# Decision and regression tree ensemble methods and their applications in automatic learning


Louis Wehenkel

Department of Electrical Engineering and Computer Science
University of Liège - Institut Montefiore

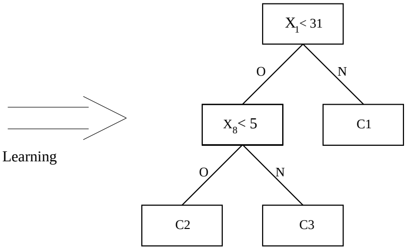

ORBEL Symposium - March 16, 2005

# Part I

# Supervised learning algorithm (Batch Mode)

- Inputs: learning sample $ls$ of $(x, y)$ observations ($ls \in (X \times Y)^*$)
- Output: a model $f_A^{ls} \in \mathcal{F}_A \subset Y^X$ (decision tree, MLP, ...)

| $X_1$ | $X_2$ | $X_3$ | $X_4$ | $X_5$ | $X_6$ | $X_7$ | $X_8$ | Y |
|---|---|---|---|---|---|---|---|---|
| 60 | 19 | 18 | 17 | 0 | 1 | 1 | 1 | C1 |
| 60 | 3 | 22 | 23 | 1 | 29 | 11 | 23 | C1 |
| 75 | 9 | 2 | 1 | 3 | 77 | 46 | 3 | C1 |
| 2 | 10 | 10 | 2 | 234 | 0 | 0 | 0 | C2 |
| 3 | 7 | 9 | 18 | 5 | 0 | 0 | 0 | C2 |
| 2 | 14 | 5 | 10 | 8 | 10 | 8 | 10 | C3 |
| 65 | 3 | 20 | 21 | 2 | 0 | 1 | 1 | ? |




- Objectives:
  - maximise accuracy on independent observations
  - interpretability, scalability

# Induction of single decision/regression trees (Reminder)

- Algorithm development (1960-1995)
  - Top-down growing of trees by recursive partitioning
    - local optimisation of split score (variance, entropy)
  - Bottom-up pruning to prevent over-fitting
    - global optimisation of complexity vs accuracy (B/V tradeoff)
- Characterisation
  - Highly scalable algorithm
  - Interpretable models (rules)
  - Robustness: irrelevant variables, scaling, outliers
  - Expected accuracy often low (high variance)
- Many variants and extensions
  - C4.5, CART, ID3 ...
  - oblique, fuzzy, hybrid ...

# Bias/variance decomposition (of average error)

Accuracy of models produced by an algorithm in a given context

- Assume problem (inputs $X$, outputs $Y$, relation $P(X,Y)$)

  and sampling scheme (e.g. fixed size $LS \sim P^N(X,Y)$).
- Take model error function (e.g. $Err_{f,Y} \equiv E_{X,Y}\{(f(X)-Y)^2\}$)

  and evaluate *expected* error of algo $A$ (i.e. $\overline{Err}_{A,Y} \equiv E_{LS}\{Err_{f_A^{ls},Y}\}$)
- We have $\overline{Err}_{A,Y} - Err_{B,Y} = Bias_A^2 + Var_A$

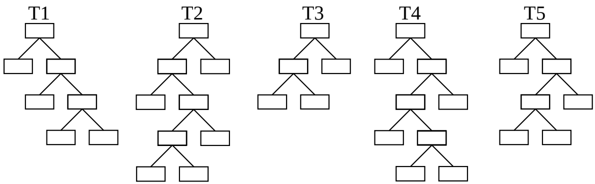
  where
  - $B$ is the best possible model (here, $B(\cdot) \equiv E_{Y|\cdot}$)
  - $Bias_A^2 = Err_{\overline{f}_A,B}$ ($\overline{f}_A$ is the average model)
  - $Var_A = \overline{Err}_{A,\overline{f}_A}$ (dependence of model on sample)

# Ensembles of trees

(How?/Why?)

- Perturb and Combine paradigm (1990-2005)
  - Build several trees (e.g. 100, by randomisation)
  - Combine trees by voting, averaging... (i.e. aggregation)
- Characterisation
  - Can preserve scalability (+ trivially parallel)
  - Does not preserve interpretability
  - Can preserve robustness (irrelevant variables, scaling, outliers)
  - Can improve accuracy significantly
- Many generic variants (Bagging, Stacking, Boosting, ...)
- Non-generic variants: (Random Forests, Random Subspace, ...)

# Extra-Trees: learning algorithm

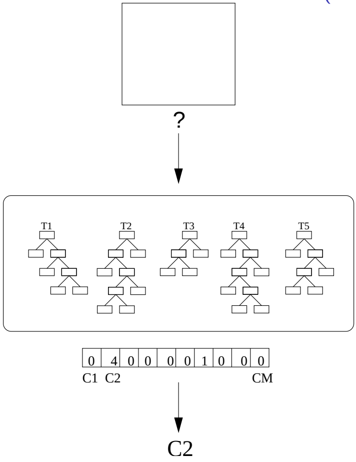



- Ensemble of trees $T_1, T_2, \ldots T_T$ (generated independently)
- Random splitting (choice of variable and cut-point)
- Trees are fully developed (perfect fit on *ls*)
- Ultra-fast ($\sqrt{n}N \log N$)

(Presentation based on [Geu02, GEW04])

