Decision trees and Random Forests

Christel Dartigues-Pallez
MinD team –I3S laboratory
dartigue@unice.fr

Decision trees

Principle for supervised learning

- Build a classification model Y=f(X)
- Select a technique of modelization
  - Decision tree, Neural network, Support Vector Machines, etc.
- Specify an evaluation protocol
  - Separating a learning sample and test
  - Evaluation criteria: error rate, area under the ROC curve, etc.
- Build the model
  - Build the model with the training sample
  - Use the test sample to check if there are features toajust
- Evaluate
  - Evaluate performance in generalization of the test sample

Type of data

- Discrete enumerative (color, zip code, etc.)
- Ordered or not
- Discrete ordinates (class of salary or class of age)
- Continues: income, weather, temperature
- Image and Video: extracting "features"
- Structured data: XML, etc.
- Text: Make a histogram by keyword
- We can transformed "easily" continues $\rightarrow$ discret $\rightarrow$ binaire

Decision theory

- Take optimal decisions in an uncertain universe
  - From an observation X, search the best decision t : p(x, t)
- Estimate p(x, t) frm the data is e complex inference problem
- Illustration
  - x is the X-ray image of a patient, t is the presence (Class C1) or absence of cancer (Class C2)

Decision trees - Analogy

Generalities– Decision trees

- Decision tree: simple and graphical classifier
  - Readability
  - Speed of learning and execution
- Goals
  - Distribute a population of individuals into homogeneous groups
  - According to a set of discriminant variables
  - According to a known target (supervised learning)
- Predicting the values of the variable to be predicted from a set of descriptors
  - Variable to predict = goal, target variable, variable of interest, class attribute, output variable
  - Descriptor = predictive variables, discriminating variables

Generalities– Decision trees

- Data structure used as a model for classification [Quinlan]
- Recursive method based on divide-and-conquer to create (more) pure subgroups
  - A subgroup is pure when all the sub-group elements belong to the same class
- Construction of the smallest possible decision tree
  - Node = Test on an attribute
  - A branch for each attribute value
  - The leaves denote the class of the object to be classified
- Error rate = proportion of instances that do not belong to the majority class of the branch
- Problems: attribute selection, termination

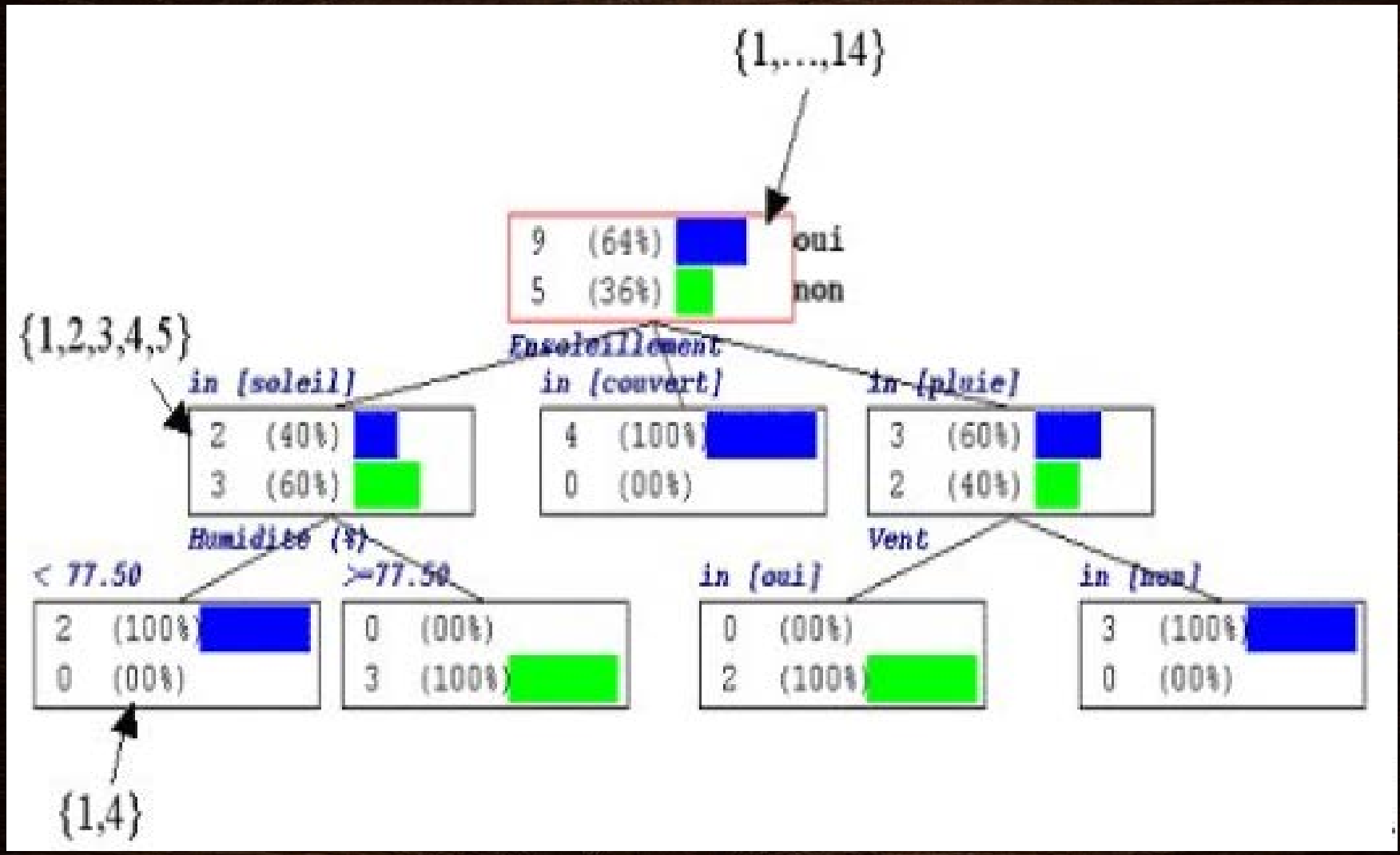
A simple example

- A set of days (a day = un example)
- Each day is characterized by a number and the weather conditions (outlook, temperature, humidity and windy)
- Goal attribut : «playing ?»
- Possibles values : yes and no (binary classification)
- Examples
  - 1, Sunny, Hot, High, False, No
  - 2, Sunny, Hot, High, True, No
  - 3, Overcast, Hot, High, False, Oui …
- Once the decision tree constructed, we can classify a new data corresponding to a new day to know whether or not we can play outside that day

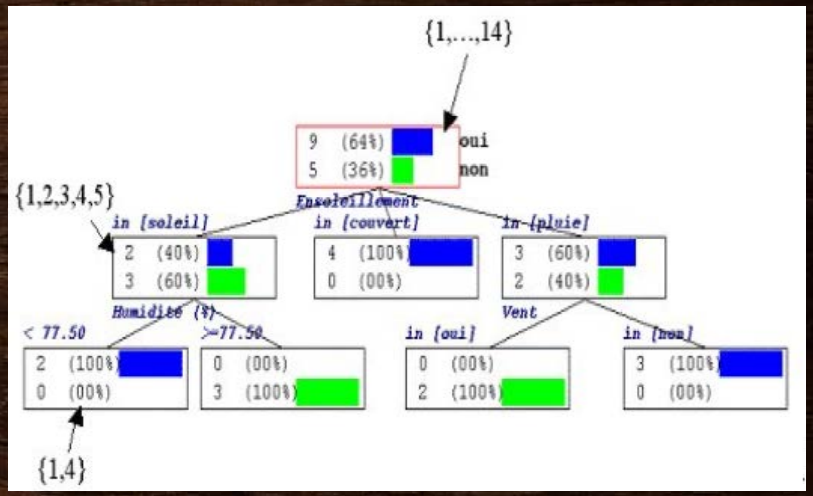
A simple example

outlook	temperature	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

A simple example



Example - explanations



- On the nodes
  - Distribution of the variable to predict
- The first node is segmented with the variable outlook: creation of 3 sub-groups
  - The first group contains 5 observations, 2 yes and 3 no
- The tree can be translated in a set ofdecision rules without loosing any information
  - Example: if outlook = sunny and humidity = high then play = yes

Building a tree

- To build a decision tree we must:
  - Choose, among the remaining variables, the segmentation variable of the current node
    - When the variable is continuous, we must determine a cutoff
  - Determine the optimal size of the tree
    - Is it a good option to produce absolutely pure leafs of the variable to predict, even though the corresponding group represents a very small fraction of the observations?
  - Affect the value of the variable to predict to the leaves

Basic algorithm

- A= BestAttribute(Examples)
- Assign A to the root
- For each value of A, create a new sub-node of the roof
- Classify all the examples in the sub-nodes
- If all examples of a sub-node are homogeneous, assign their class to the node, if not reproduce this process from this node

Termination

- Several possible criteria
  - All the attributes have been considered
  - It is no longer possible to obtain information gain
  - The leaves contain a predefined number of majority elements
  - The maximum of purity was achieved
    - All instances are in the same class
  - The tree reaches a maximum height

How many decision trees?

- Consider m Boolean attributes
- One possible decision tree for each of the m attibutes
  - 2^m ways to give values to attributes
- 2^{2^m} possible decision trees

$\rightarrow$how to select the best one?

Information theory

- Need a method to choose the attribute [Shannon & Weaver, 1949]
- At each step, at each node in the tree, calculate the information gain
  - The attribute with the highest information gain will be selected
- ID3 methods for the construction of the decision tree

Météo et match de foot

outlook	temperature	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

Attribut but

2 classes: yes et no

Predict if we can play outside or not

Temperature is nominal

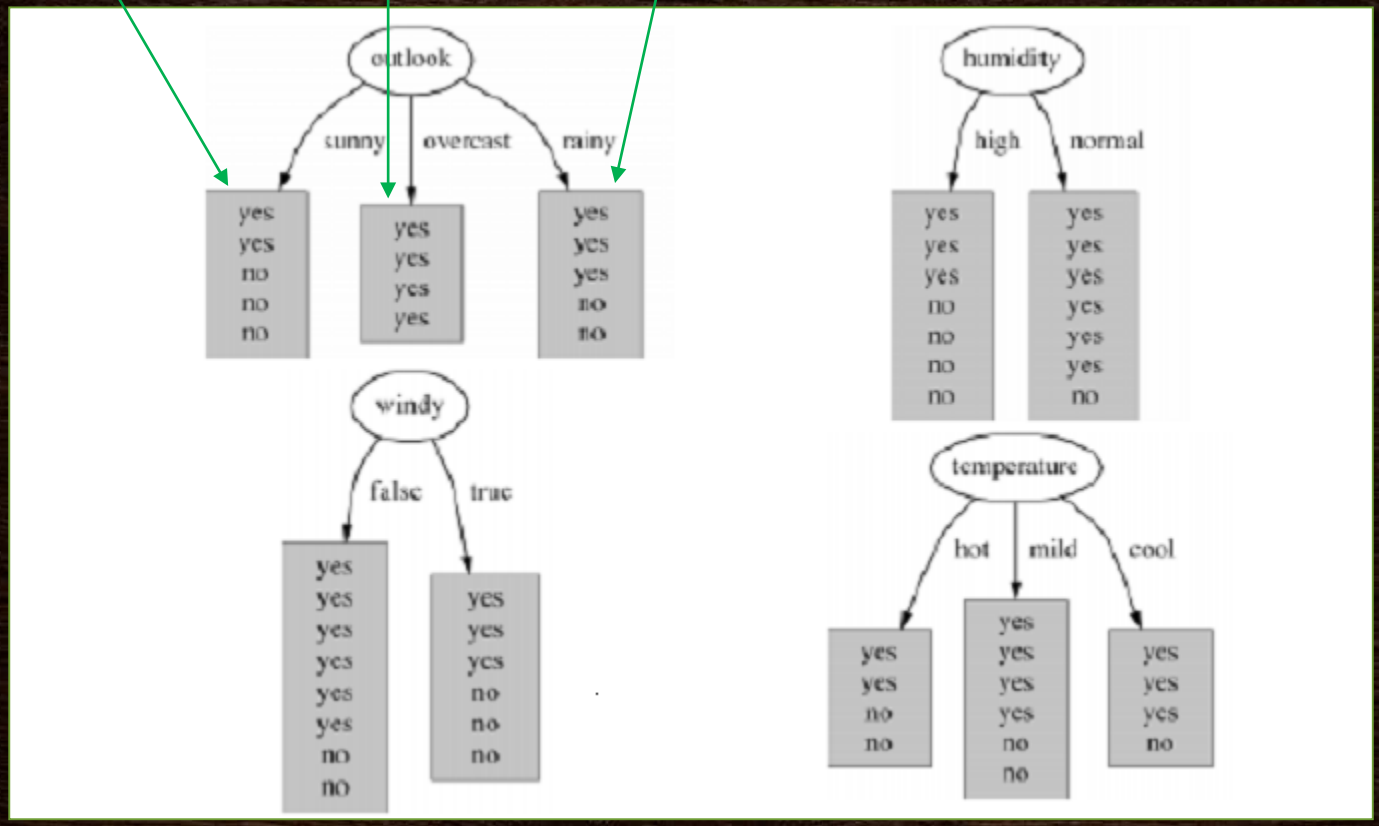
I.H. Witten and E. Frank, “Data Mining”, Morgan Kaufmann Pub., 2000.

Which attribute to the root?

