum_{j=1}^{p} X_j \widehat{\beta}_j$$

$$RSS(\beta) = \sum_{i=1}^{N} (y_i - \mathbf{x}_i' \beta)^2$$

The minimum is achieved with :

$$\widehat{\beta} = (X'X)^{-1} X' y$$

Advantages : stable, does not need a lot of data

The Letters Example, a classification problem

The Letters Example, a classification problem

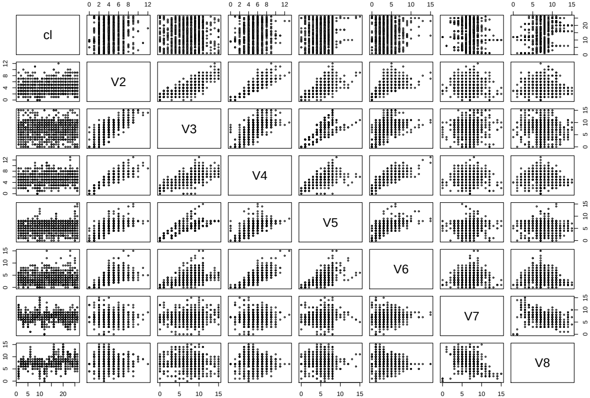
Objective : Identify each of a large number of B&W rectangular pixel displays as one of the 26 capital letters in the English alphabet. The character images were based on 20 different fonts and each letter within these 20 fonts was randomly distorted to produce a file of 20,000 unique stimuli. Each stimulus was converted into 16 primitive numerical attributes (statistical moments and edge counts) which were then scaled to fit into a range of integer values from 0 through 15.

Attribute Information :

- *letter* capital letter (26 values from A to Z)
- *x-box* horizontal position of box, *y-box* vertical position of box
- *width* of box (integer), *high* height of box
- *onpix* total # on pixels
- *x-bar* mean x of on pixels in box, *y-bar* mean y of on pixels in box
- *x2bar* mean x variance, *y2bar* mean y variance
- *xybar* mean x y correlation
- *x2ybr* mean of x * x * y, *xy2br* mean of x * y * y
- *x-ege* mean edge count left to right
- *xegvy* correlation of x-ege with y
- *y-ege* mean edge count bottom to top
- *yegvx* correlation of y-ege with x

Train on the first 16000 items and use the resulting model to predict the letter category for the remaining 4000.

Letters pairs



The logistic model

Let $Y \in A, B, ...Z$ the target variable, and $X = (X_1, ..., X_p)$ the other variables in the dataset, the explanatory variables. We may assume a model :

$$g(P[Y = a]) = \beta_0 + \sum_{j=1}^{p} \beta_j X_j$$

a is one possible level of Y, the β_j's are coefficients, and g is a *link* function. One common link is the logistic, where :

$$g(z) = log\left(\frac{z}{1 - z}\right)$$

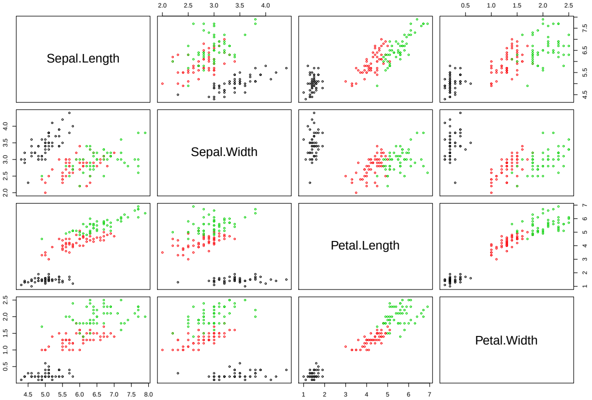
Using the data the coefficients of such a model may be estimated using the maximum likelihood approach.

Another classification exampe ; Iris

The Iris data set contains 4 explanatory variables "*Sepal.Length*", "*Sepal.Width*", "*Petal.Length*", "*Petal.Width*", "*Species*", and one target variable *Species* taking one of the three values :*Setosa*, *Virginica*, *Versicolor*.

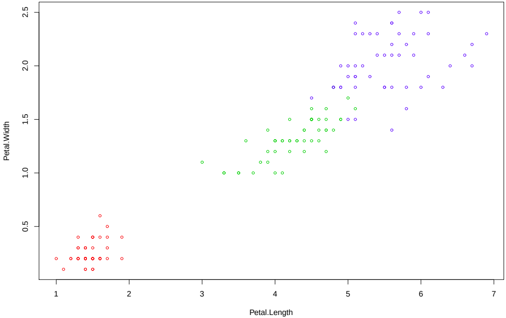
iris	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa

Iris, Scatterplots



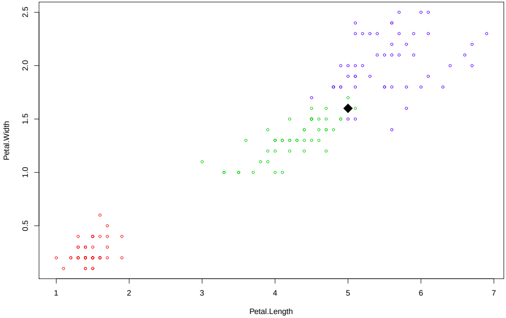
K nearest neighbors

Considering only two of the explanatory variables together with the target we may look at the data in two dimensions. Each flower is a point in R^2 having a label Y, each label presented by a color.