# Extra-Trees: prediction algorithm

- Aggregation (majority vote or averaging)



# Extra-Trees splitting algorithm (for numerical attributes)

Given a node of a tree and a sample $S$ corresponding to it

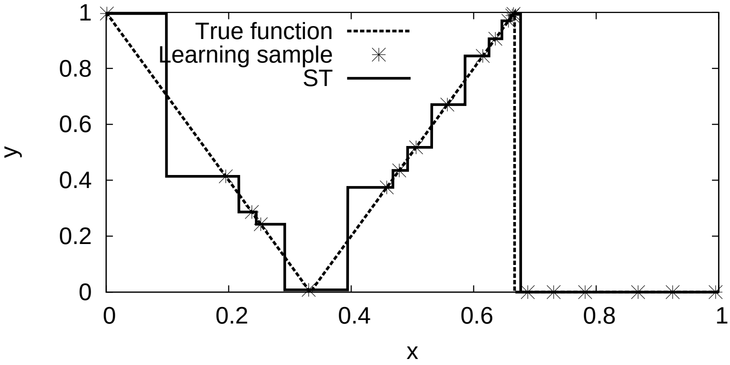
- Select $K$ attributes $\{X_1, \ldots, X_K\}$ at random;
- For each $X_i$ (draw a split at random)
  - Let $x^S_{i,\min}$ and $x^S_{i,\max}$ be the min and max values of $X_i$ in $S$;
  - Draw a cut-point $x_{i,c}$ uniformly in $]x^S_{i,\min}, x^S_{i,\max}]$;
  - Let $t_i = [X_i < x_{i,c}]$.
- Return a split $t_i = \arg\max_{t_i} \text{Score}(t_i, S)$.

NB: the node becomes a LEAF

- if $|S| < n_{\min}$;
- if all attributes are constant in $S$;
- if the output is constant in $S$;

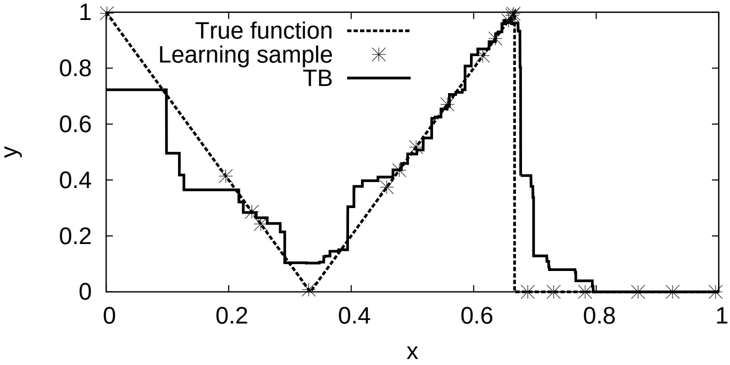
# Geometric properties

(of Single Trees)



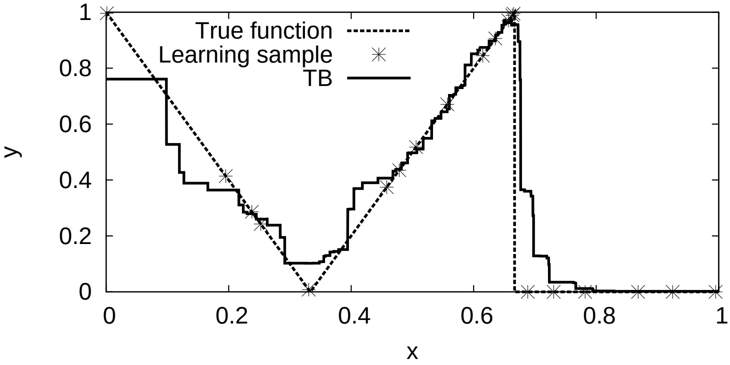

A single fully developed CART tree.

# Geometric properties (of Tree Bagging models)



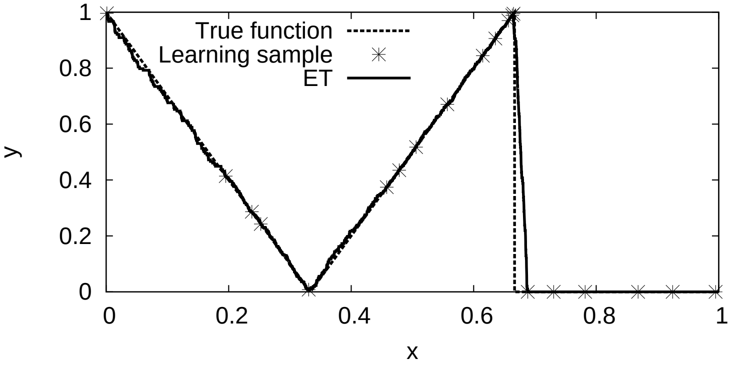

With $T = 100$ trees in the ensemble.