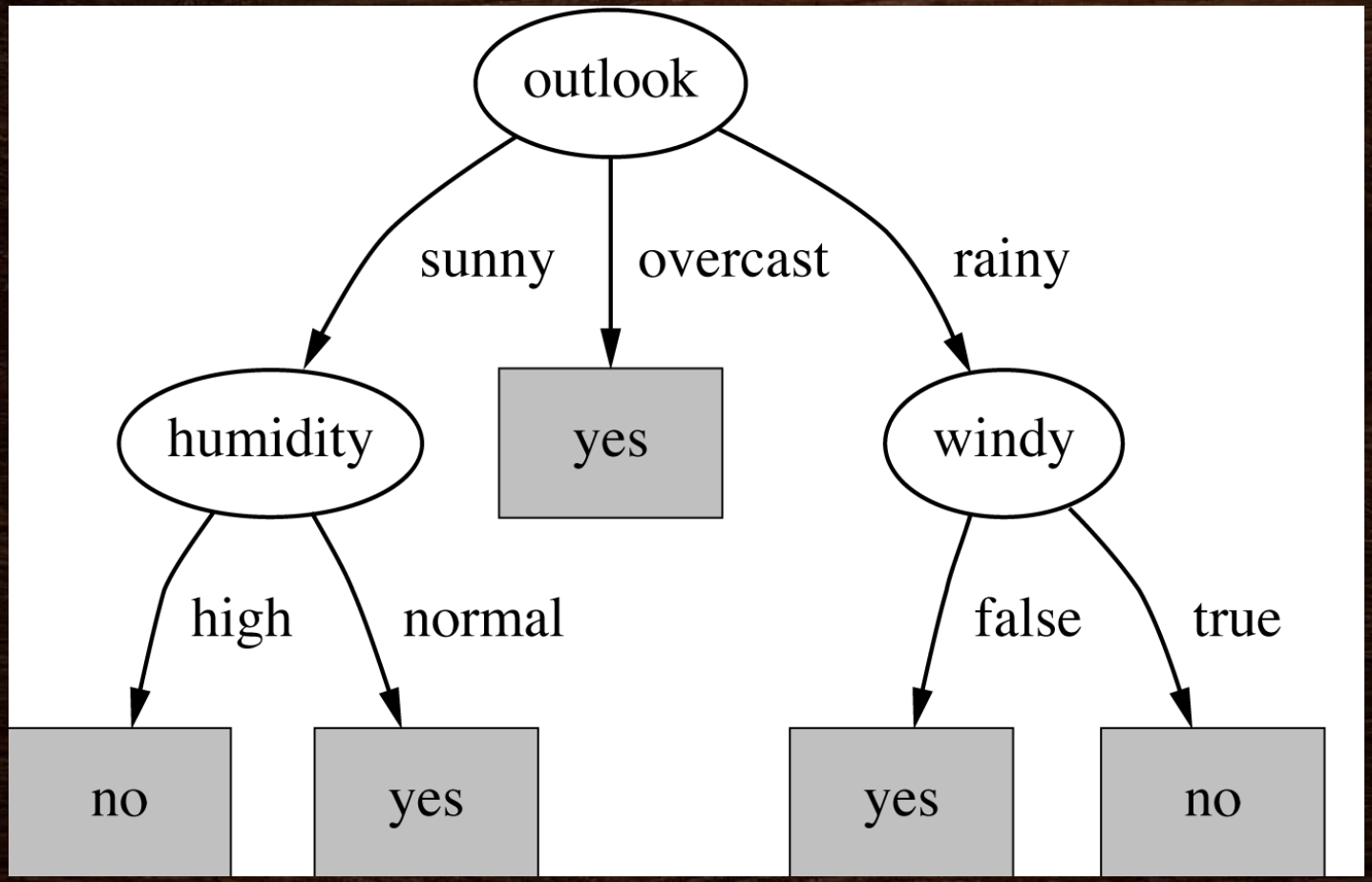
Majority class:YES

Majority class: NO

Majority class : YES

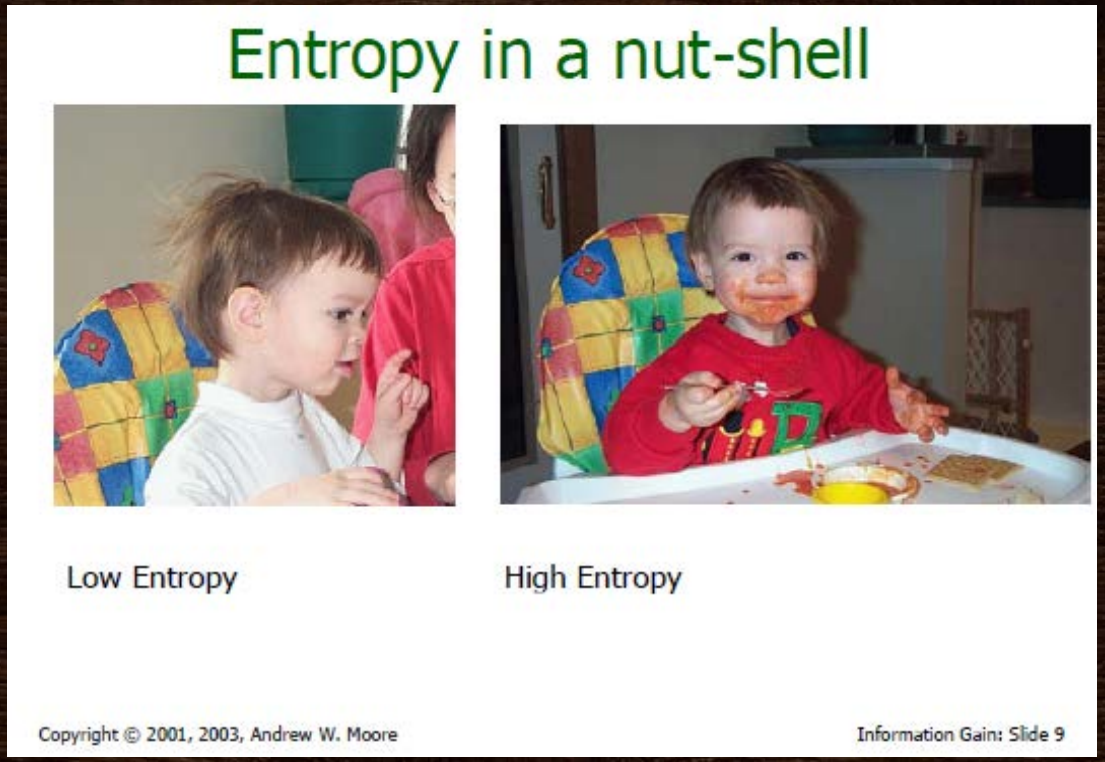


Final decision tree



Concept of entropy – Illustration

- Interpretation of entropy
  - Minimum number of bits needed to encode the class of any element



Building a decision tree – which attribute put to the root?

- Entropy = average amount of information of all messages in a system
- Shannon Entropy
  - Measure the heterogeneity of a population in terms of the class members
  - A high entropy is representative of heterogeneous data
- Consider a set of examples X : p+ represents the proportion of examples which are positive and p- represents the proportion of examples which are negative
- ((p+) + (p-) = 1)
- The entropy of X is : $H(X) = -(p+) \log_2(p+) - (p-)\log_2(p-)$

Entropy

- Simple case (2 classes)
  - $0 \leq H(X) \leq 1$
  - if $p+ = 0$ or $p- = 0$, then $H(X) = 0 \rightarrow$ pure *class*
  - if $p+ = p- = 0,5$, then $H(X) = 1$ (maximal entropy) : as many positives than negatives $\rightarrow$ worst case : we can not say anything
- General case (N classes)
  - The target attribute can have N distinct values
  - p_i ($i \in \{1,N\}$) is the proportion of examples whose value for this attribute is i in the set X of examples
  - The entropy of the set of examples X is :

$$H(X) = -(p_1 \log_2 p_1 + p_2 \log_2 p_2 + \ldots + p_N \log_2 p_N)$$

Information gain and entropy

$$\mathbf{Info}([n, m]) = \mathbf{Entropy}\left(\frac{n}{n+m}, \frac{m}{n+m}\right)$$

$$\mathbf{Entropy}(p_1, p_2) = \frac{1}{\log 2}\left(-p_1 \log p_1 - p_2 \log p_2\right)$$

- p_i is the probability of the class i
- p_i = *number of occurrences of* i / total nunber of occurrences
- This formula is generalizable

Entropy for 3 probabilities

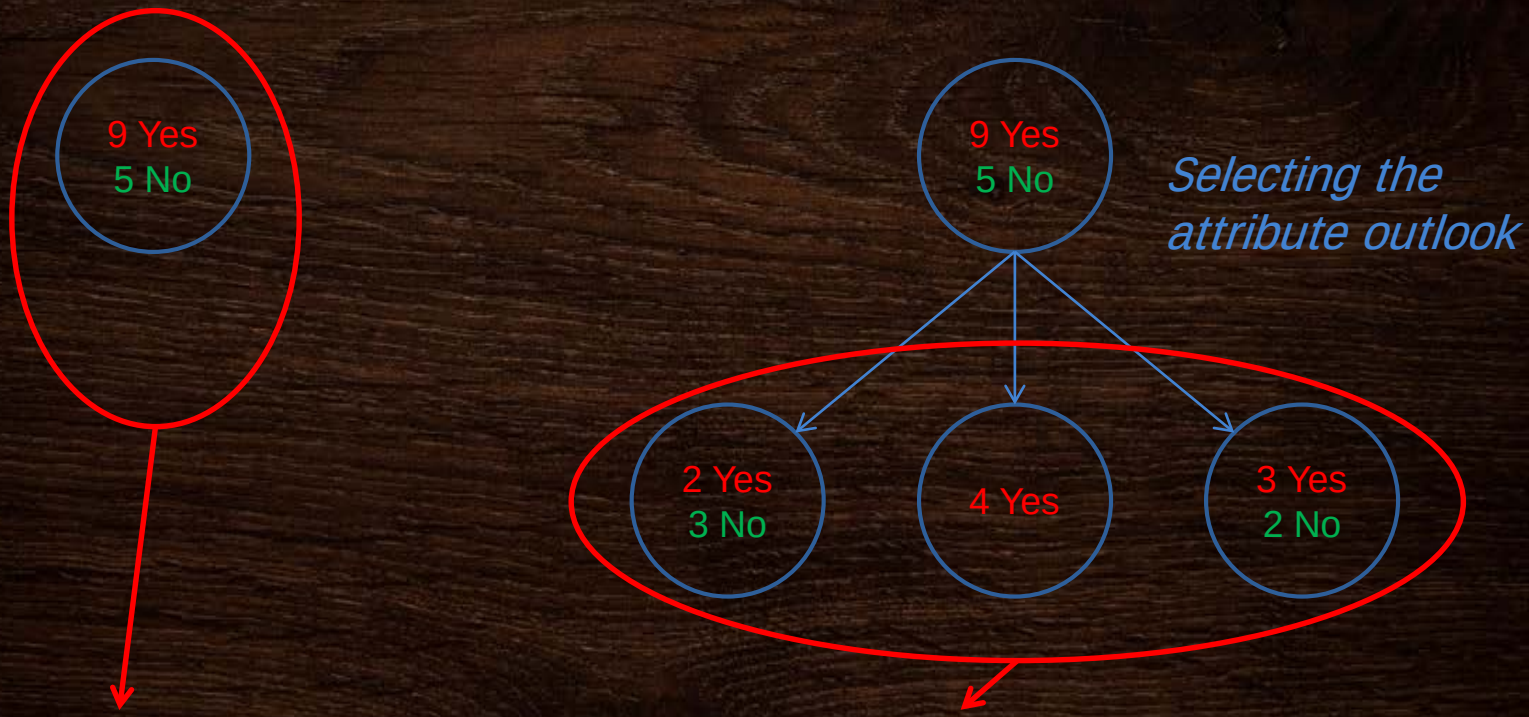
$$\mathbf{Entropy}(\mathbf{p}_1,\mathbf{p}_2,\mathbf{p}_3)=\frac{1}{\log 2}\left(-\mathbf{p}_1\log\mathbf{p}_1-\mathbf{p}_2\log\mathbf{p}_2-\mathbf{p}_3\log\mathbf{p}_3\right)$$

$$\mathbf{Entropy}(\mathbf{p}_1,\mathbf{p}_2,\mathbf{p}_3)=\mathbf{Entropy}(\mathbf{p}_1,\mathbf{p}_2+\mathbf{p}_3)+((\mathbf{p}_2+\mathbf{p}_3)\mathbf{Entropy}(\left(\frac{\mathbf{p}_2}{\mathbf{p}_2+\mathbf{p}_3}\right),\left(\frac{\mathbf{p}_3}{\mathbf{p}_2+\mathbf{p}_3}\right))$$

Property of the entropy

Calculation of information gain for an attribute

- Principle
  - Gain(attribut) = Info (before selecting the attribute) – Info (after selecting the attribute)



Selection of an attribute

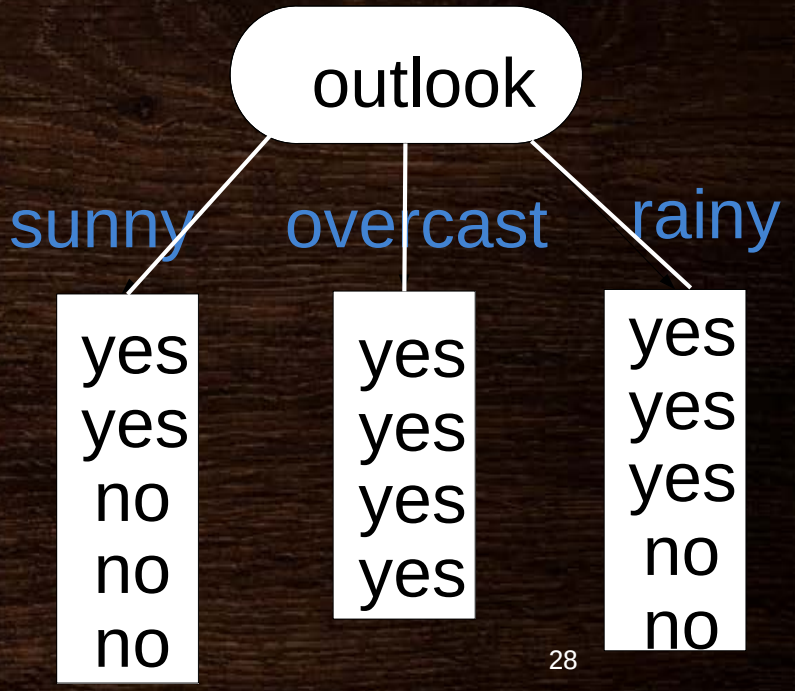
- Information gain before selecting Outlook
  - 15 elements
    - 9 Yes
    - 5 No

$$\textbf{Info}([9,5]) = \textbf{Entropy}\left(\frac{9}{14}, \frac{5}{14}\right)$$

$$\textbf{Info}([9,5]) = 0.940\ \textbf{bits}$$

Information gain after selecting Outlook

- What is my information gain if I choose the Outlook attribute?



Information gain after selecting Outlook

- What is my information gain if I choose the Outlook attribute?

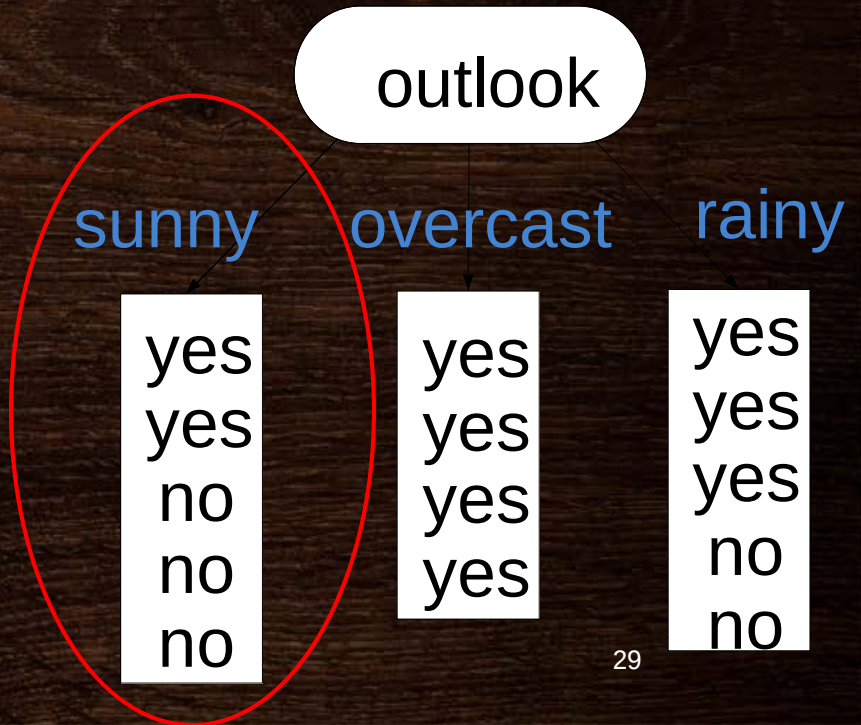
Outlook = "Sunny"

$$\mathbf{Info}([2,3]) = \mathbf{Entropy}\left(\frac{2}{5},\frac{3}{5}\right)$$

$$\mathbf{Info}([2,3]) = \mathbf{Entropy}(0.4,0.6)$$

$$\mathbf{Info}([2,3]) = \frac{1}{\log 2}\left(-0.4\log(0.4) - 0.6\log(0.6)\right)$$

$$\mathbf{Info}([2,3]) = 0.971\ \mathbf{bits}$$



Information gain after selecting Outlook

- What is my information gain if I choose the Outlook attribute?

Outlook = "Sunny"

$$\mathbf{Info}([2,3]) = \mathbf{Entropy}\left(\frac{2}{5},\frac{3}{5}\right)$$

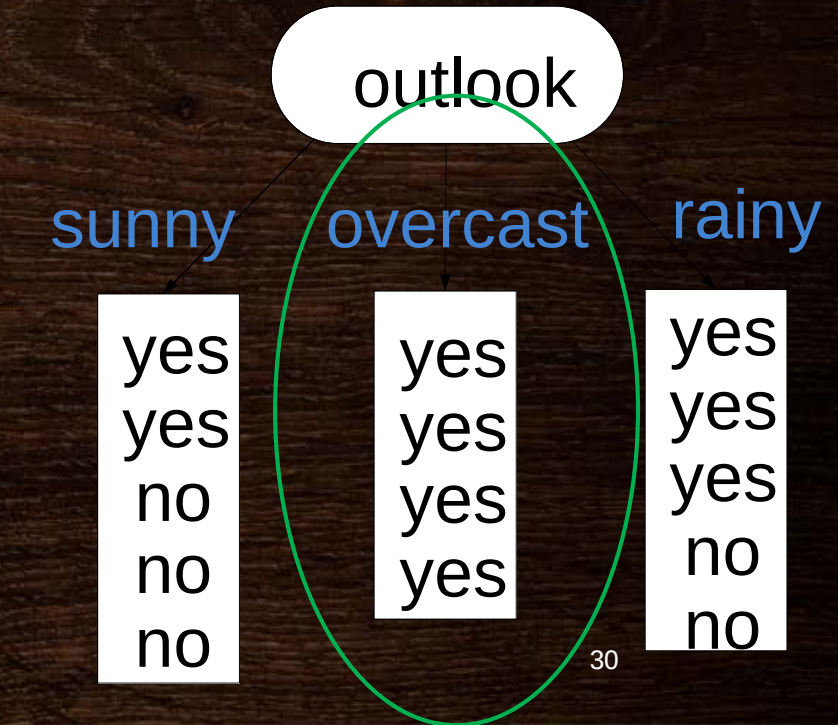
$$\mathbf{Info}([2,3]) = \mathbf{Entropy}(0.4,0.6)$$

$$\mathbf{Info}([2,3]) = \frac{1}{\log 2}\left(-0.4\log(0.4) - 0.6\log(0.6)\right)$$

$$\mathbf{Info}([2,3]) = 0.971\ \mathbf{bits}$$

Outlook = "Overcast"

$$\mathbf{Info}([4,0]) = 0.0\ \mathbf{bits}$$



Information gain after selecting Outlook

- What is my information gain if I choose the Outlook attribute?