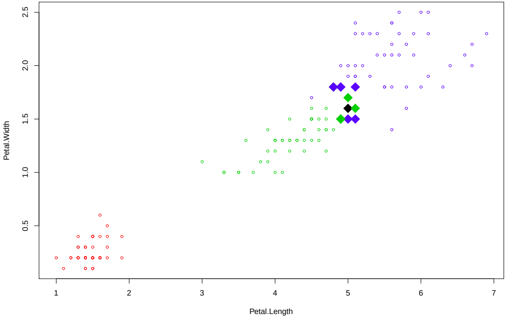
K nearest neighbors

Considering only two of the explanatory variables together with the target we may look at the data in two dimensions. Each flower is a point in R^2 having a label Y, each label presented by a color.



K nearest neighbors

Considering only two of the explanatory variables together with the target we may look at the data in two dimensions. Each flower is a point in R^2 having a label Y, each label presented by a color.



kNN

$$\widehat{Y} = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i$$

where $N_k(x)$ is the k^{th} order neighborhood of x in (X_1, X_2)
If Y is discrete binary, we apply a threshold :

$$\widehat{Y} = 1, \quad if \quad \widehat{Y} > 0.5, \quad else \quad \widehat{Y} = 1$$

Extensions

- Classification : $Y \in \{0,1\}$, $Y \in \{1,..,M\}$.
- Regression : $Y \in R$.
- $Y \in R^q$, example : spatial modeling
- $Y \in L^2(R)$, example : curves or signals.
- $X \in R^p$, when $p >> n$ like in genomics.
- $X \in L^2(R)$ as for signals.

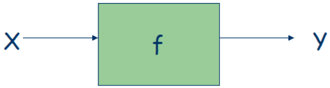
Sommaire

Statistical Learning

Supervised : Classification, Regression.

Unsupervised : Clustering, Density estimation.

Notations



We wish to estimate f using the dataset at hand

$$D_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$$

We must choose f within a class of functions, with unknown parameters.

For example : $y = f(x) = a_0 + a_1 x_1 + a_2 x_2 + a_p x_p$

$$D_n \Longrightarrow f_n(X, D_n)$$

Terminology

Input	*Output*
X	Y

- X : independent, explanatory, "predictor"
- Y : dependent, "target", "outcome"
- Both variables may be real or Multidimensional
- They may be : Quantitative, discrete (factor), ordered or not, binary or multi-class

Regression if Y is continuous.

Classification if not.

Examples

Several scientific domains, biology environment finance, industry,...

- Predict whether a patient will repeat a heart attack
- Predict stock prices within 6 months using economic parameters and the performance of an enterprise
- Identify handwritten digits of postal codes on envelops
- Estimate the glucose level of a diabetic using the infrared spectra of blood absorption.
- Identify the prostate cancer risk using clinical and demographical parameters.

Learning is essential in statistics, Data Mining and Artificial Intelligence.
We Learn from data ! !

BioInformatics

Transcriptom : Expression of N genes in p different cells

- Can we define homogenous classes of genes ? The classification must be validated by an expert.
- Some cells are cancerous others not.
  - Can we assess their status from the transcriptomic data ?
  - If yes, which are the most important genes for making the decision rule ?
  - What is the precision of the decision rule ?
  - What about its complexity ?

Regression criterion

$$X \in R^p, \quad Y \in R$$

We look for a function f to predict Y, using the entry X. We must define a risk function, $L(Y, f(X))$, and choose the function f which minimizes the risk.
For regression the quadratic risk is often used the criterion minimized using the sample at hand is :

$$MSE(f) = \frac{1}{n} \sum_{i=1}^{n} \left((Y_i - f(X_i))^2 \right)$$

Classification

In practice once the shape of f is chosen, and a data set at hand we loof for the f minimizing the empirical misclassification error :

$$\hat{L}(f) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{f(X_i)\neq Y_i}$$

The best classifier would be :

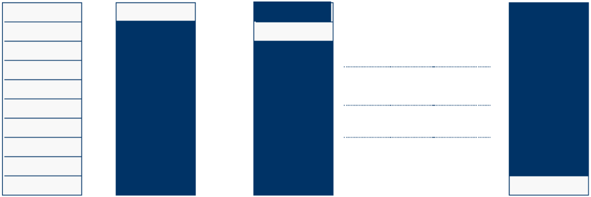
$$f^*(x) = argmin_f P(f(X) \neq Y) \ , \quad L^* = L(f^*)$$

Choosing the shape or class of f

Choose f, within a class C, which minimizes $\widehat{L}_n$.
The performance of the classifier are guaranteed to be close to the best classifier of the class if the complexity of C is controlled.
the Choice of C depends on :

- The nature of the problem we model
- Hypotheses and experiences issuing the data
- Experts opinion
- A question of mode ?

Cross Validation



$$\widehat{L}^{cv} = \frac{1}{K} \sum_{1}^{k} \widehat{L}_k$$

where $\widehat{L}_k$ is the loss for the k^{th} test sample

Some Methods

- K Nearest Neighbours
- CART
- Aggregating classifiers
- Neural Networks
- Support Vector Machines
- Bayesian Networks

Sommaire

The problem

data : $(\mathbf{X}, Y) \in R^p \times C$
$\mathbf{X}$ predictor, attributes, features
$Y \in C$ output to predict. $C = R$ or $C = \{1, ..., J\}$.
Objective :
Using the observations (X_i, Y_i) from D, construct a classifier $\widehat{f}(\mathbf{X})$ having a low generalization error :

$$R(\widehat{f}) = \frac{1}{n} \sum_{i=1}^{n} \left(L(y_i, \widehat{f}(\mathbf{x_i})) \right)$$

where L is a loss function.
L is the quadratic error in regression, or the misclassification rate in classification.

The model

Search for a partition of the space X and assign a value of Y to each class of the partition.