# Geometric properties (of Tree Bagging models)



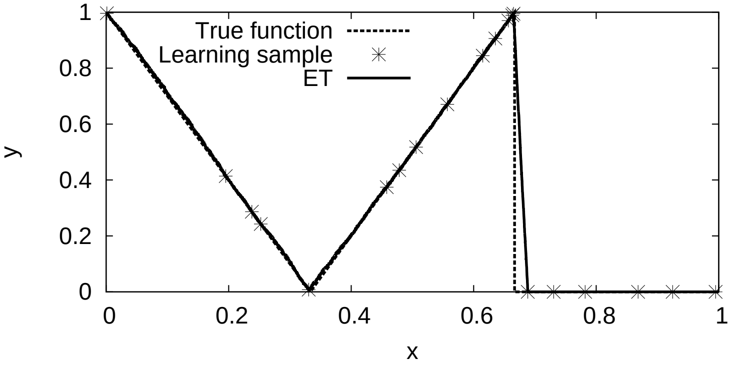

With $T = 1000$ trees in the ensemble.

# Geometric properties (of Extra-Trees models)




With $T = 100$ trees in the ensemble.

# Geometric properties (of Extra-Trees models)

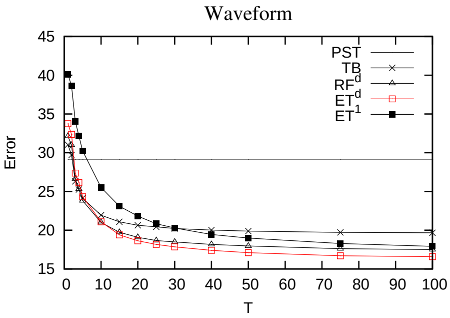


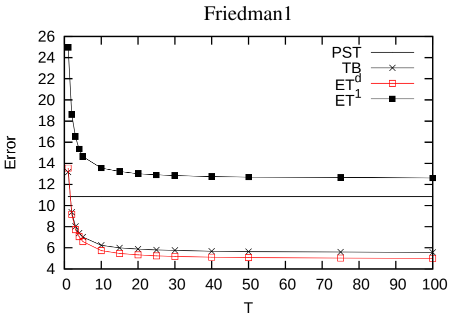

With $T = 1000$ trees in the ensemble.

# Parameters

(of the Extra-Trees learning algorithm)

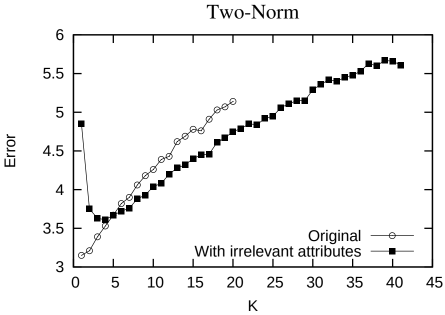
## Averaging strength $T$

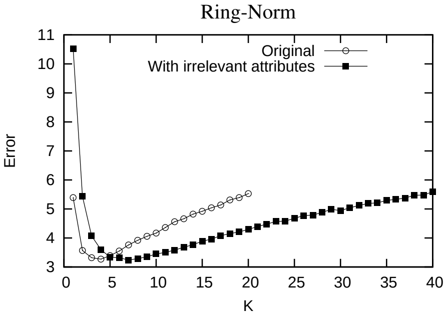


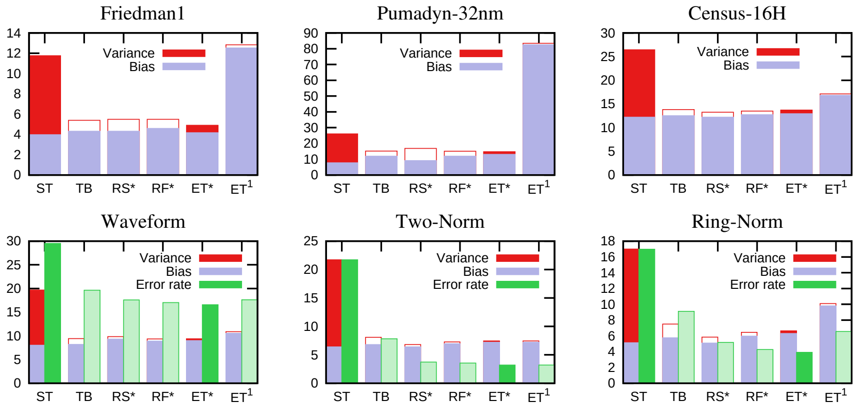
# Parameters (of the Extra-Trees learning algorithm)

## Attribute selection strength $K$ (w.r.t. irrelevant variables)



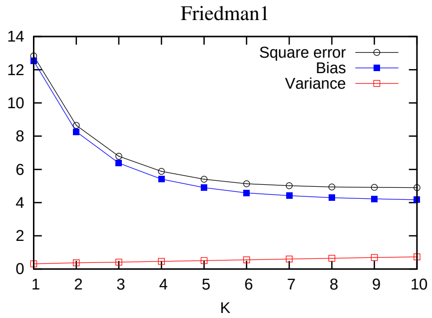
# Bias/variance tradeoff (of the Extra-Trees models)

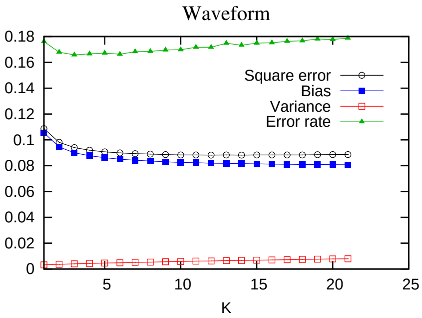


# Bias/variance tradeoff

(of the Extra-Trees learning algorithm)