Outlook = "Sunny"

$$\mathbf{Info}([2,3]) = \mathbf{Entropy}\left(\frac{2}{5},\frac{3}{5}\right)$$

$$\mathbf{Info}([2,3]) = \mathbf{Entropy}(0.4,0.6)$$

$$\mathbf{Info}([2,3]) = \frac{1}{\log 2}\left(-0.4\log(0.4) - 0.6\log(0.6)\right)$$

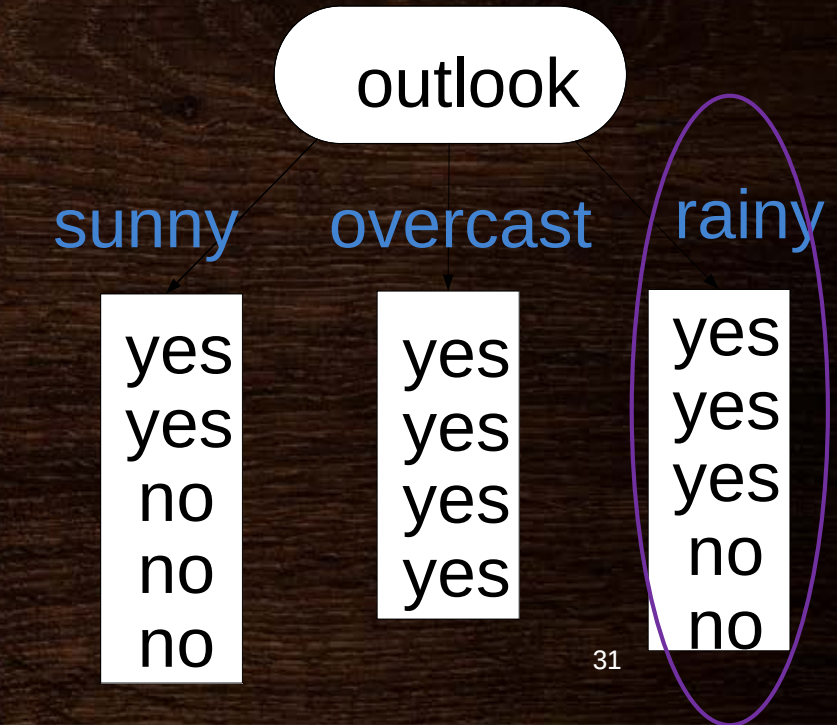
$$\mathbf{Info}([2,3]) = 0.971\ \mathbf{bits}$$

Outlook = "Overcast"

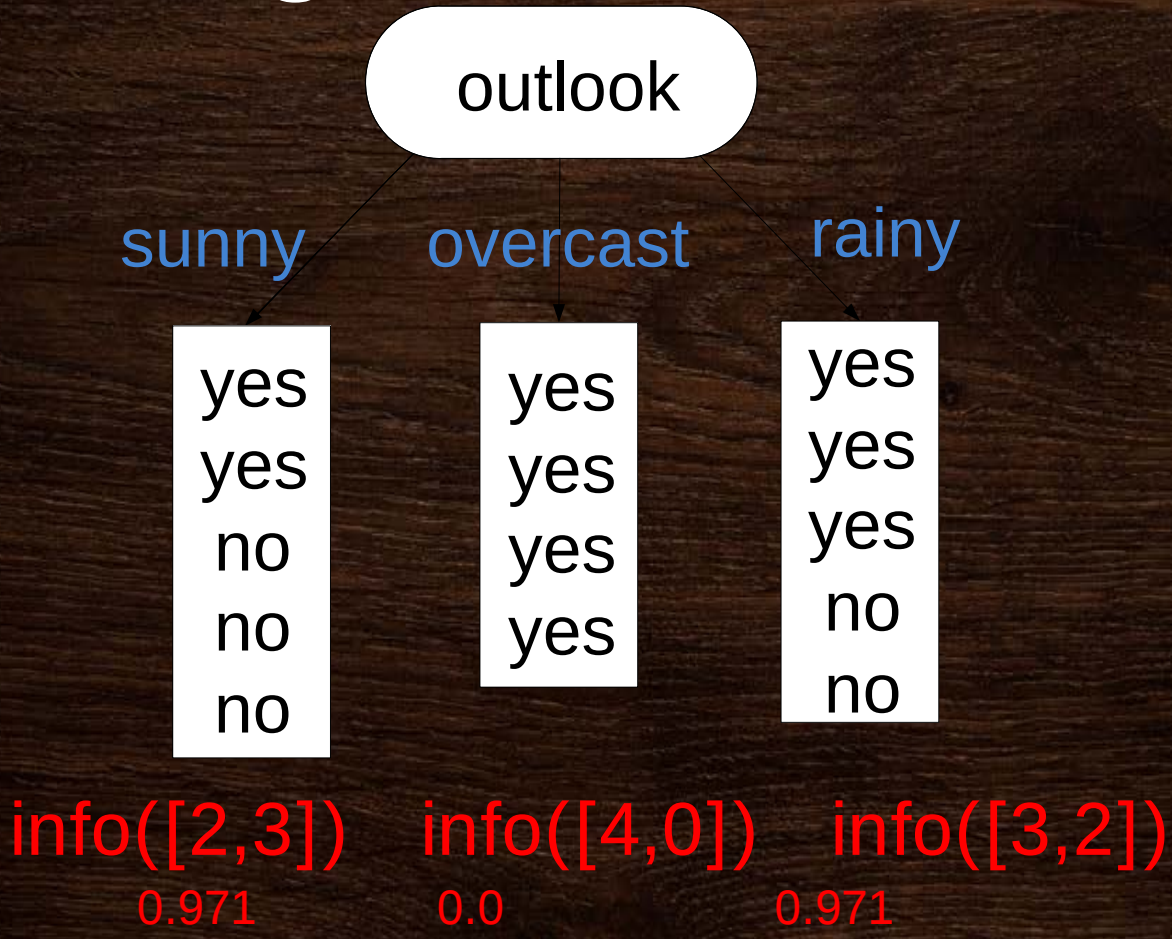
$$\mathbf{Info}([4,0]) = 0.0\ \mathbf{bits}$$

Outlook = "Rainy"

$$\mathbf{Info}([3,2]) = 0.971\ \mathbf{bits}$$



Information gain after selecting Outlook



Information gain after selecting Outlook

The value of the information for the tree after the selection of an attribute is the weighted sum of all the information of this attribute

Information for the whole tree after selecting *Outlook*:

$$\textbf{Info}([2,3],[4,0],[3,2]) = \frac{5}{14}\textbf{Info}([2,3]) + \frac{4}{14}\textbf{Info}([4,0]) + \frac{5}{14}\textbf{Info}([3,2])$$

Info([2,3],[4,0],[3,2]) = 0.693

Total information gain for Outlook

$$\textbf{gain}(\textbf{outlook}) = \textbf{Info}([9,5]) - \textbf{Info}([2,3],[4,0],[3,2])$$
$$\textbf{gain}(\textbf{outlook}) = 0.940 - 0.693$$
$$\textbf{gain}(\textbf{outlook}) = 0.247\ \textbf{bits}$$

Likewise

$$\textbf{gain}(\textbf{temperature}) = 0.029\ \textbf{bits}$$
$$\textbf{gain}(\textbf{humidity}) = 0.152\ \textbf{bits}$$
$$\textbf{gain}(\textbf{windy}) = 0.048\ \textbf{bits}$$

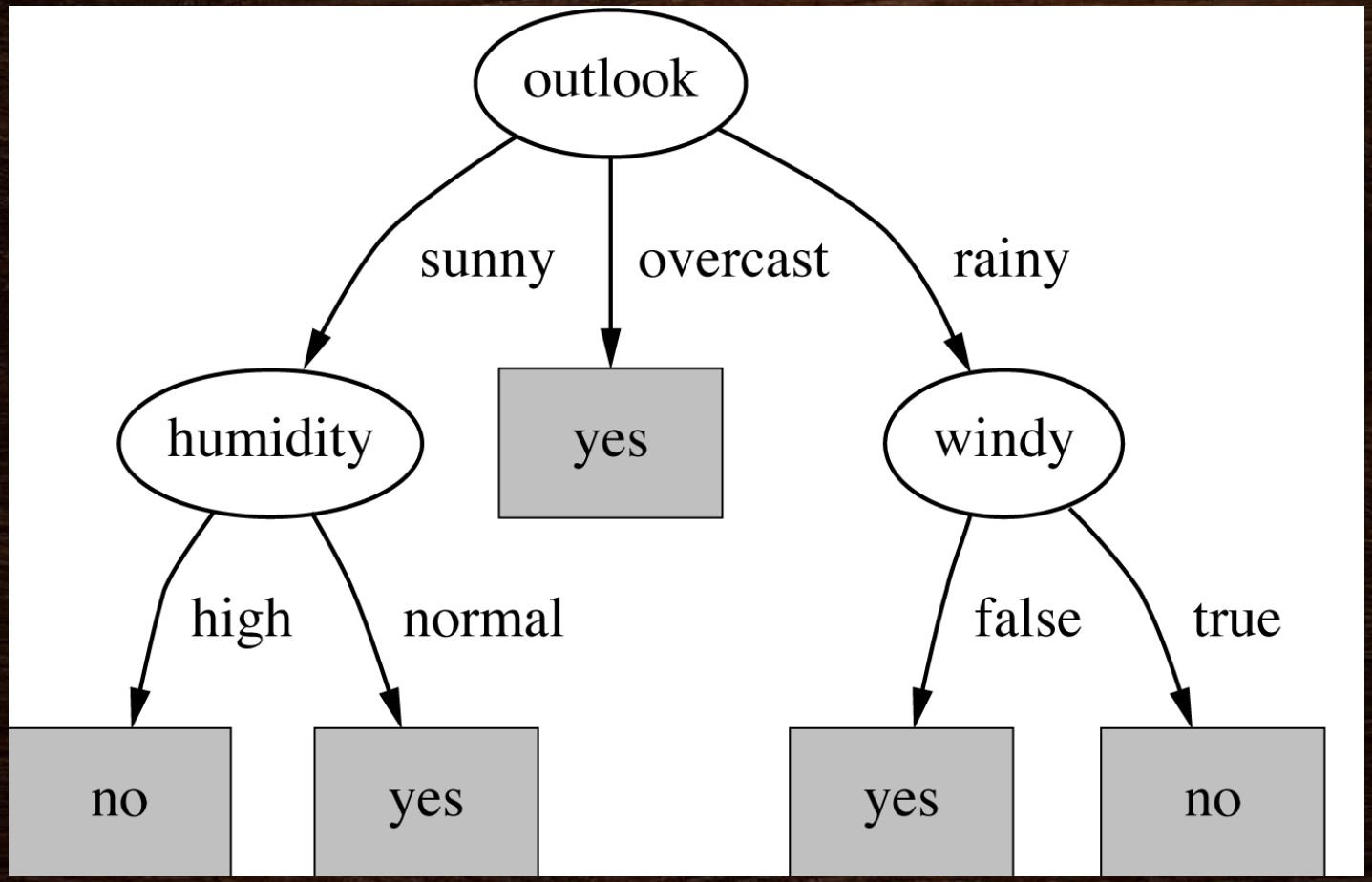
Outlook is chosen

Next step

- Selection of the second attribute
- We can examine:
  - Temperature, Humidity or Windy for Outlook = “sunny”
  - Gain(“Temperature”) = 0.571 bits
  - Gain(“Humidity”) = 0.971 bits
  - Gain(“Windy”) = 0.020 bits
- Etc.

Humidity is chosen

Final decision tree



Extensions of the algorithm

- How to treat:
  - Numeric attributes
  - Missing values
- How to simplify the model to avoid noises?
- How to tolerate noises?
- How to interpret the decision trees?

How to treat numeric attributes?

- Numeric attributes are transformed in ordinal/nominal values: discretization
- The values of the attributes are divided in intervals
  - The values of the attributes are ordered
  - Separations are put to create intervals / pure classes
  - We determine which values that imply a change of class
- This process is very sensitive to noise
- The number of classes should be controlled
  - Solution: Specify a minimum number of items per interval
  - Combine the intervals that define the same class

Missing values

- Ignore instances with missing values
  - Solution too general: missing values can be significant
- Ignore the attributes with missing values
  - Maybe not feasible
- Handle missing values as special values
  - Difficult because missing values can have special meaning
- Estimate the missing values
  - Gives the value of the most widely attribute to the considered attribute
  - Gives a value using various methods
    Example: Regression

Define the correct size of the tree

- There is a risk of overfitting: the model appears efficient (the average error is very small) but in reality it is not at all
  - The results are good but data learned on new data (not having been used to build the model) will be misclassified
- We need to find the smallest possible tree with the highest possible performance (the more a tree is small and the more it will be stable in future forecasts)
- On a comparable performance, we always prefer the simplest model, if we want to use this model to new data unknown

Model validation

- Learning: build the model on a first sample for which we know the value of the target (observed) variable (~ 70%)
- Test: Verification of the model on a second sample for which we know the value of the target variable: we compare the target to the value predicted by the model (~ 30%)
- Use the model with a new population for which we want to predict the target variable

Cross validation

- When the population is too small to extract a learning sample and another test is used to cross-validation (leave-one-out):
  - The population is divided into n samples of equal size
  - The first n-1 samples are used as a learning sample and the remainder as test sample. We obtain an error rate
  - We repeat the same operation by taking all possible n-1 samples as a learning sample and the remainder as test sample
  - We combine the n error rate obtained

Learning sample

Test sample

Advantages of classification trees

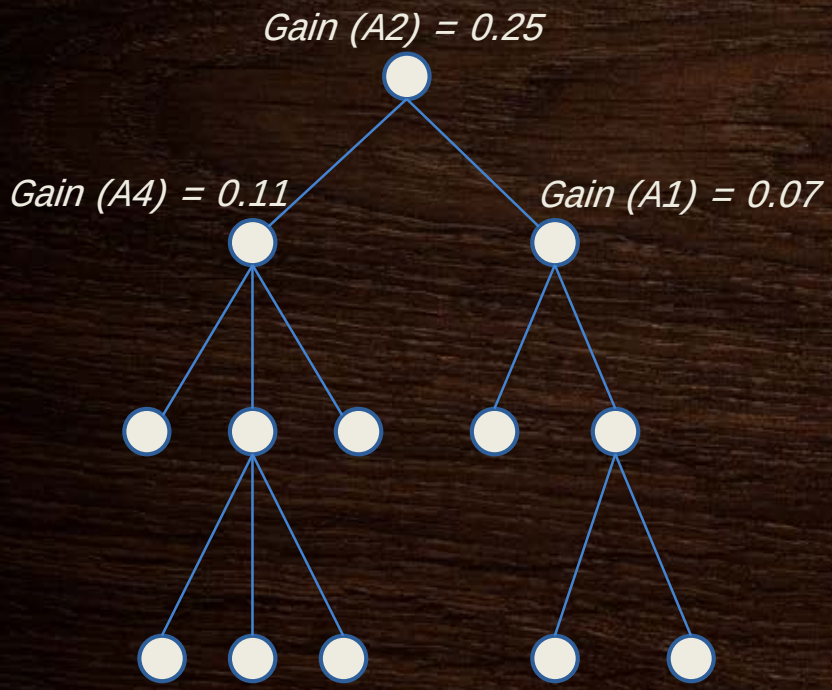
- Simple and easily interpretable rules (unlike neural networks for example)
- No need to recode heterogeneous data (especially CART)
- Treatment of missing values
- No model and no presupposition to meet (non-parametric method)
- Fast processing time