In regression :

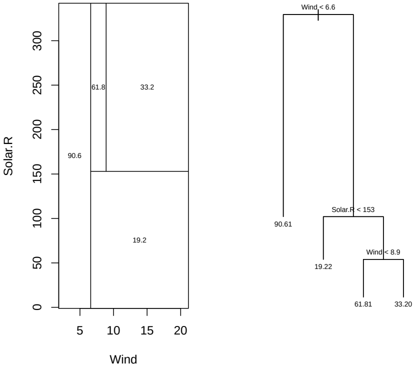
$$f(\mathbf{x}) = \sum_{j=1}^{q} c_j \mathbf{1}_{N_j}(\mathbf{x})$$

$$\widehat{c}_j = \frac{1}{Card\{i; \mathbf{x}_i \in N_j\}} \sum_{i;\mathbf{x}_i \in N_j} Y_i$$

In Classification : Y discrete having J levels

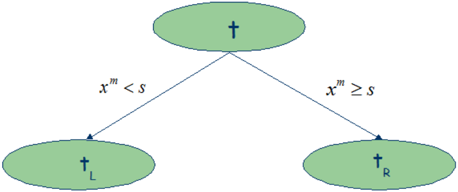
$$\widehat{c}_j = \text{The most frequent class in } N_j(\mathbf{x})$$

Example


Solar.R
0
50
100
150
200
250
300
5
10
15
20
Wind
90.6
61.8
33.2
19.2
Wind < 6.6
Solar.R < 153
Wind < 8.9
90.61
19.22
61.81
33.20

2 stages : Maximal Tree and Pruning

All the observations are in the root node.



Splitting rule : one variable and a threshold. How to do ?
Use the deviance to measure the heterogeneity of a node :

$$R(t) = \sum_{x_n \in t} (y_n - \bar{y}(t))^2$$

Optimal Splits : minimize the children's deviance

Minimize total new nodes Heterogeneity. Let s be a split of the form : $x^m < a$,

$$\Delta R\,(s,t) = R(t) - (R(t_L) + R(t_R)) \geq 0$$

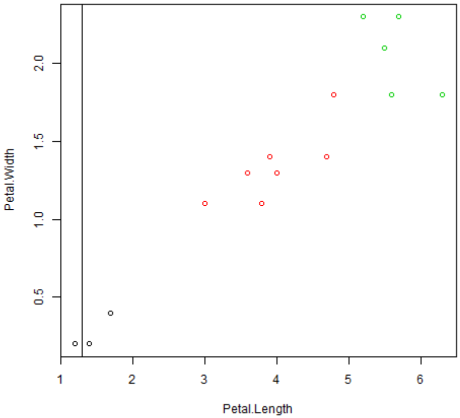
$$\Delta R\,(s,t) = max_{s \in \Sigma} \Delta R\,(s,t)$$

In classification,

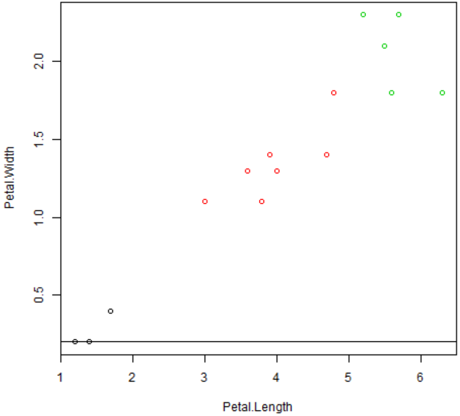
$$R(t) = -\sum_{j \in J} p_j(t) log(p_j(t))$$

where $p_j(t)$ prior probability for each class j in t.

Iris data set : search in first direction



Iris data set : search in second direction



The model

Split the root t into two children t_L et t_R Do the same recursively.
Stop when at least one of the following conditions is satified :

- very few observations in a node, *minsize*
- $\Delta R(s, t)$ is lower than a fixed threshold, *mindev*

The maximal tree :

- has low errors over learning sample
- is poor over test samples
- is too big, thus unreadable

Penalized deviance for Pruning

Tree's deviance :

$$R(T) = \frac{1}{N} \sum_{t \in \tilde{T}} R(t)$$

Penalised deviance :

$$R_\alpha(T) = \frac{1}{N} \sum_{t \in \tilde{T}} R(t) + \alpha |\tilde{T}|$$

For a subtree pruned at node t,

$$R_\alpha(T_t) = \sum_{t \in \tilde{T}_t} R(t) + \alpha |\tilde{T}_t|$$

and,

$$R_\alpha(t) = R(t) + \alpha$$

Pruning 2

Pruning takes off *weak* branches successively resulting in a sequence of embedded decreasing trees :

$$T_1 \geq T_2 \geq ... \geq T_K$$

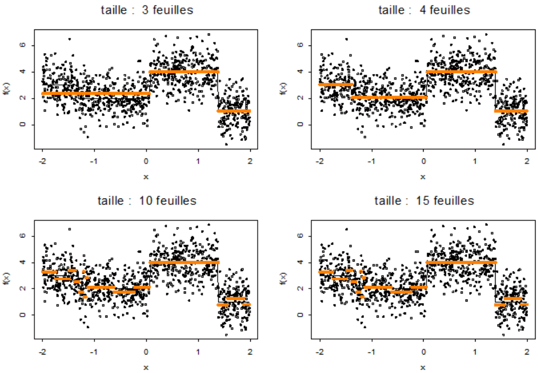
Each tree in this sequence is the optimal subtree of T_{max} with respect to its size. We have to select one tree among this sequence. It is based on the deviance estimate of each tree in the sequence. Suppose the data set S is randomly partitioned

$$S = S^{train} \cup S^{test}$$

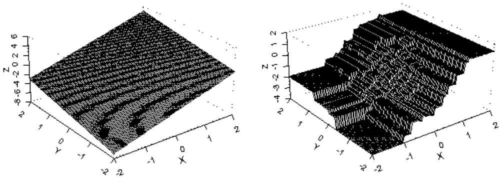
One may train the sequence using S^{train} and select the best tree estimating the deviance over S^{test}