## Effect of attribute selection strength $K$



# Extra-Trees: variants of setting $K$

Automatic tuning of $K$

- by (10-fold) cross-validation
- on (large enough) independent test sample

Default settings

- $K = \sqrt{n}$, in classification
- $K = n$, in regression

($n$ =number of variables)

Totally randomised trees

- correspond to $K = 1$
- splits (attribute and cut-point) totally at random
- ultra-fast "non-supervised" learning algorithm
- tree structures independent of output values
- akin to $K$NN, or kernel-based method

# Optimal control problem

(stochastic, discrete-time, infinite horizon)

$x_{t+1} = f(x_t, u_t, w_t)$ (stochastic dynamics, $w_t \sim P_w(w_t|x_t, u_t)$)

$r_t = r(x_t, u_t, w_t)$ (real valued reward signal bounded over $X \times U \times W$)

$\gamma$ (discount factor $\in [0, 1)$)

$\mu(\cdot) : X \to U$ (closed-loop, stationary control policy)

$J_h^\mu(x) = E\left\{\sum_{t=0}^{h-1} \gamma^t r(x_t, \mu(x_t), w_t) | x_0 = x\right\}$ (finite horizon return)

$J_\infty^\mu(x) = \lim_{h\to\infty} J_h^\mu(x)$ (infinite horizon return)

Optimal *infinite* horizon control policy

$\mu_\infty^*(\cdot)$ that maximises $J_\infty^\mu(x)$ for all $x$.

(Presentation based on [EGW03, EGW05])

# Batch mode reinforcement learning problem

Suppose that instead of system model $(f(\cdot,\cdot,\cdot), r(\cdot,\cdot,\cdot), P_w(\cdot|\cdot,\cdot))$, the only information we have is a (finite) sample $F$ of four-tuples:

$$F = \{(x_{t^i}, u_{t^i}, r_{t^i}, x_{t^i+1}), i = 1, \cdots, \#F\}.$$

Each four-tuple corresponds to a system transition

The objective of batch mode RL is to determine an approximation $\hat{\mu}(\cdot)$ of $\mu^*_\infty(\cdot)$ from the sole knowledge of $F$

(Many one-step episodes: $x_{t^i}$ distributed independently)

(One single episode with many steps: $x_{t^{i+1}} = x_{t^i+1}$)

(In general: several multi-step episodes)

# $Q$-function iteration to solve Bellman equation

Idea: $\mu^*_\infty(\cdot) \equiv$ can be obtained as the limit of a sequence of optimal finite horizon (time-varying) policies.

Define sequence of value-functions $Q_h$ and policies $\mu^*_h(t,x)$ by:

$Q_0(x,u) \equiv 0$

$Q_h(x,u) = E_{w|x,u}\{r(x,u,w) + \gamma \max_{u'} Q_{h-1}(f(x,u,w),u')\}$ $(\forall h \in \mathbb{N})$

$\mu^*_h(t,x) = \arg\max_u Q_{h-t}(x,u)$ $(\forall h \in \mathbb{N}, \forall t = 0,\ldots,h-1)$

NB: these sequences converge $(Q_h \xrightarrow{\sup} Q_\infty$ and $\mu^*_h(t,x) \xrightarrow{J^\mu_\infty} \mu^*_\infty(x))$

Alternative view: (Bellman equation)

$Q_\infty(x,u) = E_{w|x,u}\{r(x,u,w) + \gamma \max_{u'} Q_\infty(f(x,u,w),u')\}$

$\mu^*_\infty(x) = \arg\max_u Q_\infty(x,u)$

# Fitted $Q$ iteration algorithm

Idea1: replace expectation operator $E_{w|x,u}$ by average over sample
Idea2: represent $Q_h$ by model to interpolate from samples
Supervised learning (regression): does the two in a single step

---

- Inputs:
  - a set $F$ of four-tuples $((x_{t^i}, u_{t^i}, r_{t^i}, x_{t^i+1}), i = 1, \cdots, \#F)$
  - a regression algorithm $A$ $(A : ls \rightarrow f_A^{ls})$
- Initialisation: $\hat{Q}_0(x, u) \equiv 0$
- Iteration: (for $h = 1, 2, \ldots$)
  - Training set construction: $(\forall i = 1, \ldots \#F)$
    $x_i = (x_{t^i}, u_{t^i})$;
    $y_i = r_{t^i} + \gamma \max_u \hat{Q}_{h-1}(x_{t^i+1}, u)$,
  - $Q$-function fitting:
    $\hat{Q}_h = A(ls)$ where $ls = ((x_1, y_1), \ldots, (x_{\#F}, y_{\#F}))$

# Coupling with tree-based models

Use tree-based regression as supervised learning algorithm

- Tree-based methods: ‘non-divergence’ to infinity
- Kernel-based methods: ‘convergence’ (when $h \to \infty$)
- Tree structures frozen for $h > h_0 \Rightarrow$ kernel-based method

Solves at the same time

- System identification (implicitly)
- State-space discretisation (and curse-of-dimensionality)
- Bellman equation (iteratively and approximately)