Drawbacks

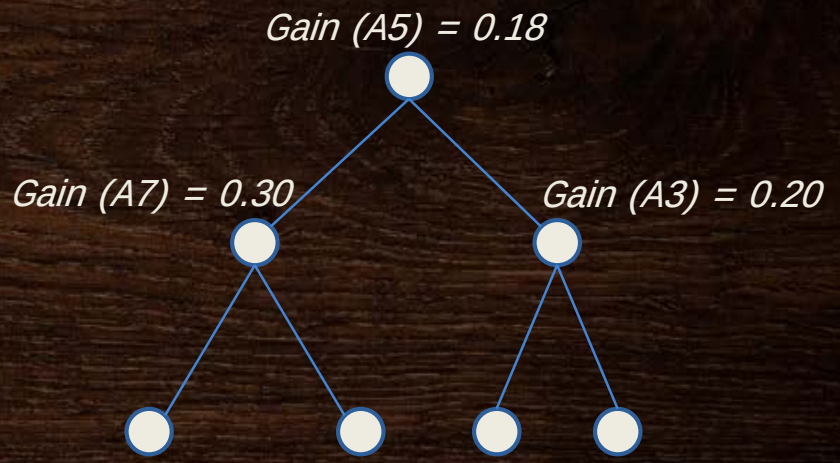
- The nodes of level n + 1 are highly dependent on those of level n (the modification of a single variable near the top can entirely change the tree)
- We always chose the best local attributes, the best global information gain is not at all guaranteed
- Learning requires a sufficient number of individuals
- Inefficient when there are many classes

Drawbacks

- We always chose the best local attributes, the best global information gain is not at all guaranteed



Solution chosen by the algorithm



Solution that should be chosen

Random Forests

Limits of decision trees

- No incremental: we must rebuild a new tree if we want to integrate new data
- Sensitive to small variations in the data (unstable)
- Result are strongly dependent on the order of attributes chosen to create the tree $\rightarrow$ generalization is difficult
- Find a minimum apparent error decision tree is generally a NP complete problem

Toward a combinaison of decision trees

- Net increased interest in trees thanks to techniques such as classifier aggregation methods (boosting, bagging, Random Forests, ...)
- Allows to take advantage of the combination of predictors
- Main idea: we use random to improve the performance of algorithms that are less performant

Some chronology

- First appearance of works on forest
  - 1992 [Ho 1992] first demonstrated the performance gain obtained when combining several classifiers that use decision trees in parallel combination
  - 1995 [Ho 1995] use the term Decision Forest to generally designate sets of decision trees,
  - 1996 [Breiman 1996] Bagging (Bootstrap Aggregating), introduced by Breiman
    - Overall principle of learning classifiers that can be used with any type of elementary classifier
    - The effectiveness is demonstrated mainly with decision trees

Some chronology

- 1996 [Freund 1996] Freund and Schapire proposed their Boosting algorithm, Adaboost , which is since a long time considered as the most efficient set of learning classifiers
- 1996 [Amit 1996] Amit and Geman present the Random Feature Selection principle. First randomization principles of decision trees

Bootstrap Aggregating

- Also known as BAGGING
  - Learning technique (classification and regression) used to
    - Improve stability
    - Reduce variance
    - Avoid overfitting (overfitting)
  - Suitable for any kind of model, but especially for decision trees
  - Principle
    - Given a training set D of size n, we generated new sets D_i of size n' <= n by uniformly sampling the examples of D with replacement
    - The M models are trained by using the m sets
    - The responses of the models are combined (average, vote, ...)

Random Feature Selection

- Or Random Tree
- Introduce par Amit and Geman in 1996
- Creation of sets of random trees for hand writing recognition
  - Too many features to consider $\rightarrow$ need to reduce complexity of decision trees
  - Also treated by Breiman that gave it its name

Random Feature Selection

- Idea
  - Randomness is introduced in the choice of partitioning rules in each node of the tree
  - Each rule is no longer chosen from the set of all available features, but from a subset of these feature
  - Principle
    - Select by random sampling without replacement a number K of features and choose the best possible rule among these K features
    - The corresponding induction algorithm is Random Tree

Definition

- Léo Breiman definition (2001)
  - A Random Forest classifier is composed of a set of elementary classifiers type of decision trees, denoted

$$\{ h(x, \Theta_k), \quad k = 1, ...L \}$$

  - Where $\{\Theta_k\}$ is a family of independent and identically distributed random vectors, and in which each tree participate in the vote in the most popular class for a given input x

Definitions

- Strength
  - Reliability of the forest
- Correlation
  - Correlation rate of elementary classifiers among themselves
- The error rate of the forest is even lower than the set of elementary classifiers reliable
- The error rate of the forest is even weaker than the elementary classifiers are uncorrelated in terms of prediction
- The challenge therefore lies in the fact of producing a set of the most efficient possible trees individually and be less correlated

Principle of random forests

- Advantages
  - Variance reduction (influence of data)
  - Easy to implement
  - Naturally parallelizable
- Drawbacks
  - Greater computation time
  - Interpretability reduced
- Introduction of randomness: making models (trees) more independent among themselves

Forest-RI

- or Random Forests - Random Input
- Introduced by Breiman in 2001
- Reference
- Use 2 randomization principes
  - Bagging
  - Random Feature Selection

Forest-RI - Algorithm

Algorithme 2 ForestRI

Entrée : T l'ensemble d'apprentissage
Entrée : L le nombre d'arbres dans la forêt
Entrée : K le nombre de caractéristiques à sélectionner aléatoirement à chaque nœud
Sortie : $foret$ l'ensemble des arbres qui composent la forêt construite

1: **pour** l de 1 à L **faire**
2: $T_l \leftarrow$ ensemble bootstrap, dont les données sont tirées aléatoirement (avec remise) de T
3: $arbre \leftarrow$ un arbre vide, *i.e.* composé de sa racine uniquement
4: $arbre.racine \leftarrow RndTree(arbre.racine, T_l, K)$
5: $foret \leftarrow foret \bigcup arbre$
6: **retour** $foret$;

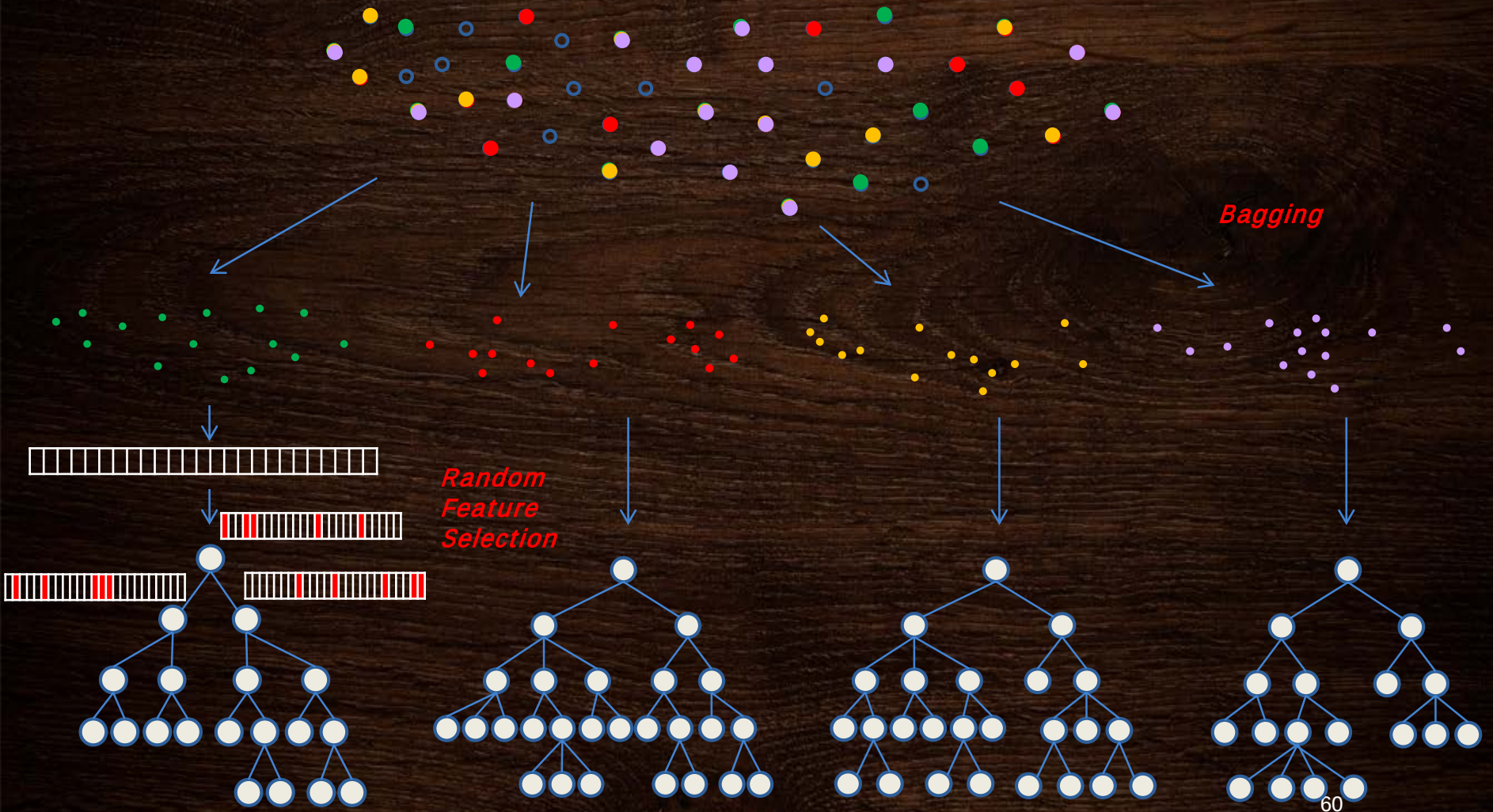
Forest-RI - Algorithm

Algorithme 3 RndTree

Entrée : n le nœud courant
Entrée : T l'ensemble des données associées au nœud n
Entrée : K le nombre de caractéristiques à sélectionner aléatoirement à chaque nœud
Sortie : n le même nœud, modifié par la procédure

1: **si** n n'est pas une feuille **alors**
2: $C \leftarrow K$ caractéristiques choisies aléatoirement
3: **pour tout** $A \in C$ **faire**
4: Procédure CART pour la création et l'évaluation (critère de Gini) du partitionnement produit par A, en fonction de T
5: $partition \leftarrow$ partition qui optimise le critère Gini
6: $n.ajouterFils(partition)$
7: **pour tout** $fils \in n.noeudFils$ **faire**
8: $RndTree(fils, fils.donnes, K)$
9: **retour** n

Bagging and Random Feature Selection



Important data

- Data to parametrized
  - Number of trees in the forest
  - Depth of the trees
  - Number of features considered to build each tree
  - Choice of the features among all in initial features

Forest-RC (or Random Forests - Random Combinations)

- Variant of Forest-RI, Defined Breiman in 2001
- Overcomes some Forest-RI limits when there are few features
  - Choose for K a relatively large part of all of these features can lead to a loss of diversity in the set
- Instead of selecting the best feature among K chosen at random, calculate K linear combinations of features F (with F not necessarily equal to K)

Forest-RC (or Random Forests - Random Combinations)

- Principle
  - Randomly choose F characteristics
  - Combine them with the sets of K coefficients also randomly chosen in the interval [-1, 1]
  - Once the K linear combinations are generated, choose the one that offers the best partitioning
- More suitable than Forest-RI to problems whose feature space is small
- But the performance gain is often not significant enough for algorithm harder to control that Forest-RI

Extremely Randomized Trees

- Or Extra-Trees
- Introduced by Geurts in 2006
- Principle
  - Suppression of the principle of Bagging
    - Use the whole train set for each classifier
    - Reduce the bias
  - Reinforcement of random factor in the induction process of the trees
  - Reduce the variance

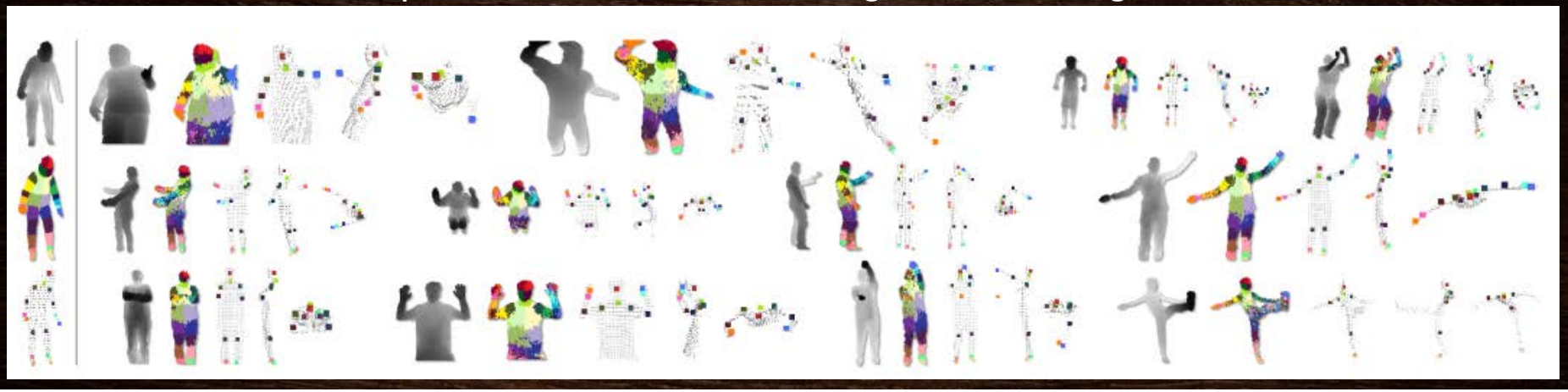
Other algorithms

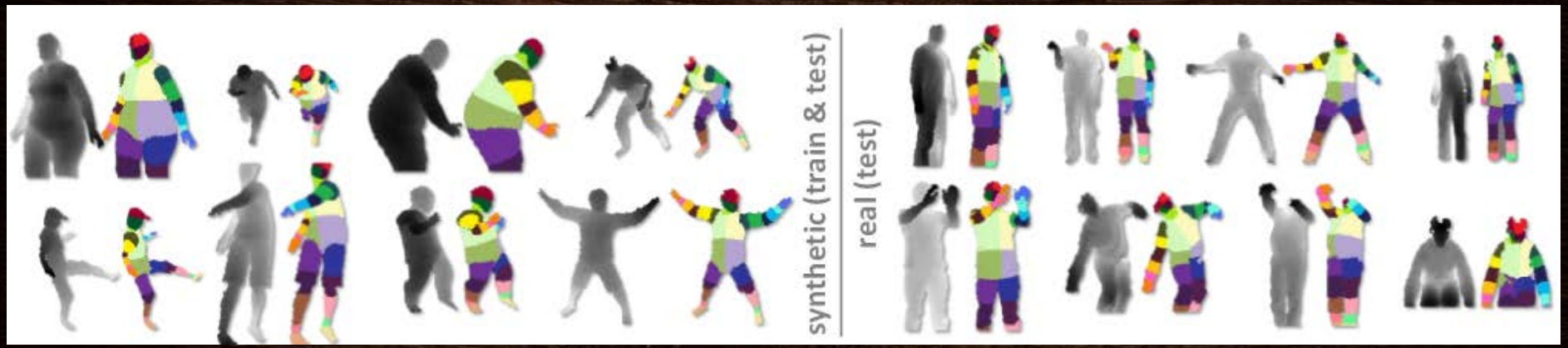
- Extremely Randomized RF (Geurts 2006)
- Weighted Voting Random Forest (Robnik-Sikonja 2004)
- Balanced Random Forests (Chen 2004)
- Weighted Random Forests (Chen 2004)
- Rotation Forest (Rodriguez 2006)
- Probabilistic Random Forest (Breitenbach 2002)
- On-line Random Forest,
- Hierarchical Random Forest
- "Meta" Random Forests (Boinee 2005)
  - Bagged Random Forest (BgRF)
  - AdaBoosted Random Forest (ABRF), …