$$\hat{R}^{test}(T) = \frac{1}{|S^{train}|} \sum \hat{R}^{test}(t)$$

Simulation p=1



Simulation p=2

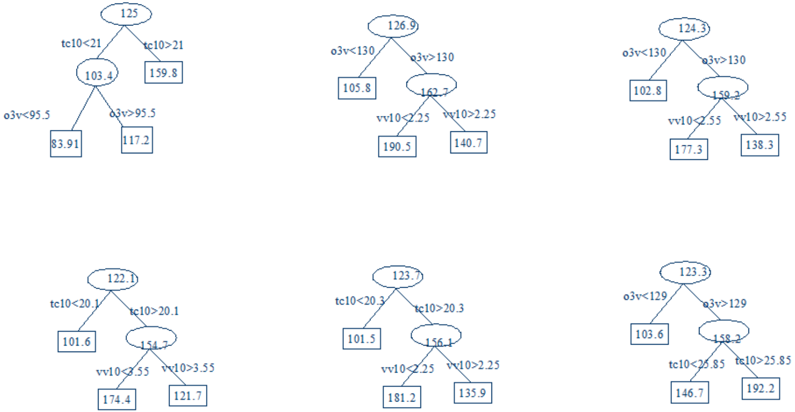


Advantages and drawbacks

- Working in high dimension
- Variables of different natures
- Regression - Classification
- Model easy to interpret
- Interactions between variables used
- Dealing with missing data
- Variables importance
- Many extensions possible

Drawback : Instability

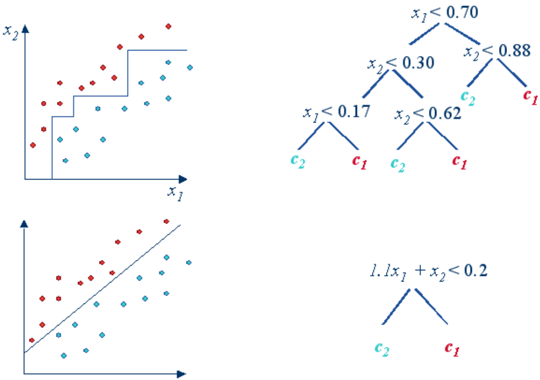
Instability



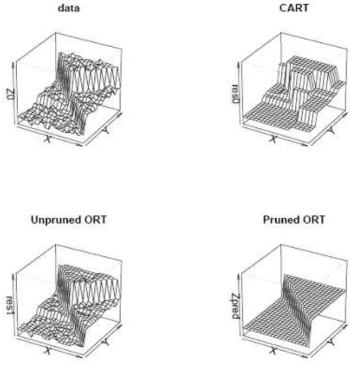
Extensions

- Oblique CART, OC1.
- Multivariate CART in regression and classification
- Multiple Regression within each node.
- Bayesian Cart.
- Bootstrap within nodes (Direct approach of instability).

Extensions


x_2
x_1
$x_1 < 0.70$
$x_2 < 0.30$
$x_2 < 0.88$
$x_1 < 0.17$
$x_2 < 0.62$
c_2
c_1
c_2
c_1
c_2
c_1
$1.1x_1 + x_2 < 0.2$
c_2
c_1

Oblique Regression Trees



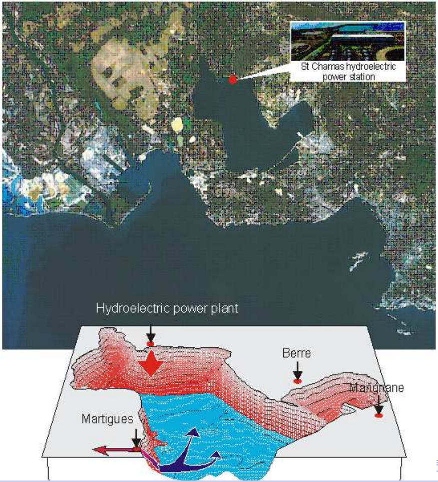
Multidimensional or functional output

- Predict a vector and/or a functional. $Y \in R^d$, or $Y \in L^2(R)$
- Predict the daily ozone profile
- Predict the size distribution of zooplanktons (indicator of changes in climate)
- Predict the profiles of sea salinity

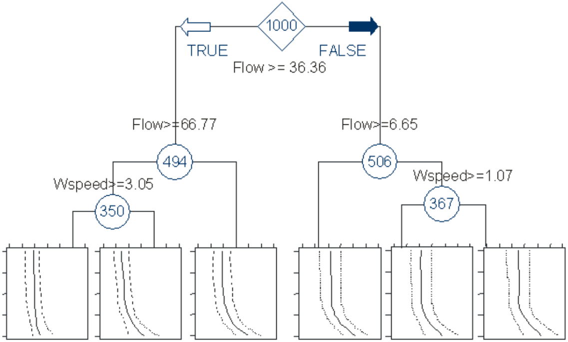
The regression function has the form :

$$f(x) = E[Y|X = x] = \sum_{j=1}^{q} f_j I(X \in N_j)$$

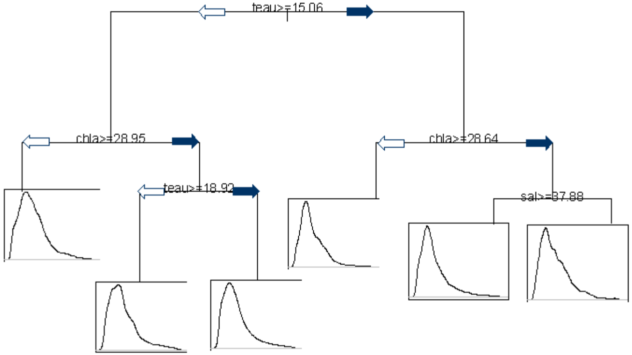
Examples


St Chamas hydroelectric power station
Hydroelectric power plant
Berre
Marignane
Martigues