Generality of the framework

- Non strong hypothesis on $f$, $r$ (discrete, continuous, high-dimensional)
- Minimum-time problems (define $r(x,u,w) = 1_{Goal}(f(x,u,w))$)
- Stabilisation problems (define $r(x,u,w) = ||f(x,u,w) - x_{ref}||$)

# Illustration - Electric power system stabilisation

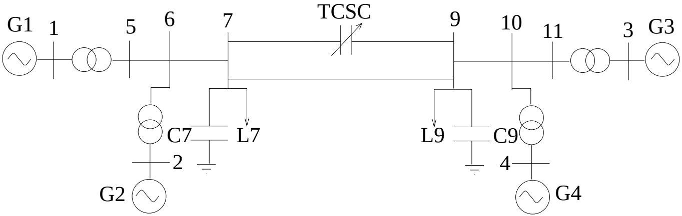



Figure: Four-machine test system

- Use of simulator + 1000 random episodes (60s, $\Delta t$ =50ms)
- 5-dimensional $X \times U$ space; $\mathcal{F}$ contains 1100,000 four-tuples.
- "Reward": power oscillations and loss of stability ($\gamma = 0.95$)
- Policy learning by fitted $Q$-function iteration ($h = 100$) with Extra-Trees ($T = 50$; $K = 5$; $n_{\text{min}} = 2$)

# Electric power system stabilisation

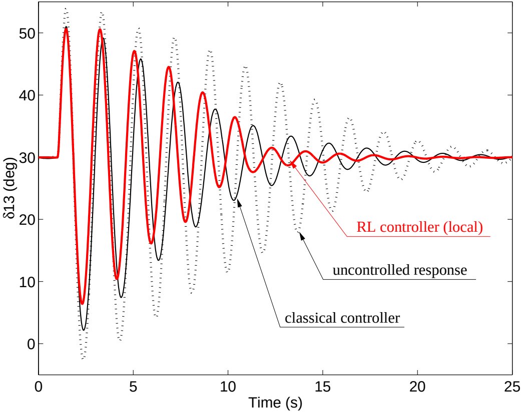



Figure: The system responses to 100 ms, self-clearing, short circuit

# Electric power system stabilisation

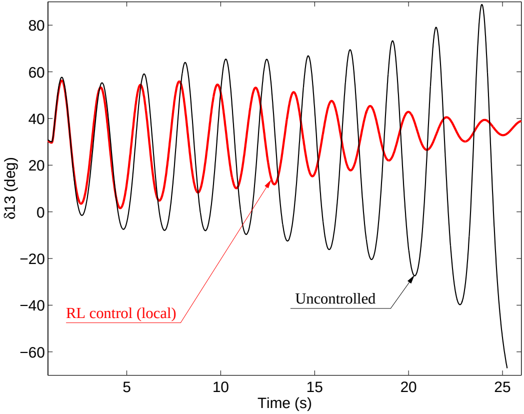



Figure: 100 ms short circuit cleared by opening line

# Electric power system stabilisation

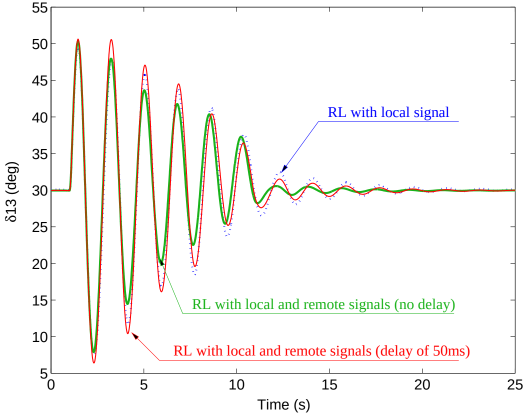



Figure: Local vs remote signals with/without communication delay

# Generic pixel-based image classification

Challenge:

*Create a robust image classification algorithm by the sole use of supervised learning on the low-level pixel-based representation of the images.*

Question:

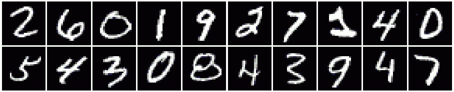
*How to inject invariance (scale, translation, orientation) in a generic way into a supervised learning algorithm ?*

NB: work used mainly on Extra-Trees, but other supervised learners could also be used (e.g. SVMs, $KNN\ldots$).

(Presentation based on [MGPW04, MGPW05])

# Examples

- Hand written digit recognition *(0, 1, 2, ..., 9)*



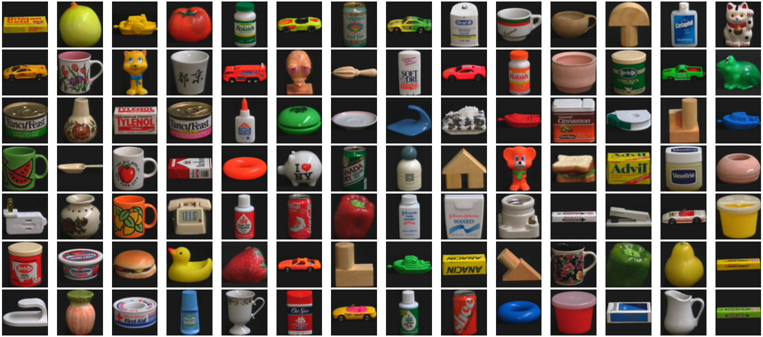

- Face classification *(Jim, Jane, John, ...)*

# Examples

- Texture classification *(Metal, Bricks, Flowers, Seeds, ...)*

# Examples

- Object recognition *(Cup X, Bottle Y, Fruit Z, ...)*



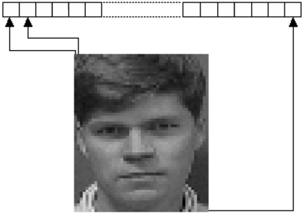
# Principle of solution

(global)