Some concrete examples

Implementation of RF in the Kinect

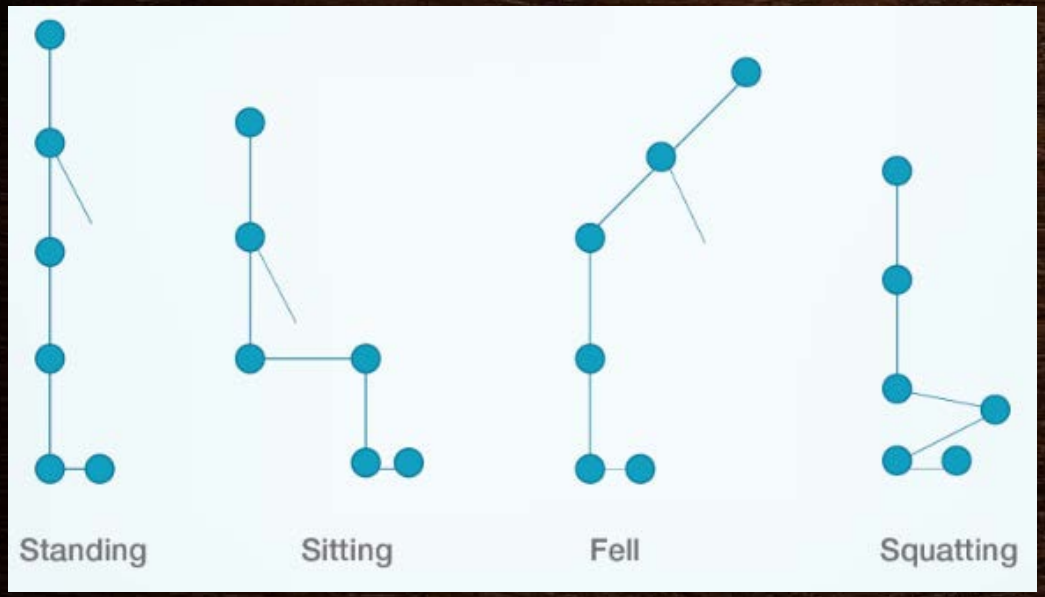
- Real-Time Human Pose Recognition in Parts from Single Depth Images (2011)
  - 3 trees, depth of de 20, 300 000 images for training





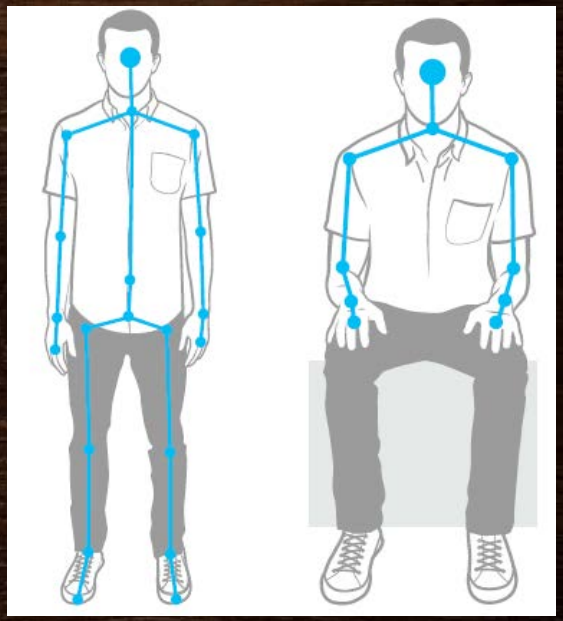
Static position recognition

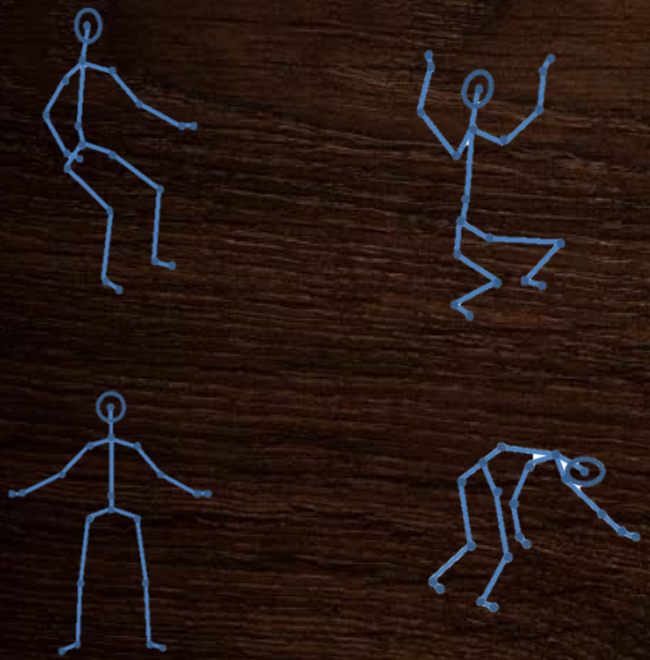
- Field of recognition of positions and movements from the Kinect
  - PFE Stéphanie Lopez – Perside Gbehounou



Static position recognition

- From the Kinect, skeleton
  - 20 joints





Static position recognition

- Learning set
  - 11 classes
  - 1690 examples

Positions		Annotations
Debout	Main droite levée	0
	Main gauche levée	1
	2 mains levées	2
	2 mains baissées	3
		4
	Main droite levée	5
	Main gauche levée	6
	2 mains levées	7
	2 mains baissées	8
Accroupi	Main droite levée	9
	Main gauche levée	
	2 mains levées	
	2 mains baissées	10

Positions

Static position recognition

- Question to ask
  - Which vector use to create the forest?
- First idea
  - Coordinates X, Y and Z of each joint
  - Vector with 60 features
- Is it sufficient?

Static position recognition

- To build the forest, we take, to select each node, $\sqrt{N}$ features (N being the total number of features)
- If we consider 60 features in total, we obtain between 7 and 8 features
  - Not enough
- We must therefore find out how to ways to increase the feature vector
  - These new features may not be correlated
  - How to choose them?

Static position recognition

- We consider all the distances between each joint
  - 20 * 19 additional values
- We add all the possible angles between the joints
  - 20 * 19 * 18 additional values
- A total vector composed of 7280 features

Action recognition

- Some examples of actions we want to recognize
  - Talking on the phone
  - Writing in a computer
  - Drinking water
  - Writing on the board
  - Etc.
- Current thesis Aya ALY

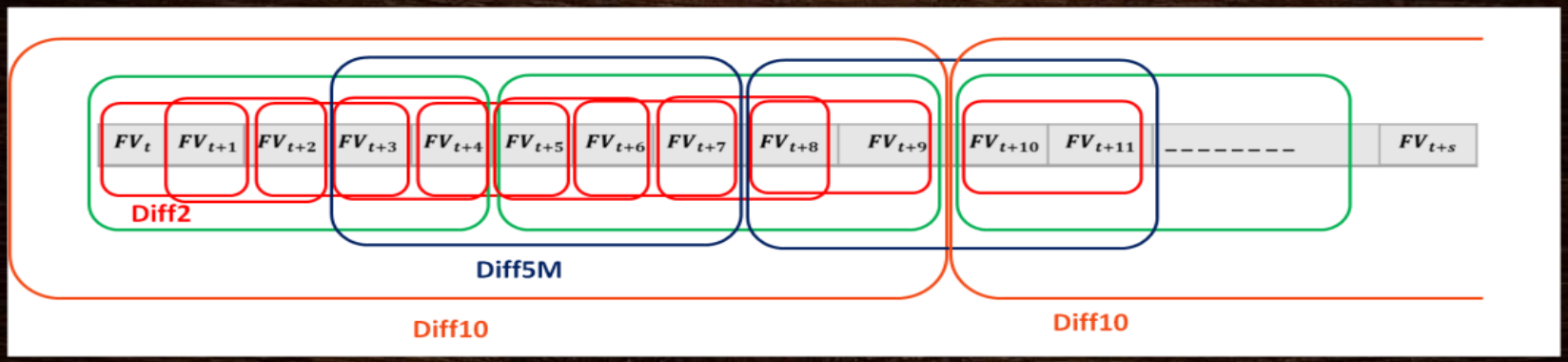
Action recognition

- Problem much more complex that static position recognition
  - We can dring
    - Standing
    - Sitting
    - Quickly
    - Slowly

- We can’t say that an action is a suite of fixed static positions
  - Possibility to have parasite movements
- Which vector to use? Still an open problem

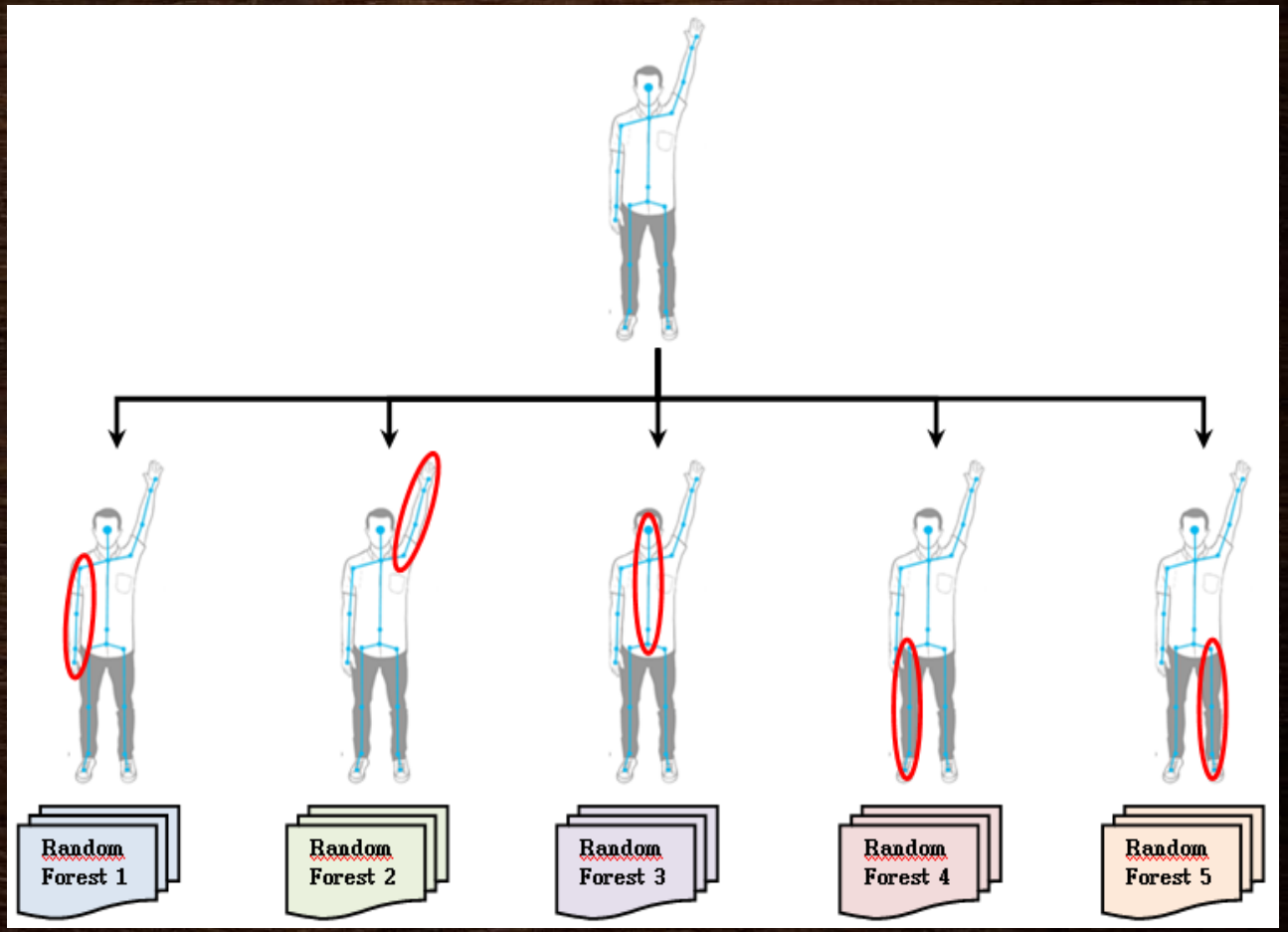
Action recognition

- First step
  - Use of sliding windows
  - Average of the values of angles and distances
    - For 2 consecutives windows
    - For 5 consecutives windows
    - For 10 consecutives windows
  - Technique issue from image processing to capture time varying positions



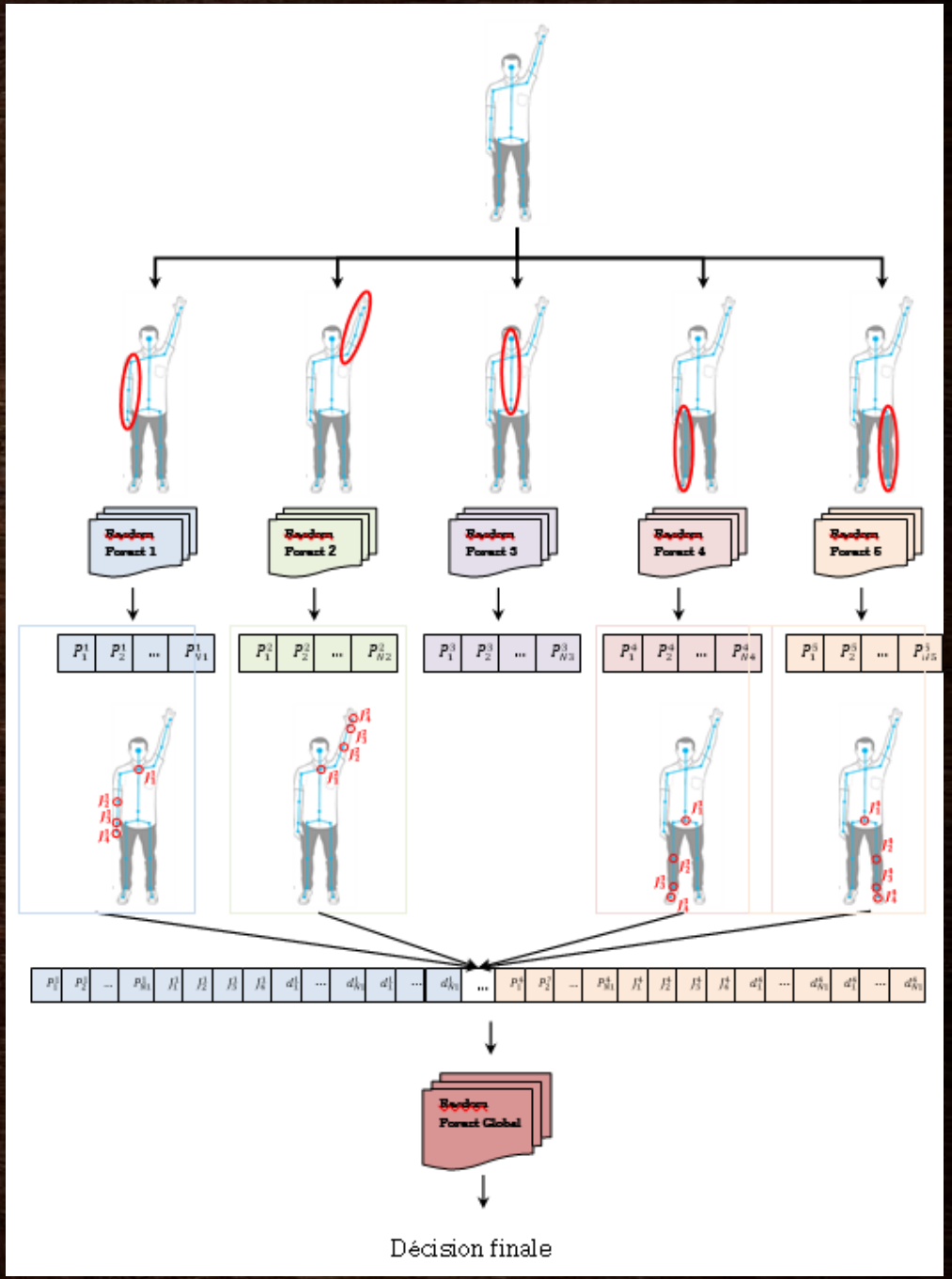
Action recognition

- Strategic bias: keep only data from the skeleton to have the most simplest possible data
- Use Hierarchical Random Forest
- How can/must we combine the sub-forests?