Predicting Salinity profiles



Modeling zooplankton sizes' densities



How is it done

Main difficulty : generalize the univariate criterion :

$$R(t) = \sum_{x_n \in t} (y_n - \bar{y}(t))^2$$

What if y_n are no more scalars, but vectors ?
A Natural idea

$$R(t) = \sum_{x_n \in t} ||y_n - \bar{y}(t)||^2$$

Where we must define the norm constrained to the property :

$$\Delta R\,(s, t) = R(t) - (R(t_L) - R(t_R)) \geq 0$$

Multivariate case

- When Y is a vector $Y \in R^d$, if the d components are independent, we can use the Euclidian norm.
- If not, we transform the data Y by projection onto an orthogonal basis where the Euclidian norm may be used.

Sommaire

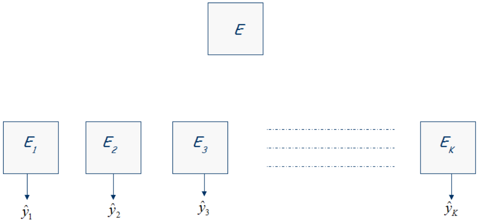
Why might we agregate

- Instabilty ?
- Multiple models ?
- Boost ?

Bagging, Boosting, ...

- Freund : Weak Learner $\Rightarrow$ Strong learner
  (vote within several "learners"),
  "Boosting" (1995)
- Breiman : Unstable "Classifier" $\Rightarrow$ Stable (by bootstrap aggregation)
  "Bagging" (1996), "Arcing" (1999)

Aggregation



In Regression,

$$f^{(a)}(x) = \frac{1}{K}\sum_{k=1}^{K}\widehat{f}_k(x)$$

In Classification,

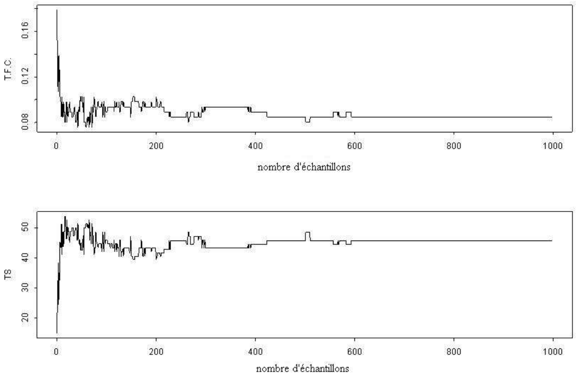
$$f^{(a)}(x) = Argmax_j \sum_{k=1}^{K}\mathbf{1}_{\widehat{f}_k(x)=j}$$

Example : Ozone prediction

Station		MSE^{CART}	MSE^{BAG}	TS^{CART}	TS^{BAG}
VTRL	Moyenne	905.6	561.8	57.8	67.7
	Ecart-type	112.2	91.4	5.7	4.4
	Gain (%)		**38**		-17.1
RBRT	Moyenne	831.5	522.6	49.8	60.2
	Ecart-type	96.7	73.6	6	4.9
	Gain (%)		**37.1**		-20.9
ROUSS	Moyenne	702.5	468.7	58.3	66.2
	Ecart-type	80.3	64	4.6	4.1
	Gain (%)		**33.3**		-13.5
SSLP	Moyenne	721.5	482.1	40.6	52.4
	Ecart-type	81	71	5.8	5.6
	Gain (%)		**33.2**		-29
PDBC	Moyenne	661.6	455.9	55.8	63.5
	Ecart-type	98.3	67.5	4.8	4.3
	Gain (%)		**31.1**		-13.7

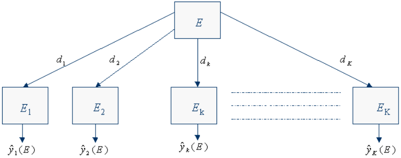
TABLE – MSE : Mean Squared Error, TS=Threat Score

Number of bootstrap samples



Boosting

$Y \in \{0, 1\}$



$$\epsilon_k = \sum_{i=1}^{n} d_k(i)|\widehat{y}_k(i) - y_i|, \quad \beta_k = \frac{1-\epsilon_k}{\epsilon_k}, \quad w_k = log(\beta_k)$$

$$d_{k+1}(i) = d_k(i)\beta_k^{|\widehat{y}_k(i)-y_i|}$$

$$\widehat{y}^a(\mathbf{x}) = 1, \quad if \quad \sum_{k;\widehat{y}_k(\mathbf{x})=1} w_k \geq \sum_{k;\widehat{y}_k(\mathbf{x})=0} w_k$$

Example-Breast Cancer

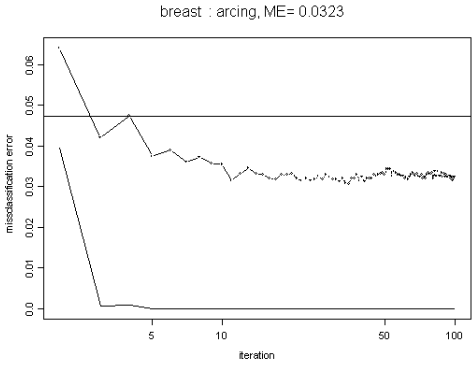


FIGURE – Learning and Test errors of the boosted classifier

Datasets from ML Benchmark

Simulated	name	Variables	Observations	levels
	waveform	22	5000	3
	Ringnorm	21	7400	2
Real				
	Iono	35	351	2
	Glass	10	214	6
	Breast	10	683 (+16)	2
	DNA	61	3190	3
	Vowel	11	990	11