- Learning sample of $N$ pre-classified images,

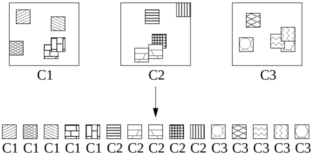
$$ls = \{(\mathbf{a}^i, c^i), i = 1, \dots, N\}$$

$\mathbf{a}^i$: vector of pixel values of the entire image
$c^i$: image class



# Principle of solution

(local)

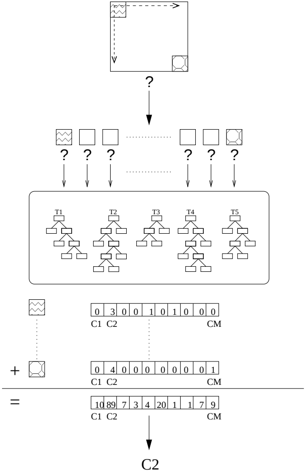



Learning sample of $N_w$ sub-windows (size $w \times w$, pre-classified),

$$ls = \{(\mathbf{a}^i, c^i), i = 1, \ldots, N_w\}$$

$\mathbf{a}^i$: vector of pixel-values of the sub-window
$c^i$: class of mother image (from which the window was extracted)

# Local approach: prediction



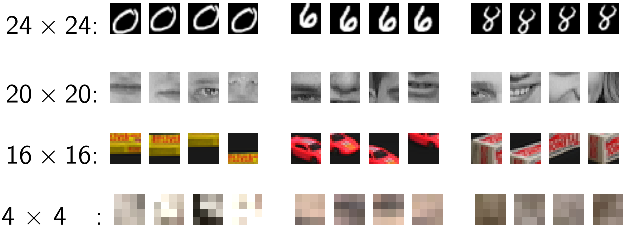
# Datasets and protocols

| Datasets | # images | # base attributes | # classes | $N_w$ | $w$ |
|---|---|---|---|---|---|
|  MNIST | 70000 | 784 $(28 * 28 * 1)$ | 10 | 360,000 | 24 |
|  ORL | 400 | 10304 $(92 * 112 * 1)$ | 40 | 120,000 | 20 |
|  COIL-100 | 7200 | 3072 $(32 * 32 * 3)$ | 100 | 120,000 | 16 |
|  OUTEX | 864 | 49152 $(128 * 128 * 3)$ | 54 | 120,000 | 4 |

- MNIST: $LS$ = 60000 images ; $TS$ = 10000 images
- ORL: Stratified cross-validation: 10 random splits $LS = 360$; $TS = 40$
- COIL-100: $LS$ = 1800 images ; $TS$ = 5400 images (36 images per object)
- OUTEX: $LS$ = 432 images (8 images per texture) ; $TS$ = 432 images (8 images per texture)

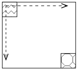
# A few results: accuracy

| DBs | Extra-Trees | Extra-Trees with sub-windows | State-of-the-art |
|---|---|---|---|
| MNIST | 3.26% | 2.63% | 0.5% [DKN04] |
| ORL | 4.56% $\pm$ 1.43 | 1.66% $\pm$ 1.08 | 2.0% [Rav04] |
| COIL-100 | 1.96% | 0.37% | 0.1% [OM02] |
| OUTEX | 65.05% | 2.78% | 0.2% [MPV02] |



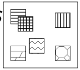
# A few results: CPU times

- Learning stage: depends on parameters
  MNIST: 6h, ORL: 37s, COIL-100: 1h, OUTEX: 11m
- Prediction: depends on parameters and sub-window sampling
  - *Exhaustive (all sub-windows)*

    ?

    MNIST: 2msec, ORL: 354msec
    COIL-100: 14msec, OUTEX: 800msec
  - *Random subset of sub-windows*

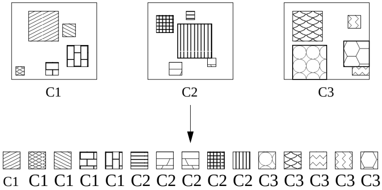
    ?