Action recognition

- Hierarchical RF



Sensors

- Partnership with the team Rainbow, platform Wcomp
- Working hypotheses
  - Set of available services and devices with their metadata
  - The infrastructure is dynamic
    - These objects can appear and disappear opportunistically
- Goal
  - Learn "their behavior" and the influence or impact of this behavior on the final configuration of the environment

Sensors

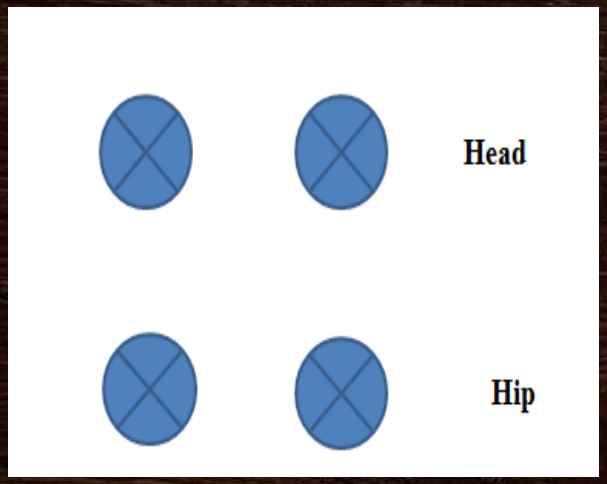
- Purpose?
  - Activity monitoring at home
    - We want to learn from communicating objects such as a telephone, a phone charger, a mattress with sensors, a Kinect, …

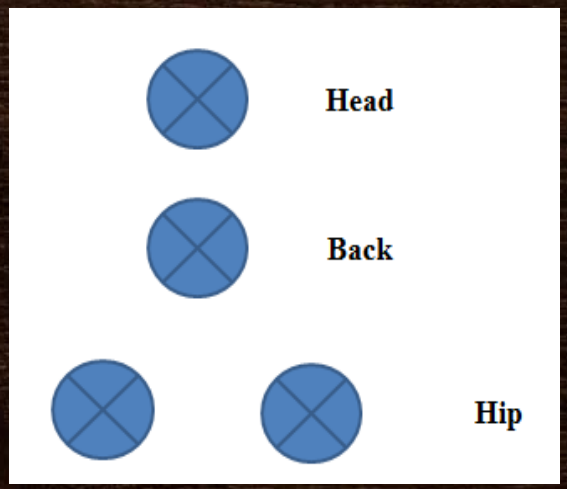
Sensors

- Preliminary work
  - A mattress provided with sensors
  - Different positions identified on the mattress (lying in the middle, lying n the right side of the matress, etc.)
  - Final objective = monitoring of babies lying on a mattress
  - Starting hypothesis
    - 2 possible dispositions for the sensors
  - Goal
    - Find the best disposition

Sensors

- Vector representing the data
  - Data issues from the 4 sensors
- Problem
  - The vector is too small : the random forests cannot be efficient





Sensors

- Solution
  - Reinforce the feature vector with new uncorrelated features
  - We add average 2 by 2 and 3 by 3 from the various sensors
  - A vector with 40 values obtained
- Still not enough to have consistent results
  - No logic in the created forest, no convergence when adding trees
  - SVM offer more valid results in this kind of problems

Sensors

- Two possible strategies for learning from different components
  - For each component, we use a forest to learn (or SVM in the case of mattresses, for example) and we then combine predictions of each classifier in another global classifier to have an overall prediction
  - For all components, we learn with a single classifier (eg RF)

Sensors

- Open questions to be answered
  - What strategy is the most effective?
  - What happens if we learn with 5 components and one component disappears?
  - What will happen if we learned with Kinect, mattresses, clock, lamp and a second clock appears?
  - What will happen if we learned with the IPhone 5 ans the Iphone 6 appears?
  - How can the forest be updated as quickly as possible?
  - Etc.

Short text classification

- Thesisse (Ameni Bouaziz)
- Collaboration with a company that provides a tool to give questionnaires to a large number of people and who wants to combine the responses
- After each question, answers are collected and classified by the system
- A driver modifies this classification by hand while the next question is asked
- Optimizing the classification algorithm is an crucial point

Valider | Deselect | Run clustering | Answers : 21 (new : 0) | Freeze groups | Add new answers

n°28 : Que vous manque-t-il pour mieux faire ? Pour l'Entreprise Régionale CentreLoire

Coopération entre services

- Une meilleure cohésion @
- la confiance @
- mutualiser @
- Coopération entre services @
- Des fonctions supports au service des opérationnels @
- le travail d'équipe @
- Décloisonner @
- Une culture d'entreprise (Fierté d'appartenir à la famille Lyonnaise des Eaux) @

Autre

- adhésion d'une partie des agents de terrain et d'une partie d'encadrants @
- un service client performant @
- audit des contrats déficitaires @
- Qu'on accepte qu'on a des choses que lesquelles on doit progresser @

Homogénisation de pratiques

- Homogénisation de pratiques @
- simplification du reporting interne @

Coopération entre services(38%)

Autre(19%)

Homogénisation de pratiques(14%)

Une stratégie claire(9%)

Un peu plus de moyens(9%)

Un poids dans les décisions(9%)

+++

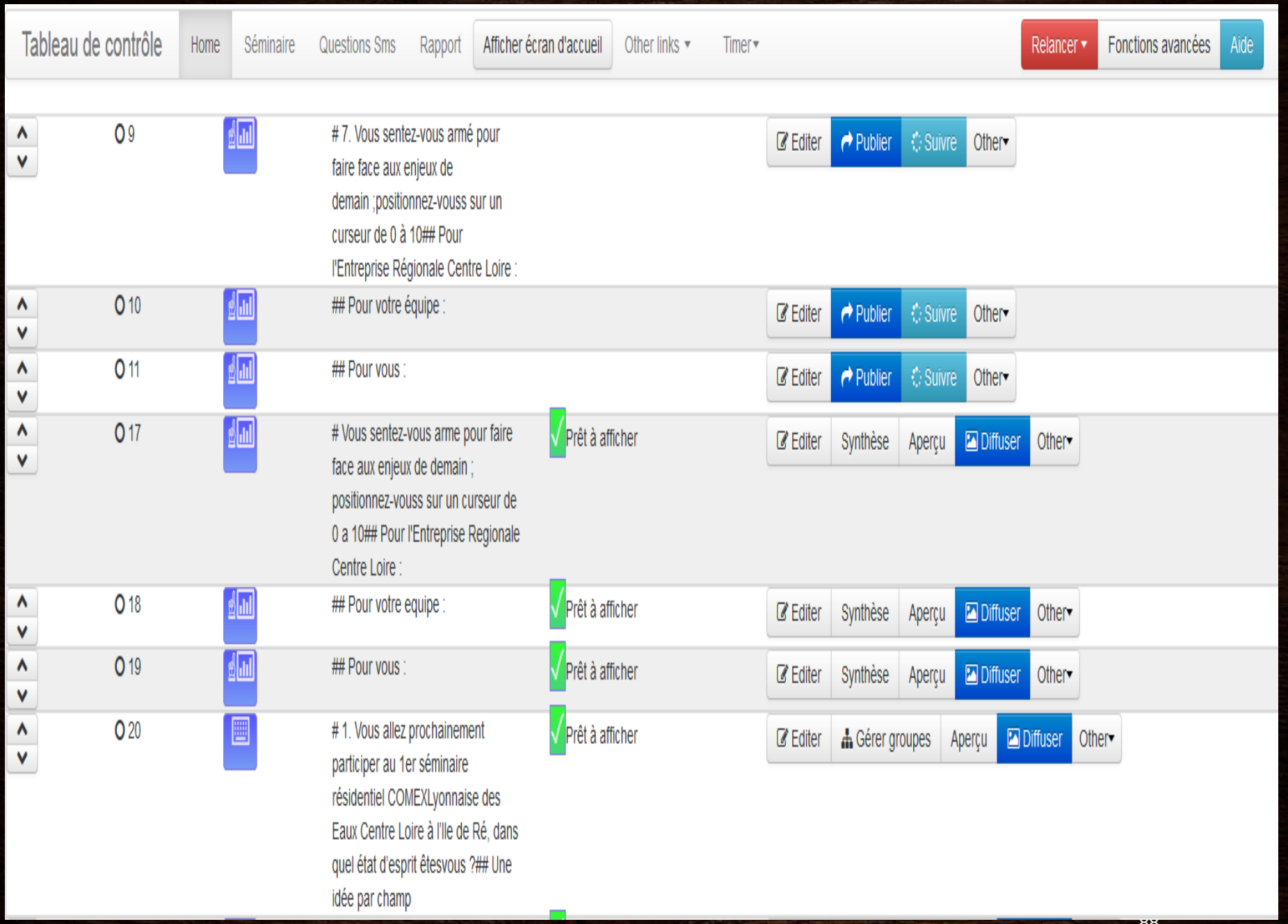


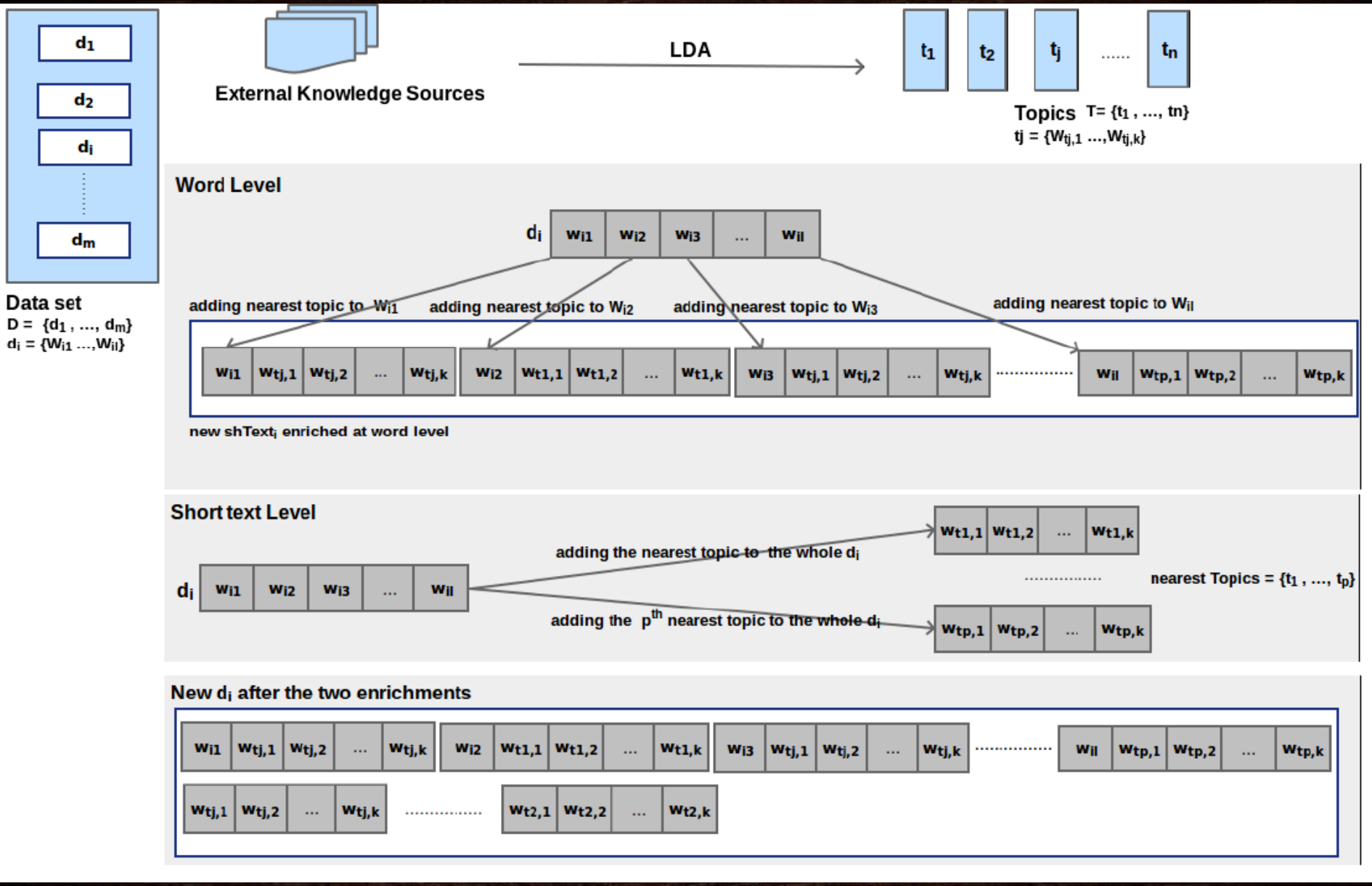
Tableau de contrôle
Home
Séminaire
Questions Sms
Rapport
Afficher écran d'accueil
Other links
Timer
Relancer
Fonctions avancées
Aide
9
7. Vous sentez-vous armé pour faire face aux enjeux de demain ;positionnez-vouss sur un curseur de 0 à 10## Pour l'Entreprise Régionale Centre Loire :
Editer
Publier
Suivre
Other
10
Pour votre équipe :
Editer
Publier
Suivre
Other
11
Pour vous :
Editer
Publier
Suivre
Other
17
Vous sentez-vous arme pour faire face aux enjeux de demain ; positionnez-vouss sur un curseur de 0 a 10## Pour l'Entreprise Regionale Centre Loire :
Prêt à afficher
Editer
Synthèse
Aperçu
Diffuser
Other
18
Pour votre equipe :
Prêt à afficher
Editer
Synthèse
Aperçu
Diffuser
Other
19
Pour vous :
Prêt à afficher
Editer
Synthèse
Aperçu
Diffuser
Other
20
1. Vous allez prochainement participer au 1er séminaire résidentiel COMEXLyonnaise des Eaux Centre Loire à l'Ile de Ré, dans quel état d'esprit êtesvous ?## Une idée par champ
Prêt à afficher
Editer
Gérer groupes
Aperçu
Diffuser
Other

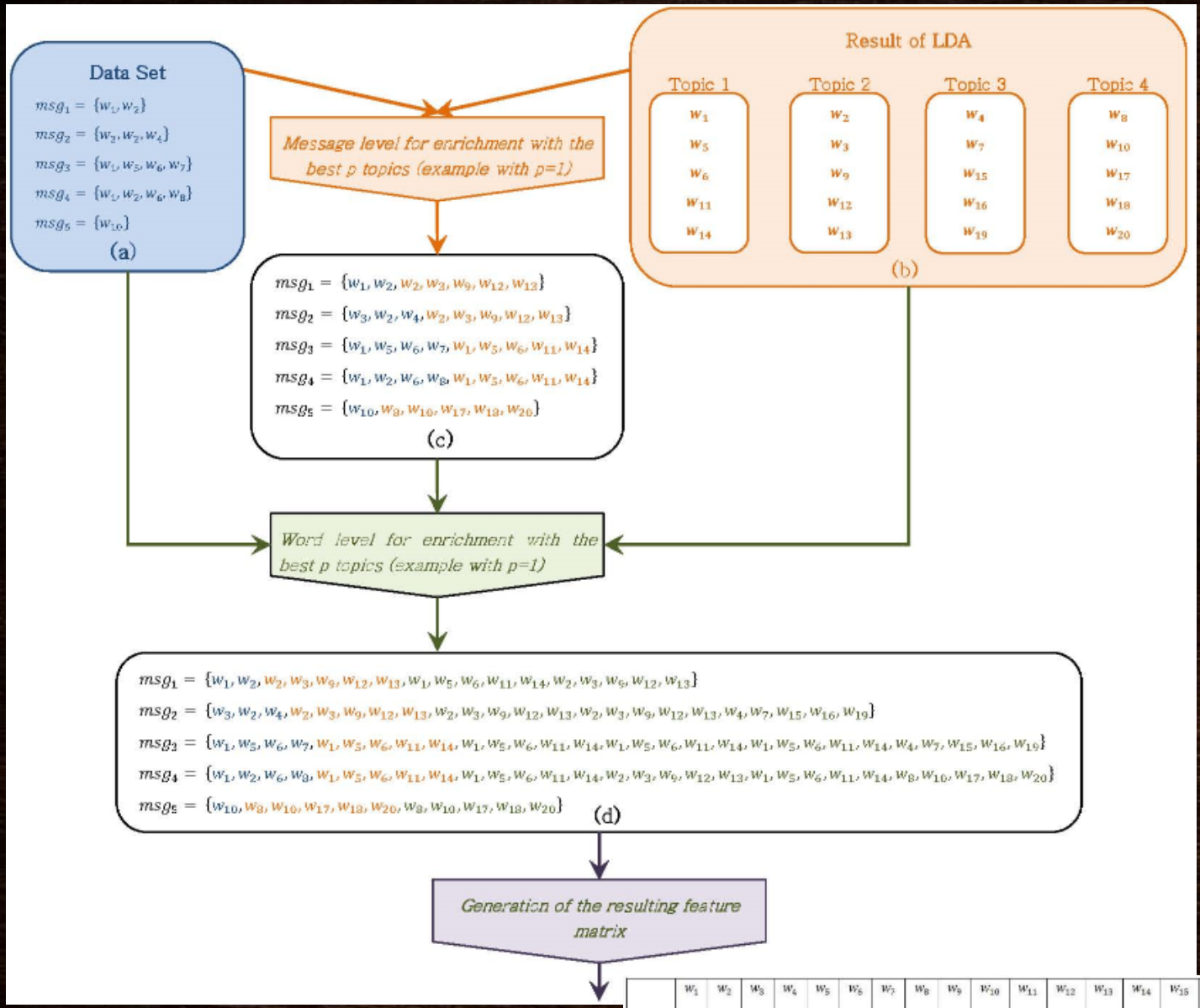
Short text classification

- Problem
  - These are ultra short-texts
  - Very few information to represent each sentence
  - Taking into account the context related to the words is even more important
- Vector used for the forest
  - We consider all the words presents in all the questions ans answers
  - We put 0 if this word is not used in the sentence and +1 each time the word is present in the sentence → vecteur quasi vide