Random Forests

- Construct bootstrap samples of the data
- Leave the OOB sample aside
- For each node of the tree, select the optimal split searching over only $log(p)$ variables among the p ones, selected randomly.
- Don't prune the tree
- Aggregate the trees like in bagging
- Random Features : random linear combination of the selected variables at each node

RF Properties

- Each tree has a low bias (but high variance)
- Trees are not correlated
- The correlation is defined to be the one computed between trees' predictions over OOB samples.
- Very high performences, "Best of the chelf classifier"
- Computational complexity reduced
- Possible parallelization

Variables importance in RF

- Set $N_i = 0$, $M_i = 0$ et $M_{ij} = 0$, for $i = 1..N$ et $j = 1..p$
- N_i = Number of times observation i appears in a OOB.
- M_i = Number of times observation i appears in a OOB and is misclassified
- M_{ij} = Number of times observation i appears in a OOB and is misclassified after permutation of the values of variable j in the OOB.
- For variables $j = 1, p$, For each tree $k = 1, K$ in the forest

  If observation i is in OOB_k, $N_i = N_i + 1$

  If observation i is in OOB_k and misclassified, $M_i = M_i + 1$

  Perturb randomly the values of variable j in OOB_k. If observation i is in OOB_k and is misclassified after permutation, $M_{ij} = M_{ij} + 1$
- Importance of variable j is $= \frac{1}{n} \sum_i Z_i(j)$ where $Z_i(j) = \frac{(M_{ij} - M_i)}{N_i}$.

Variables importance- Comments

- Insensitive to the nature of the resampling used (bootstrap samples with or without replacement).
- Stable in presence of correlations between variables.
- Invariant to normalization (using standard deviation of $Z_i(j)$)
- Stable w.r.t. data perturbations. Bootstrapping VI is unnecessary.

Multi Class direct generalisations - Some principles

- The base classifier chooses a set of *plausible* classes for each example
- Use the *pseudoloss* error, which penalizes the week hypothesis who failed in :
  - Not including the right label
  - Including a wrong label
- In final : Choose the most appearing label among the set of plausible labels predicted

Example of a multiclass algorithm : Hastie 2007

$Y \in \{0,1\}$	$Y \in \{1,..,J\}$
$\epsilon_k = \sum_{i=1}^n d_k(i)\lvert y_i - \widehat{y}_k(i)\rvert$	$\epsilon_k = \sum_{i=1}^n d_k(i)\mathbf{1}_{\lvert y_i \neq \widehat{y}_k(i)\rvert}$
$\beta_k = \frac{1-\epsilon_k}{\epsilon_k}$	$\beta_k = (J-1)\frac{1-\epsilon_k}{\epsilon_k}$
$d_{k+1}(i) = d_k(i)\beta_k^{\lvert y_i - \widehat{y}_k(i)\rvert}$	$d_{k+1}(i) = d_k(i)\beta_k^{\mathbf{1}_{\lvert y_i \neq \widehat{y}_k(i)\rvert}}$
$w_k = log(\beta_k)$	$w_k = log(\beta_k)$
$\widehat{y}^a(\mathbf{x}) = \mathbf{1}_{\sum_{k;y_k(\mathbf{x})=1} w_k \geq \sum_{k;y_k(\mathbf{x})=0} w_k}$	$\widehat{y}^a(\mathbf{x}) = Argmax_j\{\sum_{k;y_k(\mathbf{x})=j} w_k\}$

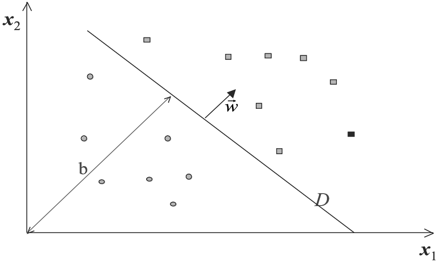
Sommaire

Linear Separation

$\mathcal{S}$= n *i.i.d.* sample of $(\mathcal{X}, \mathcal{Y}) \subseteq (\mathbb{R}^p, \{-1, +1\})$

$$\mathcal{S} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \subseteq (\mathcal{X} \times \mathcal{Y})^n .$$

We look for a function : $f(x) = sign(\langle w.x \rangle + b)$



The Perceptron, primal form

$$
\begin{array}{l}
\eta > 0 \\
\mathbf{w}_0 = 0; \quad b_0 = 0; \quad k = 0 \\
R = max_{1 \leq i \leq n} ||\mathbf{x}_i|| \\
\textit{repeat} \\
\qquad \textit{for } i = 1..n \\
\qquad\quad \textit{If} \quad y_i(\langle \mathbf{w}^{(k)}, \mathbf{x}_i \rangle + b^{(k)}) \leq 0 \quad \textit{then} \\
\qquad\qquad \mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \eta y_i \mathbf{x}_i \\
\qquad\qquad b^{(k+1)} = b^{(k)} + \eta y_i R^2 \\
\qquad\qquad k = k + 1 \\
\text{While there are errors in the internal loop}
\end{array}
$$

k is the number of errors.

Remark : The output has the form : $\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$

Observations and sample margins

- The margin of an observation is : $\gamma_i = y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b)$
- It is positive for a well classified observation
- We are interested by the observations margin distribution.
- The margin of a hyperplane w.r.t. to S is the minimum of this distribution.
- The margin of a sample is the maximum margin over all the hyperplanes.
- The hyperplane which achieves this maximum is the maximum margin hyperplane. If $||w|| = 1$, the margin is the geometrical distance to the plan.

