DATA: to bin or to keep?

- **Current situation:**

  Huge amount of digital data (big data!)
  (social network,etc.)
  Huge amount of features
  Distributed nature
  BUT Powerful computers

  Answer: **we keep!**

- **Aims: understand / extract information from data to knowledge**

  1) Prediction
  2) Decision
  3) Action

MINING

- **Data coming from diverse domains**
  Insurance
  Marketing, big stores
  Medicine, Banking
  Social, etc.
- **Analysis** for
  – Traffic distribution w.r.t. times
  – Bank scores : client faithfulness, classes (bad customers)
  – Recommandation systems (Amazon)
  – Customer behavior prediction
  – Network intrusion detection
  – Website design,etc.

Supermarket example

- **Every day, huge Excel table M (boolean matrix)**
  - **m** columns (1 per item to sell)
  - **n** lines (1 per customer)
  - Init value: 0 (null matrix)
  - Final value:

M_{ij} = 1 iff customer i buys item j

- **M nxm boolean matrix – 364/year**
- **Size estimation:**

10^4 (items) x 8 (hours)x10^3(customers)

More or less 10^8B data/day/store

What is Data Mining?

- **Variables: examples…**
- **Data= variable value or set of variable values**
- **Record/vector: examples…**
- **Tables: examples…**
- **Variables types:**
  - Discrete: finite number of values
  - Continuous: infinite number of values
  - Numeric: in IR or IN or Z
  - Symbolic: color, …
- **How do we get data? Sensors, computers, human record, etc…**

Data mining: extract knowledge from data;-)

From explicit to implicit knowledge!

How Data Mining?

- **Historically:**
  - **Probability/statistics**
  - **Then IT techniques**
- **Proba/stat:**
  - Diagrams
  - Measure of Central tendency (mean, median, mode)
  - Measure of dispersion (range, quartile, standard deviation)
  - Variance, covariance, correlation, etc.
  - Derived methods (PCA,etc.)
- **IT:**
  - Algorithms (neural networks, algo gene.,DT,etc.)
  - Logic (ILP)

Marketing example (review)

- **If we can extract the following "property":**

  if c buys bread **and** butter **then**
  she/he buys marmelade
- **We can do a lot ;-) (explain)**
- **Such a rule: association rule**
- **Automatic extraction ;-)**
- **Very general and powerful concept…**

More formally

- A set of items I - a set of transactions T
- A subset of items X: itemset
- A transaction Id is associated to a list of items X (an itemset) (Excel table)

X --> Y

means

Every transaction including X includes Y

- Some obvious properties
- Need some flexibility "*a large part of*"...