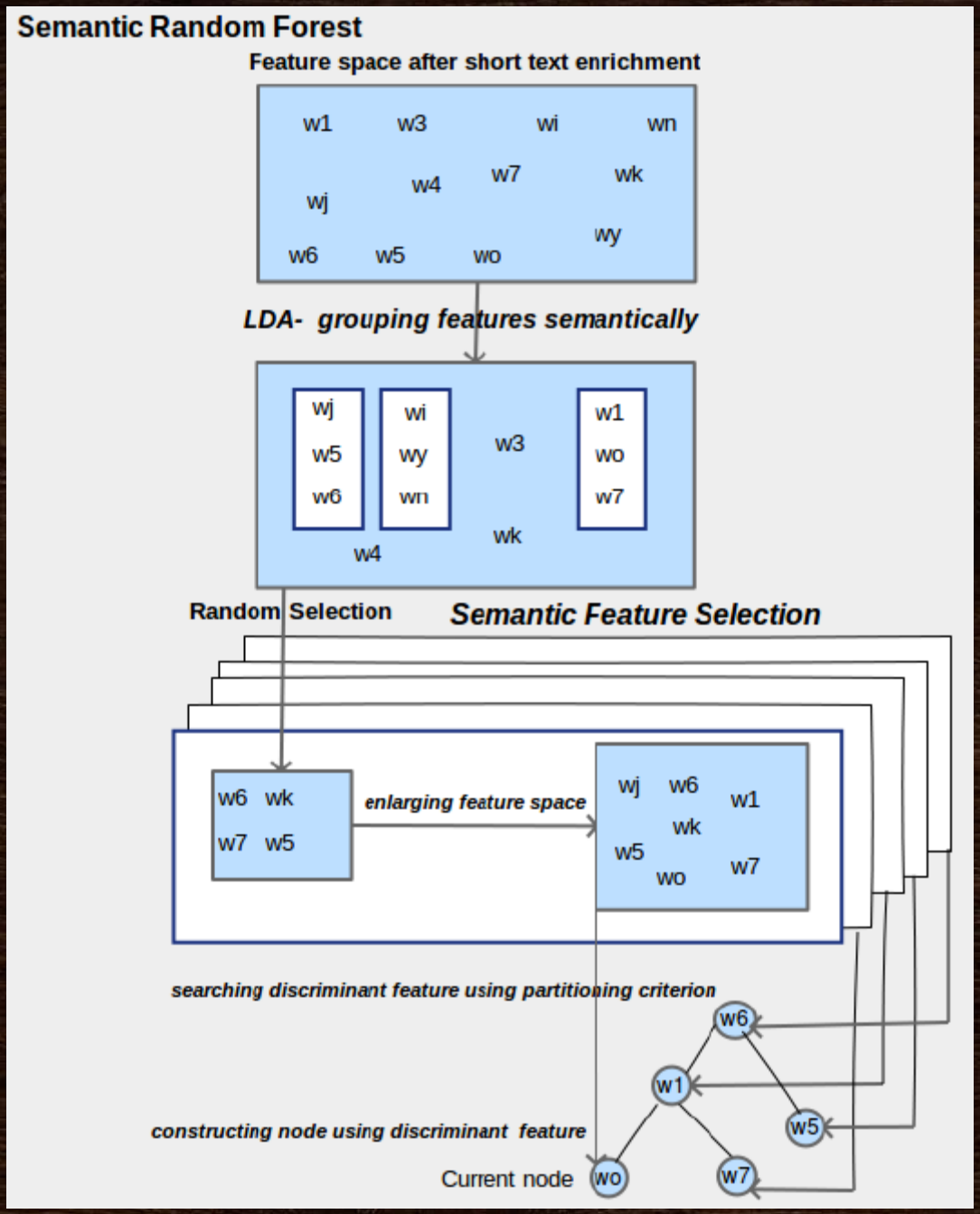
Short text classification

- This solution doesn’t take into account the context in which the sentence has been said
- Very little work combine knowledge and random forests
- One possible solution
  - Enrich the vecto with words that belong to the same context
  - Use an algorithm which groups in topics the words present in a corpus (LDA)
  - We enrich each vector with the words of the k topics the most closed semantically with the words of the sentence


d1
d2
di
dm
Data set
D = {d1 , ..., dm}
di = {Wi1 ...,Wil}
External Knowledge Sources
LDA
t1
t2
tj
.....
tn
Topics T= {t1 , ..., tn}
tj = {Wtj,1 ...,Wtj,k}
Word Level
di
wi1 wi2 wi3 ... wil
adding nearest topic to Wi1
adding nearest topic to Wi2
adding nearest topic to Wi3
adding nearest topic to Wil
wi1 wtj,1 wtj,2 ... wtj,k
wi2 wt1,1 wt1,2 ... wt1,k
wi3 wtj,1 wtj,2 ... wtj,k
wil wtp,1 wtp,2 ... wtp,k
new shTexti enriched at word level
Short text Level
di
wi1 wi2 wi3 ... wil
adding the nearest topic to the whole di
adding the pth nearest topic to the whole di
wt1,1 wt1,2 ... wt1,k
wtp,1 wtp,2 ... wtp,k
nearest Topics = {t1 , ..., tp}
New di after the two enrichments
wi1 wtj,1 wtj,2 ... wtj,k
wi2 wt1,1 wt1,2 ... wt1,k
wi3 wtj,1 wtj,2 ... wtj,k
wil wtp,1 wtp,2 ... wtp,k
wtj,1 wtj,2 ... wtj,k
wt2,1 wt2,2 ... wt2,k

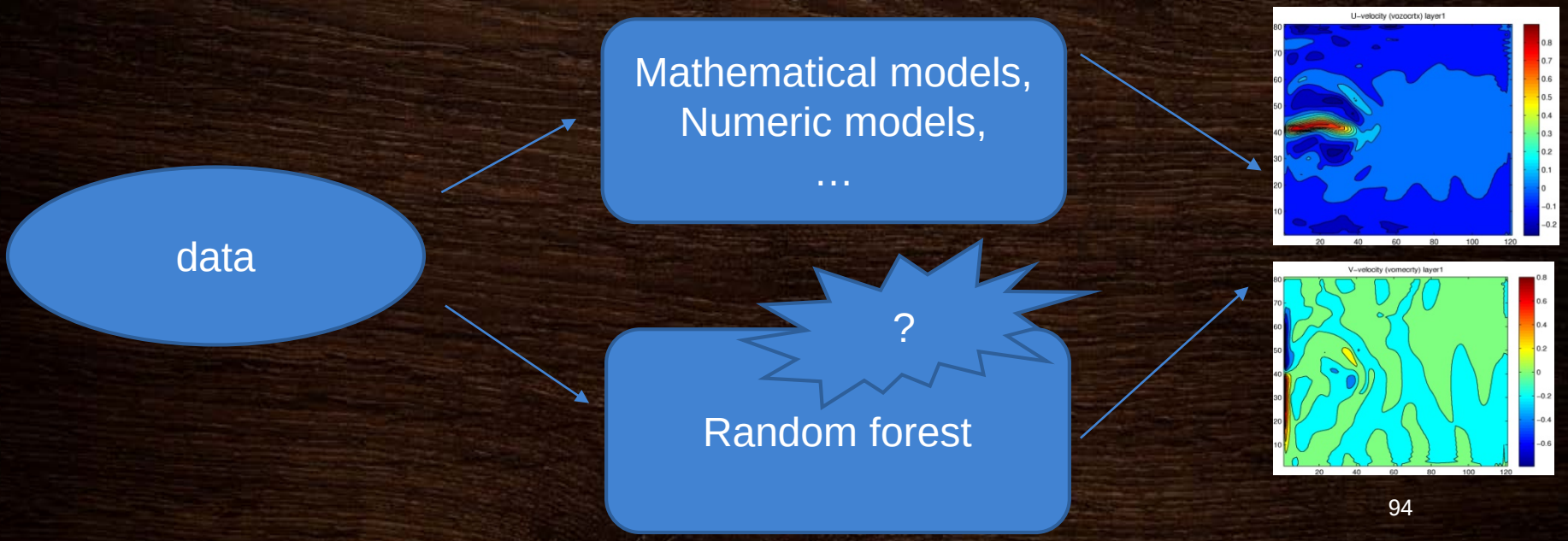


	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}	w_{11}	w_{12}	w_{13}	w_{14}	w_{15}	w_{16}	w_{17}	w_{18}	w_{19}	w_{20}
msg_1	2	3	2	0	1	1	0	0	2	0	1	2	2	1	0	0	0	0	0	0
msg_2	0	4	4	2	0	0	1	0	3	0	0	3	3	0	3	1	0	0	1	0
msg_3	5	0	0	1	5	5	2	0	0	0	4	0	0	4	1	1	0	0	1	0
msg_4	4	2	1	0	3	4	0	2	1	1	3	1	1	3	0	0	1	1	0	1
msg_5	0	0	0	0	0	0	0	2	0	3	0	0	0	0	0	0	2	2	0	2

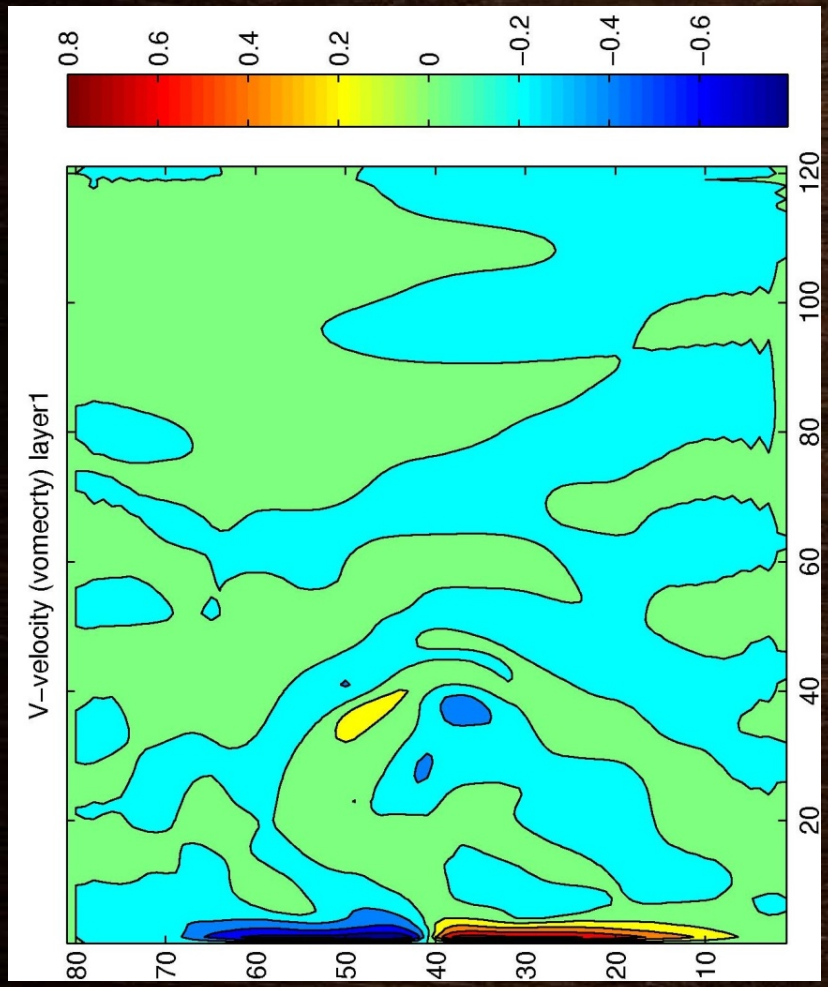

Semantic Random Forest
Feature space after short text enrichment
w1
w3
wi
wn
w4
w7
wk
wj
wy
w6
w5
wo
LDA- grouping features semantically
wj
w5
w6
wi
wy
wn
w3
w1
wo
w7
wk
w4
Random Selection
Semantic Feature Selection
w6
wk
w7
w5
enlarging feature space
wj
w6
w1
wk
w5
w7
wo
searching discriminant feature using partitioning criterion
w6
w1
w5
constructing node using discriminant feature
Current node
wo
w7

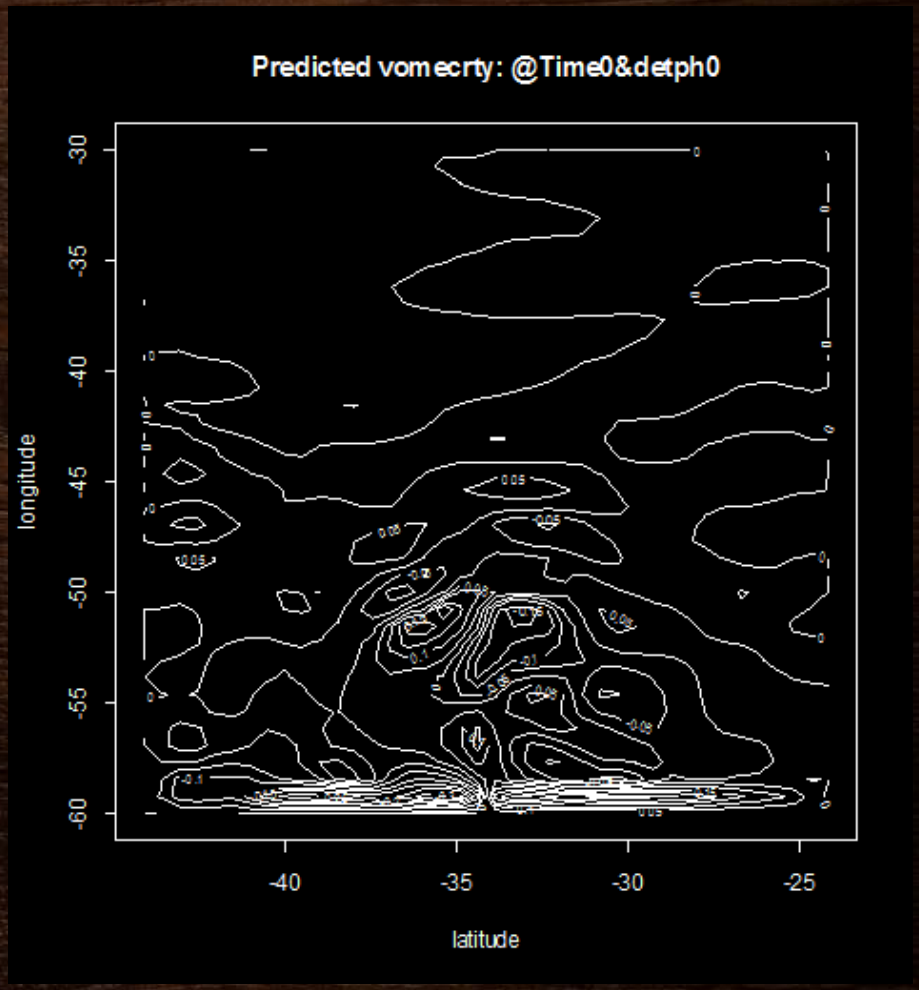
Current prediction

- To predict ocean curren, mathematicians consider a complex model
- Question: random forest can be as efficient as the mathematic model?