    MNIST: 1msec, ORL: 10msec
    COIL-100: 5msec, OUTEX: 33msec

# Sub-windows of random size

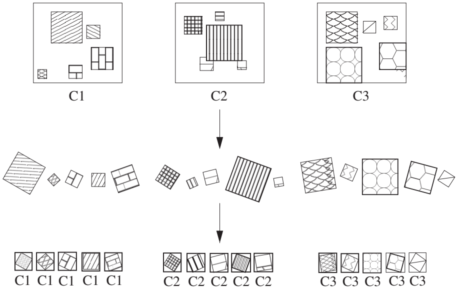
(robustness w.r.t. scale)

- Extraction of sub-windows of random size
- Rescaling to standard size



# Sub-windows of random size and orientation (more robustness)

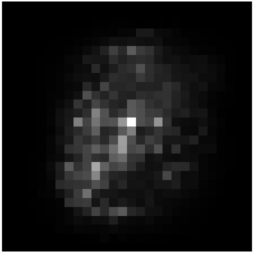
- Extraction of sub-windows of random size
- + Random rotation
- Rescaling to standard size



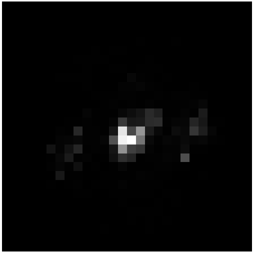
# Attribute importance measures (global approach)

Compute information quantity (Shannon) brought by each pixel in each tree, and average over the trees.

ORL (faces)

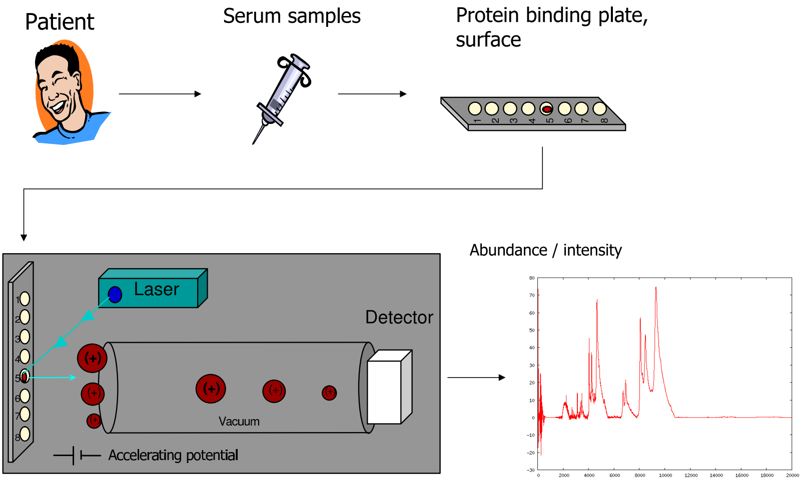


MNIST (all digits)



MNIST (0 vs 8)

# Part II




SELDI-TOF MS:

Surface Enhanced Laser Desorption/ Ionisation Time of Flight Mass Spectrometry

Patients

Normal

Disease

SELDI-TOF

m/z values (±15000 )

Patients (10-500)

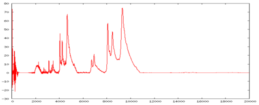
| 0 | 10.23 | ... | 20,234 | Class |
|---|---|---|---|---|
| 0.3 | 28.34 | ... | 123 | Normal |
| -123 | 0 | ... | 17 | ... |
| 56 | -123 | ... | -23 | Normal |
| ... | ... | ... | ... | Disease |
| | | | | ... |
| 89 | -123 | ... | 12 | Disease |

Machine Learning, statistical learning, pattern recognition

Classification model

New patient

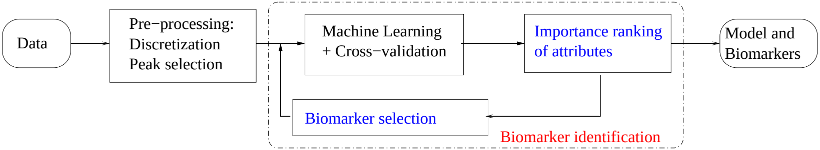
SELDI-TOF



Classification model

Disease or Normal

# Supervised learning based methodology




(Presentation based on [GFd+04])

# RA and IBD

**RA** Early diagnosis of Rhumatoid Arthritis

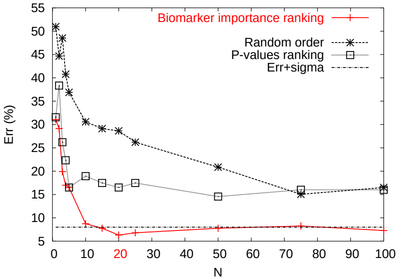
**IBD** Better understanding of Inflammatory Bowel Diseases

Datasets collected at University Hospital of Liège.

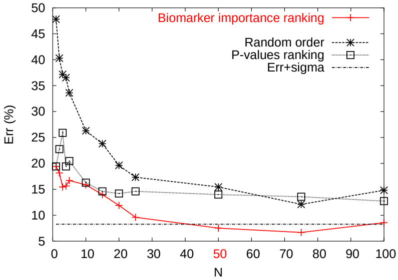
| Patients | | | Number of attributes | | | | |
|---|---|---|---|---|---|---|---|
| Dataset | #target | #others | Raw | $p = .3\%$ | $p = .5\%$ | $p = 1\%$ | Peaks |
| RA | 68 | 138 | 15445 | 1026 | 626 | 319 | 136 |
| IBD | 240 | 240 | 13799 | 1086 | 664 | 338 | 152 |