The Optimization problem

Find $(w, b) \in \mathbb{R}^p \times \mathbb{R}$ such that :

$$\begin{array}{ll} \textit{Minimize}_{w,b} & \frac{\|w\|^2}{2} \\ \textit{Under} & y_i(\langle w.x_i\rangle + b) \geqslant 1 \forall i \in \{1,\dots,n\} \end{array}$$

Solution :

$$w^* = \sum_{i=1}^{n} \alpha_i^* y_i x_i = \sum_{i \in sv} \alpha_i^* y_i x_i.$$

and

$$b^* = -\frac{\max_{y_i=-1}\left(\langle w^*.x_i\rangle\right) + \min_{y_i=+1}\left(\langle w^*.x_i\rangle\right)}{2}.$$

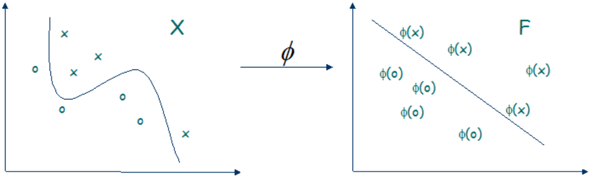
where $\alpha^* = (\alpha_1^*, \dots, \alpha_n^*)$ are the Langrangian coefficients and $sv = \{i \in \{1,\dots,n\}\,;\, \alpha_i^* \neq 0\}$.
The decision function is :

$$\widehat{f(x)}_n = sign\left(\sum_{i \in sv} \alpha_i^* y_i \langle x_i.x\rangle + b^*\right)$$

Non linear separation

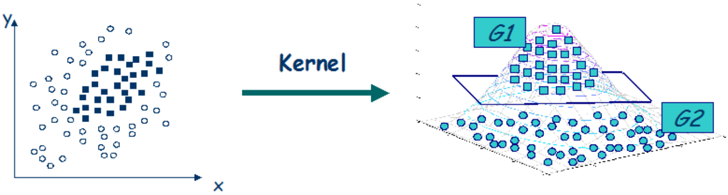
$$\begin{aligned} \phi : \quad & \mathcal{X} \rightarrow \mathcal{F} \\ & \mathbf{x} \rightarrow \phi(\mathbf{x}) \end{aligned}$$

$\mathcal{X}$ is the "attribute space" and $\mathcal{F}$ "the *feature space*"



If $Q < q$, dimension reduction, example PCA.
ϕ is non linear in general. A linear separator is learned in $\mathcal{F}$.

Non linear separation 2



Kernels and linear separation in $\mathcal{F}$

$$f(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b = \left\langle \sum_{i=1}^{n} \alpha_i y_i \phi(\mathbf{x}_i), \phi(\mathbf{x}) \right\rangle + b$$

$$= \sum_{i=1}^{n} \alpha_i y_i \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}) \rangle + b$$

For all x and z in $\mathcal{X}$, define $K(x,z) = \langle \phi(x), \phi(z) \rangle$.
If we know K we do not need to know ϕ to compute f, as :

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$$

We use Kernels associated to non linear transformations. For example :

$$K(x,z) = \langle x, z \rangle^2 = \sum_{(i,j)=(1,1)}^{(n,n)} x_i x_j z_i z_j, \quad \phi(x) = (x_i x_j)_{(i,j)=(1,1)}^{(n,n)}$$

More generally, polynomial kernels have the fom :

$$K(x,z) = (\langle x, z \rangle + c)^d$$

Kernel Properties

A kernel K must check for the following properties inherited from the dot product :

$$K(x,z) = K(z,x), \quad K(x,z)^2 \leq K(x,x)K(z,z)$$

For a finite $\mathcal{F}$, a symmetric K is a Kernel if and only if the matrix $\mathbf{K} = K(\mathbf{x}_i, \mathbf{z}_j)_{i,j=1}^{n}$ is semi-finite positive (eigen values non negative).
The generalization of the dot product in a Hilbert space $\mathcal{F}$ can be written :

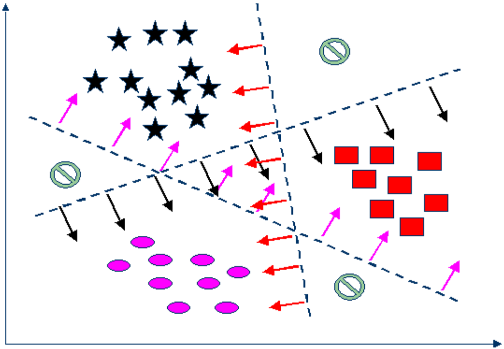
$$K(\mathbf{x},\mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle = \sum_{i=1}^{\infty} \lambda_i \phi_i(\mathbf{x}) \phi_i(\mathbf{z}), \lambda_i \geq 0$$

Mercer's theorem gives the N.S.C for a continuous symmetric function to admit such development. Different techniques exist to construct kernels. Construction by transforming the data, construction combining several kernels ...