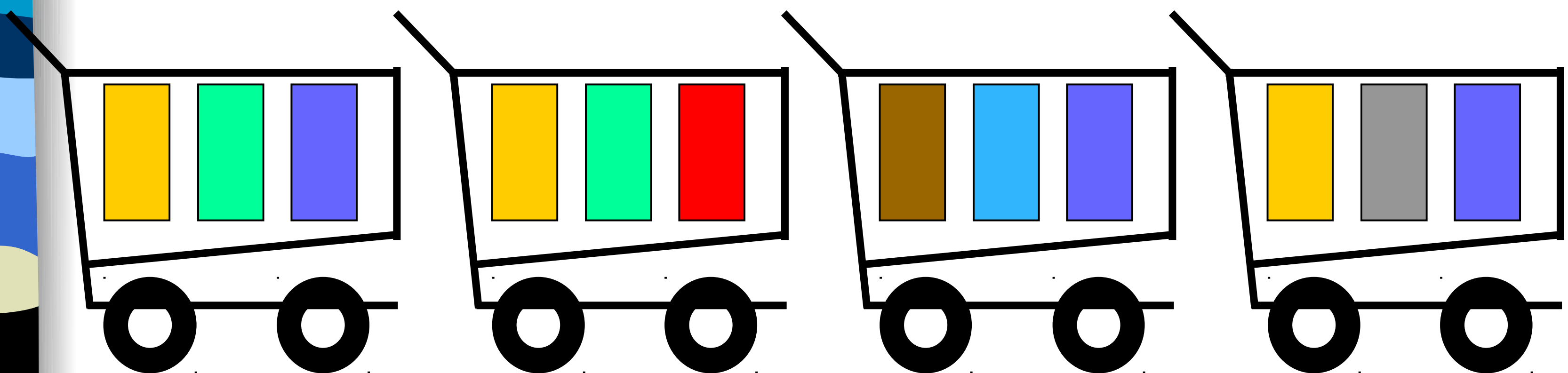
Relevance of a rule

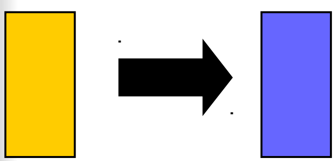
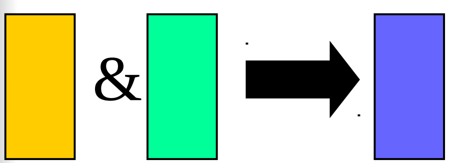
- Notion of support of an itemset X: **Pourcentage de gus qui font X**
  supp(X) = frequency of transactions including X
- Can be considered as a probability P(X)
- Support of a rule X -->Y

  supp(X U Y)
- Need a threshold (explain)
- if supp(X)>threshold: frequent itemset FIS (or r-FIS)
  Seuil pour le support
- Confidence of a rule X -->Y

  supp(X U Y)/supp(X)
- Can be considered as conditional probability P(Y/X)
- Obviously threshold still needed

Example 1



rule	Support	Confidence
	2/4	2/3
	1/4	1/2

LIFT

$$P(X/Y) = \frac{supp(XUY)}{supp(X)}$$

$$lift = \frac{P(Y/X)}{P(Y)} = \frac{supp(XUY)}{supp(X) \times supp(Y)}$$

X -> Y
Y -> Z Pas X -> Z

X -> Y
- s = 0.4
- c = 0.85
- l = 5

Y -> Z
- s' = 0.3
- c' = 0.72
- l' = 10

On ne peut pas déduire :

X -> Z
- s''
- c''
- l''

Computation

- every subset of a r-FIS is a r-FIS !
- every r-FIS of size k is made up with k r-FIS of size k-1 ! (useless)
- every r-FIS of size k is union of 2 r-FIS of size k-1 differing from 1 element (usefull)

algorithm

- Start from r-FIS(1) (support threshold r given)
- Build up r-FIS(2) with 2 elements of r-FIS(1) differing from 1 element
- etc.
- Build up r-FIS(k) with 2 elements of r-FIS(k-1) differing from 1 element

FIS generation (1994)

```
L1 = {frequent 1-ensemble} ;
for (k = 2 ; Lk-1 ≠ ∅ ; k++)
 {
    Ck = apriori-gen(Lk-1); // Generate new candidates
foreach transactions t ∈ DB do
{ // Counting
Ct = { subset(Ck, t) }; // get subsets of t candidates
foreach c ∈ Ct do c.count++;
}
Lk = { c ∈ Ck |c.count >= minsup } ; // Filter candidates
}
Answer = {Lk} ;
```

Rules generation

```
// Input: threshold, Lk associated FIS
// Sortie : ruleset
Rules = ∅ ;
for (k = 2 ; Lk-1 ≠ ∅ ; k++) do
{
  Foreach subset S ≠ ∅ of Lk do
{  Conf (S → Lk-S) = Sup(I)/Sup(S)
   If Conf >= threshold then
 {
  rule = “ S → ( Lk-S ) ” ;
  Rules = Rules ∪ {r} ;
}
}
Answer = Rules ;
```

Complexity and more...

- Very high.. optimization needed;-)
- Other relevant parameters

lift of a rule X --> Y

informally: what is the ratio of confidence I get by observing X instead of observing nothing

formally:

supp(X U Y)/supp(X) *supp(Y)

- Can be interpreted as P(Y/X)/P(Y).. has to be > 1 ;-)
- 1-P(Y): prob not to buy Y (knowing nothing)
- 1-P(Y/X): prob not to buy Y knowing X
- **conviction** = ratio of these 2 numbers

Lessons learned

- ML method: **computationally expensive**
- Find mathematical properties to
  - **guide the search**
  - **optimize complexity**
- **Assoc. rules** --> monotony of r-FIS
- **Decision trees** --> information gain
- **ILP** --> info. gain + accept uncompleteness
- **Bayesian network** --> independance assumption
- **Genetic algorithm**: fitness function

Conclusion

- AR: Very simple and powerful concept
- Every serious database gets the option to compute association rules (Oracle, etc.)
- High complexity --> a lot of optimization works (even recently)
- The most well known algo:
  - Agrawal et al. 1994
- F. Borgelt (apriori.exe implementation)
- More powerful concept needed !