Toolbox: Single trees, Tree Bagging, Tree Boosting, Random Forests, Extra-Trees

# Biomarker identification



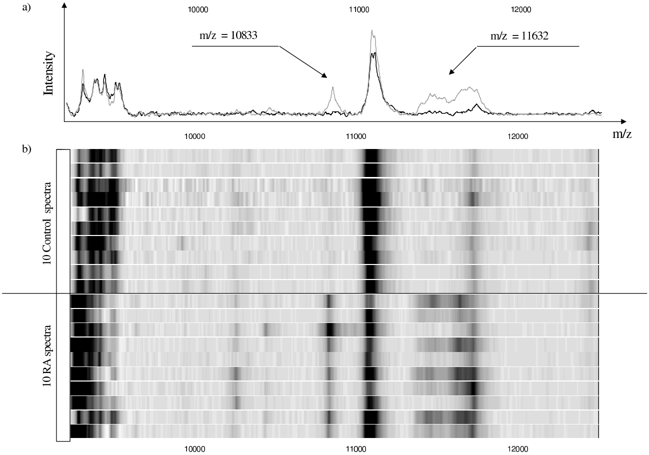

RA dataset




IBD dataset

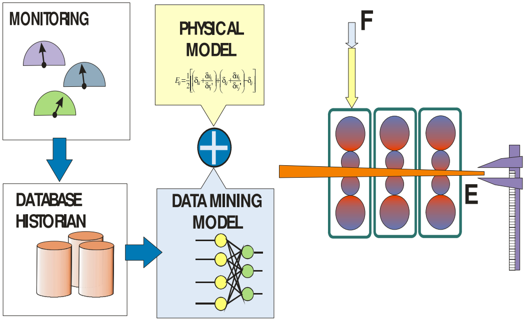
Figure: Variation of accuracy with number of biomarkers (Tree Boosting)

# Graphical visualisation of biomarker identification (RA)



# Steal-mill control

(ULg, PEPITe, ARCELOR)




- Development of a friction model, taking into account steel quality and temperature.
- Improve pre-setting of steel-mill controller
- Reduce waste

# Wide area control of power systems (ULg, PEPITe, Hydro-Québec)


- Improve emergency control scheme
  - Churchill-Falls power plant
- Reduce probability of blackout
  - Reduce over/under-tripping
  - Adjust load shedding scheme
- Database generation
  - 10,000 real-time snapshots sampled (several years)
  - Massive time-domain simulations
- New rules in operation
- Methodology has been adopted

# Failure analysis of manufacturing process (PEPITe, Valéo)

## Problem

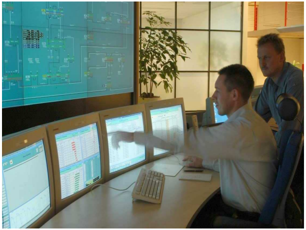
- Car reflector manufacturing line
- High, unexplained defect rate
- 40 process parameters (T,H, pH, flow...) measured every 5 minutes

## Approach

- Two-month period data collection
- Database of 10,000×40 measurements
- Data mining using PEPITo software
- Identification of the root cause
- Default rate reduced by 20%

# SCADA system data mining

(PEPITe, AREVA, TENNET)



Challenges faced by TENNET (South NL subsystem)

- Minimise exchanges of reactive power
  - Formalise operators actions
  - Discover optimal network states
  - Optimise forecasting of industrial loads
- Decide of network upgrades effectively
  - Justify long-term planning decisions
  - Validation of state estimator

Goal of this project: show the value of Data Mining with respect to these challenges.

Based on 6 months, 15´ sampling of 3200 data-points

- 900 status, 2200 analog, 100 calculated
- Database: 16000 rows, 3200 columns

T. Deselaers, D. Keysers, and H. Ney.
Classification error rate for quantitative evaluation of content-based image retrieval systems.
In *Proceedings of the 7th International Conference on Pattern Recognition*, 2004.

D. Ernst, P. Geurts, and L. Wehenkel.
Iteratively extending time horizon reinforcement learning.
In *Proceedings of the 14th European Conference on Machine Learning*, 2003.

D. Ernst, P. Geurts, and L. Wehenkel.
Tree-based batch mode reinforcement learning.
To appear in *Journal of Machine Learning Research*, 2005.

P. Geurts.
*Contributions to decision tree induction: bias/variance tradeoff and time series classification*.
Phd. thesis, Department of Electrical Engineering and Computer Science, University of Liège, May 2002.

P. Geurts, D. Ernst, and L. Wehenkel.
Extremely randomized trees.
*Submitted for publication*, 2004.

P. Geurts, M. Fillet, D. de Seny, M.-A. Meuwis, M.-P. Merville, and L. Wehenkel.
Proteomic mass spectra classification using decision tree based ensemble methods.
*Submitted for publication*, 2004.

R. Marée, P. Geurts, J. Piater, and L. Wehenkel.
A generic approach for image classsification based on decision tree ensembles and local sub-windows.
In *Proceedings of the 6th Asian Conference on Computer Vision*, 2004.

R. Marée, P. Geurts, J. Piater, and L. Wehenkel.
Random subwindows for robust image classification.
To appear in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2005.

T. Mäenpää, M. Pietikäinen, and J. Viertola.
Separating color and pattern information for color texture discrimination.
In *Proceedings of the 16th International Conference on Pattern Recognition*, 2002.

S. Obdržálek and J. Matas.
Object recognition using local affine frames on distinguished regions.
In *Electronic Proceedings of the 13th British Machine Vision Conference*, 2002.

S. Ravela.
Shaping receptive fields for affine invariance.
In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2004.