General form of extensions

1. Write a global optimization problem for J hyperplanes simultaneously
2. Data extension $(n, p) \Rightarrow (nJ, p+1)$ and use binary SVM
3. Combine several binary hyperplanes each used in :
   - One versus one approach
   - One versus others approach
4. Aggregate hyperplanes :
   Winner takes all, majority vote, Pseudo-loss, ADAG, DDAG, RADAG, ECOC

One versus Rest



Agregating one versus rest classifiers

- We have J hyperplanes
- Winner takes all
  Each hyperplane gives a decision function

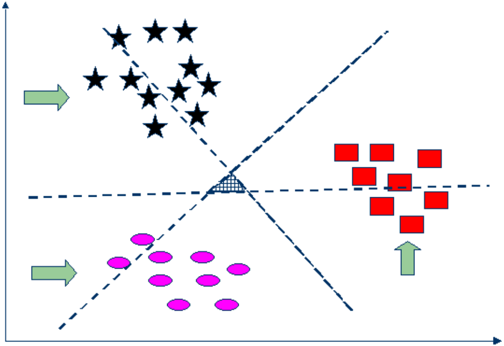
$$f_k(x) =< w_k, x > +b_k$$

  And

$$f(x) = Argmax_k f_k(x)$$

- No upper bounds available for the generalization

One versus One approach



Aggregating 1 vs 1classifiers

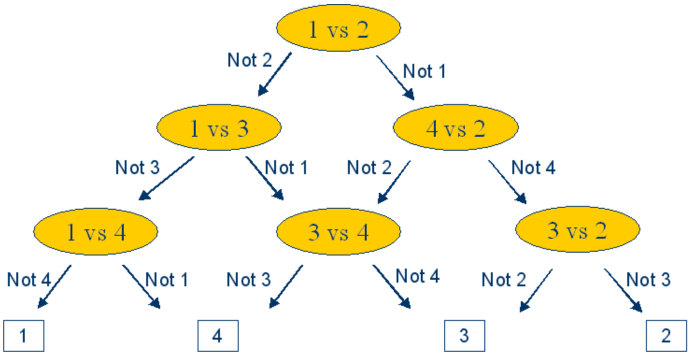
- We have $J(J-1)/2$ hyperplanes, giving the binary decision functions $f_{lh}(x)$
- Majority vote : Set $f_l(x) = f_{l1}(x) + f_{l2}(x) + ... + f_{lK}(x)$

$$f(x) = Argmax_k f_k(x)$$

- Problem : Ambiguity region...

Objective : Reduce this region without increasing generalization error.

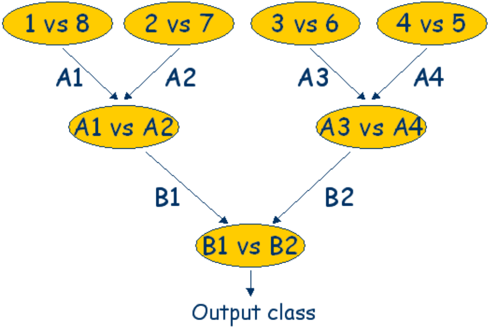
DDAG


1 vs 2
Not 2
Not 1
1 vs 3
4 vs 2
Not 3
Not 1
Not 2
Not 4
1 vs 4
3 vs 4
3 vs 2
Not 4
Not 1
Not 3
Not 4
Not 2
Not 3
1
4
3
2

ADAG, Tennis tournament

- Learning time
- Execution time ($J-1$ evaluations, $J-1$ stratas)
- Generalization error bounded
- But
  - Depend on the order of the classes
  - Errors are cumulated at each hyperplane.

ADAG, Tennis tournament



LAB 1 - TP1

1. Run the script TP-Reg.R
2. Run the script TP-Class.R
3. Change the data set in each script choosing one from those proposed in R (see function datasets)
4. Download a data set from the UCI machine learning into R and apply the corresponding script on it.

Annexes

1. The horizontal position, counting pixels from the left edge of the image, of the center of the smallest rectangular box that can be drawn with all "on" pixels inside the box.
2. The vertical position, counting pixels from the bottom, of the above box.
3. The width, in pixels, of the box.
4. The height, in pixels, of the box.
5. The total number of "on" pixels in the character image.
6. The mean horizontal position of all "on" pixels relative to the center of the box and divided by the width of the box. This feature has a negative value if the image is "left-heavy" as would be the case for the letter L.
7. The mean vertical position of all "on" pixels relative to the center of the box and divided by the height of the box.
8. The mean squared value of the horizontal pixel distances as measured in 6 above. This attribute will have a higher value for images whose pixels are more widely separated in the horizontal direction as would be the case for the letters W or M.
9. The mean squared value of the vertical pixel distances as measured in 7 above.
10. The mean product of the horizontal and vertical distances for each "on" pixel as measured in 6 and 7 above. This attribute has a positive value for diagonal lines that run from bottom left to top right and a negative value for diagonal lines from top left to bottom right.
11. The mean value of the squared horizontal distance times the vertical distance for each "on" pixel. This measures the correlation of the horizontal variance with the vertical position.
12. The mean value of the squared vertical distance times the horizontal distance for each "on" pixel. This measures the correlation of the vertical variance with the horizontal position.
13. The mean number of edges (an "on" pixel immediately to the right of either an "off" pixel or the image boundary) encountered when making systematic scans from left to right at all vertical positions within the box. This measure distinguishes between letters like "W" or "M" and letters like "I" or "L."
14. The sum of the vertical positions of edges encountered as measured in 13 above. This feature will give a higher value if there are more edges at the top of the box, as in the letter "Y."
15. The mean number of edges (an "on" pixel immediately above either an "off" pixel or the image boundary) encountered when making systematic scans of the image from bottom to top over all horizontal positions within the box.
16. The sum of horizontal positions of edges encountered as measured in 15 above.

References

- Freund, Y. *Boosting a weak learning algorithm by majority.*. Inf. Comput. 121, No.2, 256-285 (1995)
- Trevor, H. Tibshirani, R. and Friedman, J. *The elements of statistical learning. Data mining, inference and prediction. (English)*. Springer Series in Statistics. New York, NY : Springer.
- Breiman L., Friedman J., Olshen J., Stone C., *Classification And Regression Trees*, 1984.
- Breiman L. Random Forests, *Journal of Machine Learning Research*, 2001.
- Cristianini, N. & Shawe-Taylor J. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, 2000.