Current prediction





Current prediction

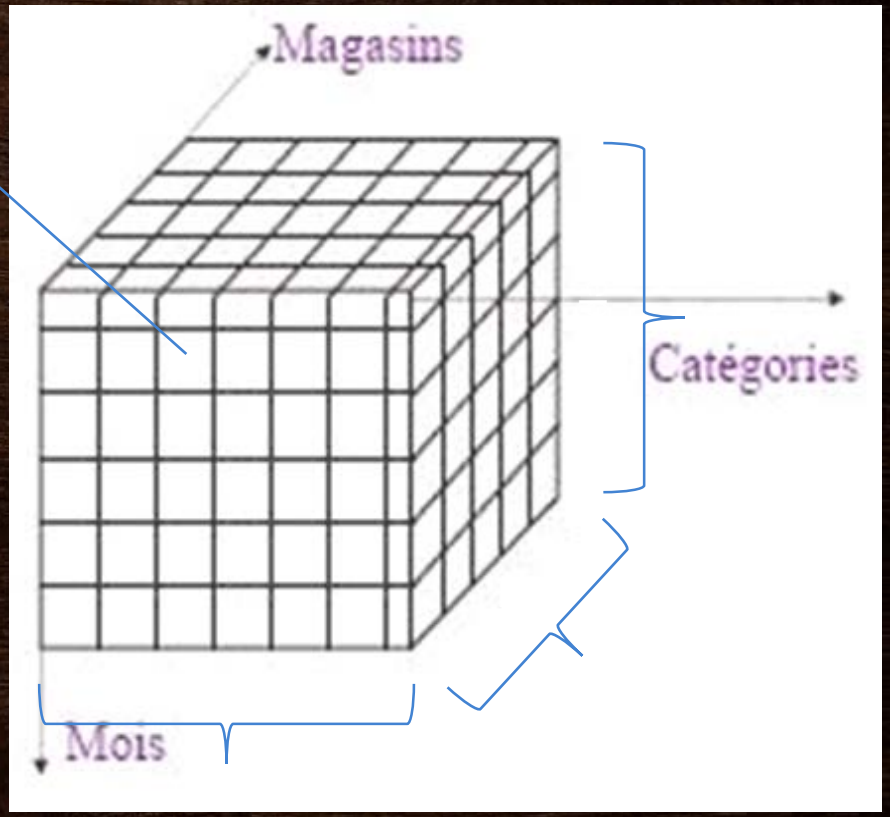
- Data taken into account for learning
  - time_counter: from 0 to 359, represents 360 recording moments, during one year
  - depth: from 0 to 10, represents 11 depth levels of the ocean. in fact, deptht=depthv=depthu=depth
  - nav_lat: dim = 81 × 121, the coordinate of grid along the latitude
  - nav_lon: dim = 81 × 121, the coordinate of grid along the longitude
  - sossheig: dim = 360 × 81 × 121, pressure of the surface of ocean
  - vosaline: dim = 360 × 11 × 81 × 121, the salinity
  - votemper: dim = 360 × 11 × 81 × 121, the temperature
  - sozotaux: dim = 360 × 81 × 121, wind stress along i-axis
  - sometauy: dim = 360 × 81 × 121, win stress along j-axis
  - vozocrtx: dim = 360 × 11 × 81 × 121, zonal current
  - vomecrty: dim = 360 × 11 × 81 × 121, meridional current

Current prediction

- Graphical representation of the data

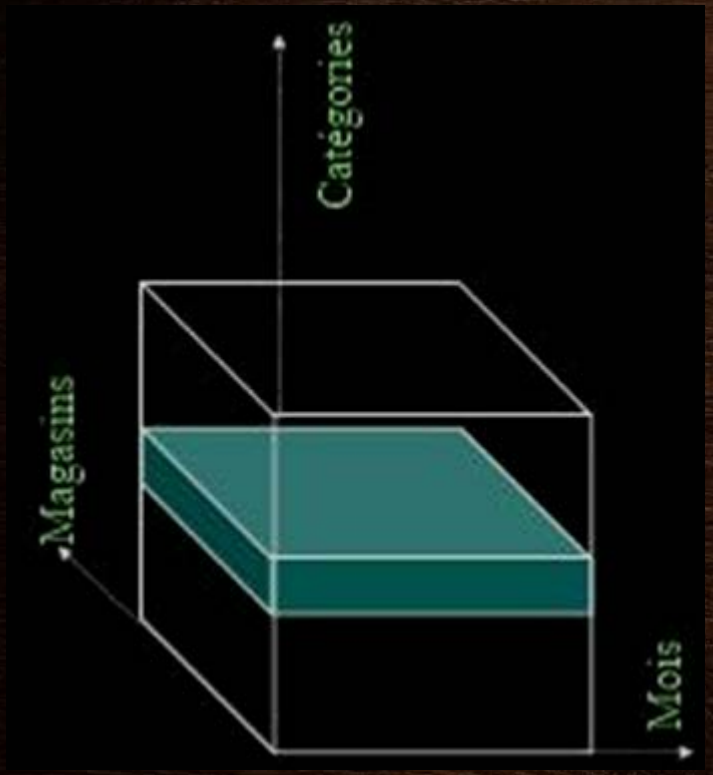
0~359

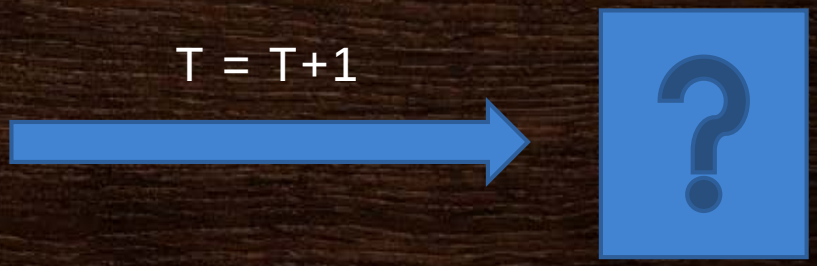
Sossheig
Vosaline
Votemper
Sozotaux
Sometauy
vozocrtx
vomecrty



Current prediction

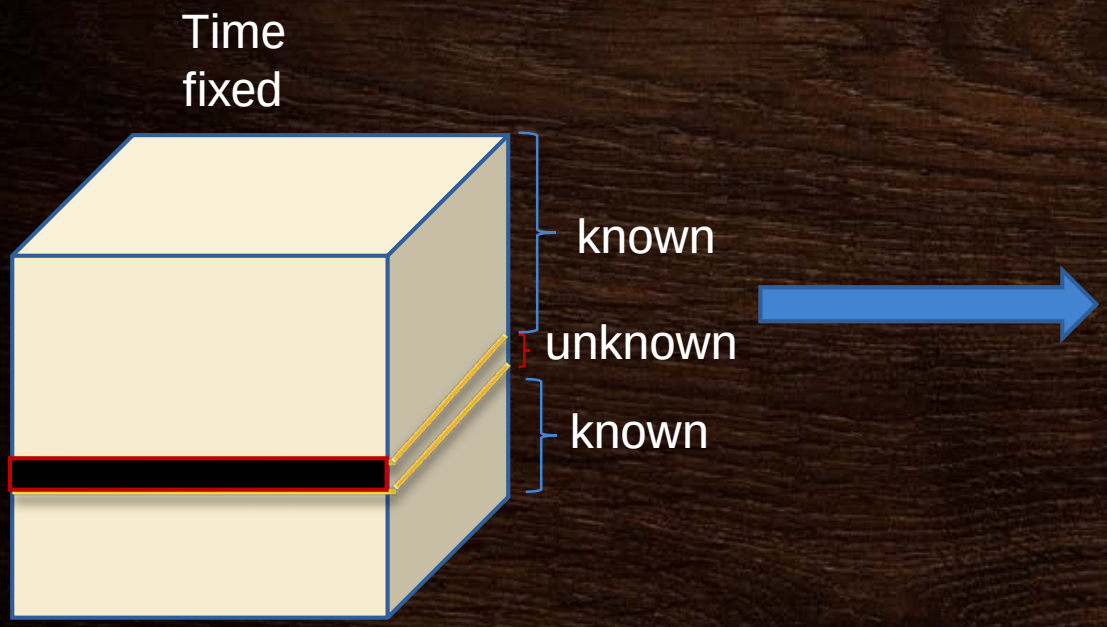
- Different possible actions
  - We learn with all the date and we try to predict a step at time t+1

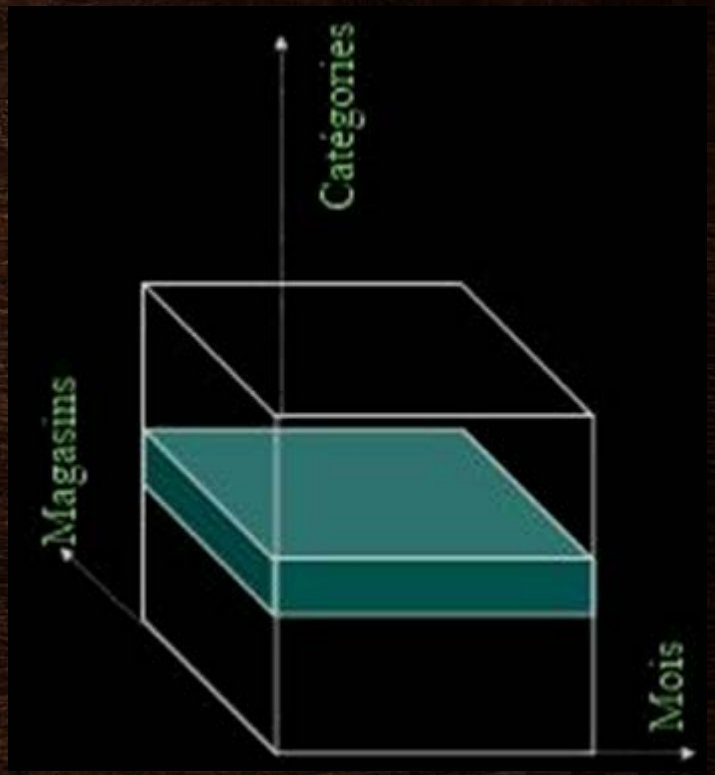




Current prediction

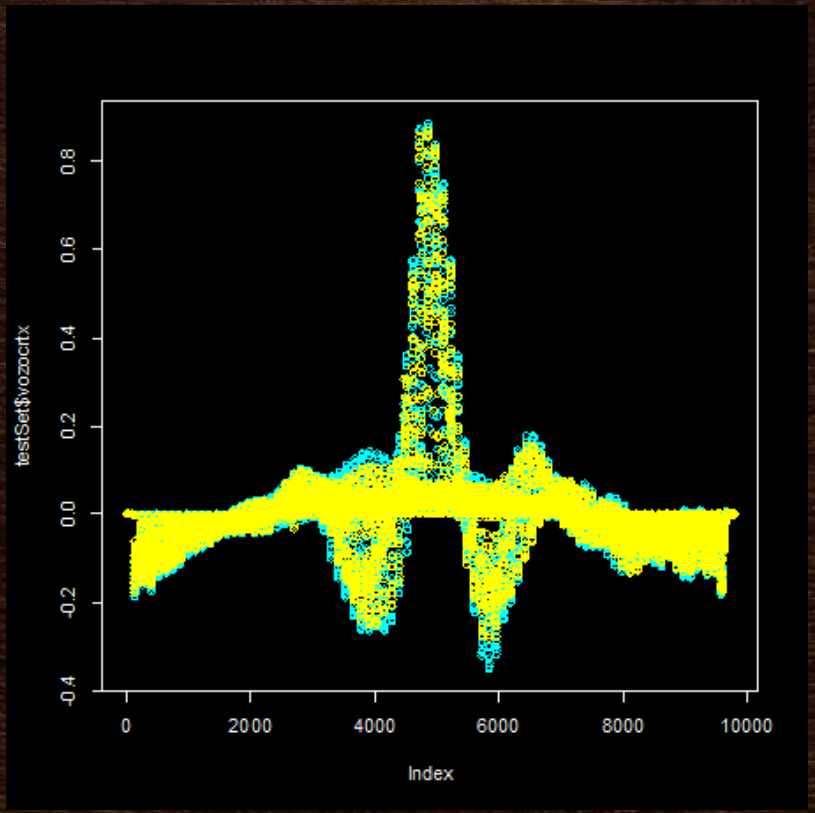
- Different possible actions
  - We learn with 80 steps and we try to predict the 81^{th} one





Current prediction

- Some results
  - We learn with 10 consecutive moments and we predict the 11th one

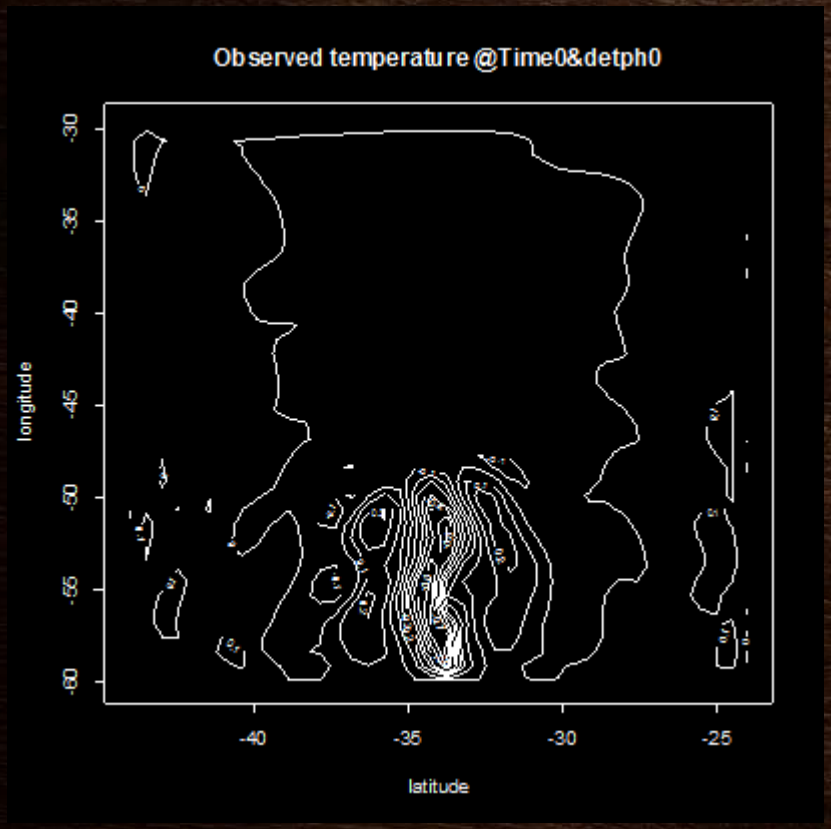


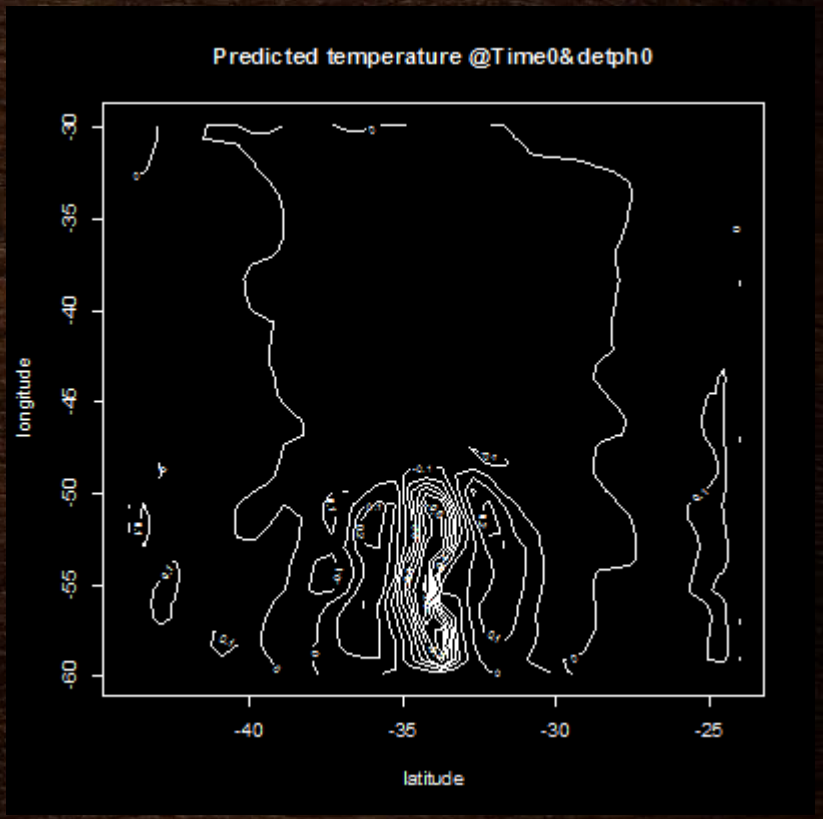
Blues: predictions of testing set

Reds: observations of testing set

Current prediction

- Temperature prediction





Multi-terminal learning system for the discovery of new personalized interests

- SharingBox (https://www.youtube.com/watch?v=lFZ_S2MylnAk)
  - Offers to individuals and companies a technological and IT solution for the animation of events (weddings, birthday evenings, corporate parties, punctual events, events in long courts, ...)
  - Photo kiosks fully customizable: the terminal can be decorated in the colors of the event or the company, the photos taken are customizable in terms of filters applied to the photos (sepia, ...), decorations, Photos
  - Subject:
    - Developing a mobile application for searching for terminals located near the user. The idea of this workis to deploy this functionality by proposing to the user the terminals associated with events that are best suited to him (opening of store, thematic evening, launch of a new product, sports event, etc.).
    - Master project possible…

Any questions?