Workshop Machine-Learning & Finance
Université Paris Dauphine
Machine-Learning pour la Finance: Enjeux et Challenges

Stephan Clémençon

Institut Mines Télécom - Télécom ParisTech - Université Paris Saclay

March 20, 2016

Agenda

- **Machine-Learning**: un bref tour d'horizon
- Les défis des applications du Machine-Learning à la Finance:
- Exemples: scoring/rating, du supervisé au non supervisé

Machine Learning - Le contexte

Une accumulation de données **massives** dans de nombreux domaines:

- Biologie/Médecine (génomique, métabolomique, essais cliniques, imagerie, *etc.*
- Grande distribution, marketing (CRM), e-commerce
- Moteurs de recherche internet (contenu multimedia)
- Réseaux sociaux (Facebook, Tweeter, ...)
- Banque/Finance (risque de marché/liquidité, accès au crédit)
- Sécurité (ex: biométrie, vidéosurveillance)
- Administrations (Santé Publique, Douanes)
- Risques opérationnels

"Big Data" - Le contexte

Un **déluge de données** qui rend inopérant:

- les outils basiques de
  - stockage de données
  - gestion de base de données (MySQL)
- le prétraitement reposant sur l'expertise humaine
  - indexation, analyse sémantique
  - modélisation
  - intelligence décisionnelle

"Big Data" - Le contexte

Une multitude de briques technologiques et de services disponibles pour:

- La parallélisation massive (Velocity)
- Le calcul distribué (Volume)
- La gestion de données sans schéma prédéfini (Variety)

parmi lesquels:

- Le modèle de programmation MapReduce: calculs parallélisés/distribuées
- Framework Hadoop
- NoSQL: SGBD Cassandra, MongoDB, bases de données orientées graphe, moteur de recherche Elasticsearch, *etc.*
- Clouds: infrastructures, plate-formes, logiciels *as a Service*

promus par Google, Amazon, Facebook, *etc.*

"Big Data" - Les opportunités

Des avancées spectaculaires pour

- la **collecte** et le **stockage** (distribué) des données
- la **recherche** automatique d'objets, de contenu
- le **partage** de données peu structurées
- L'**analyse** (prédictive) et la **visualisation** des données

Les Données : un moteur pour la technologie, la science, l'économie

- Moteurs de recherche, moteurs de recommandation
- Maintenance prédictive
- Marketing viral à travers les réseaux sociaux
- Détection des fraudes
- Médecine individualisée
- Publicité en ligne (retargeting)

"Big Data" - Les opportunités

Ubiquité

De nombreux secteurs d'activité sont concernés:

- (e-) Commerce
- CRM
- Santé
- Défense, renseignement (*e.g.* cybersécurité, biométrie)
- Banque/Finance
- Transports "intelligents"
- *etc.*

Analyse des Masses de Données - Recherche

Afin d'exploiter les données massives (prédiction, interpétation), développer des technologies mathématiques permettant de résoudre les problèmes computationnels liés:

- aux contraintes du quasi-temps réel (Vélocité)
  $\rightarrow$ apprentissage automatique séquentiel ("on-line") $\neq$ batch, par renforcement
- au caractère distribué/massif des données/ressources (Volume)
  $\rightarrow$ apprentissage automatique distribué/randomisé
- à la complexité des données (Variété)
  $\rightarrow$ représentations parcimonieuses (graphes, texte, séries temporelles)

Analyse des Masses de Données - Recherche

Domaines

- Probabilité, Statistique
- **Machine-Learning**
- Optimisation
- Traitement du signal et de l'image
- Analyse Harmonique Computationnelle
- Analyse sémantique
- *etc.*

Goals of Statistical Learning

- Statistical issues cast as M-estimation problems:
  - Classification
  - Regression
  - Density level set estimation
  - Compression, sparse representation
  - ... and their variants
- Minimal assumptions on the distribution
- Build realistic M-estimators for special criteria
- Questions
  - Theory: optimal elements, consistency, non-asymptotic excess risk bounds, fast rates of convergence, oracle inequalities
  - Practice: numerical optimization, convexification, randomization, relaxation, constraints (distributed architectures, real-time, memory, *etc.*)

Main Example: Classification (Pattern Recognition)

- (X, Y) random pair with unknown distribution P
- $X \in \mathcal{X}$ observation vector
- $Y \in \{-1, +1\}$ binary label/class
- *A posteriori* probability $\sim$ regression function

$$\forall x \in \mathcal{X}, \quad \eta(x) = \{Y = 1 \mid X = x\}$$

- $g : \mathcal{X} \to \{-1, +1\}$ classifier
- Performance measure = classification error

$$L(g) = g(X) \neq Y \quad \to \min_g$$

- Solution: Bayes rule

$$\forall x \in \mathcal{X}, \quad g^*(x) = 2\{\eta(x) > 1/2\} - 1$$

- Bayes error $L^* = L(g^*)$

Main Paradigm - Empirical Risk Minimization

- Sample $(X_1, Y_1), \ldots, (X_n, Y_n)$ with i.i.d. copies of (X, Y),class of classifiers
- Empirical Risk Minimization principle

$$\widehat{g}_n = \operatorname*{Argmin}_{g \in} L_n(g) := \frac{1}{n} \sum_{i=1}^{n} \{g(X_i) \neq Y_i\}$$

- Best classifier in the class

$$\bar{g} = \operatorname*{Argmin}_{g \in} L(g)$$

- Concentration inequality

  With probability $1 - \delta$:

$$\sup_{g \in} \mid L_n(g) - L(g) \mid \leq C \sqrt{\frac{V}{n}} + \sqrt{\frac{2 \log(1/\delta)}{n}}$$

Machine-Learning - Achievements

- Numerous applications:
  - Supervised anomaly detection
  - Handwritten digit recognition
  - Face recognition
  - Medical diagnosis
  - Credit-risk screening
  - CRM
  - Speech recognition
  - Monitoring of complex systems
  - *etc.*
- Many "off-the-shelf" methods
  - Neural Networks
  - Support Vector Machines
  - Boosting
  - Vector quantization
  - *etc.*
- Many softwares available, ex: http://scikit-learn.org

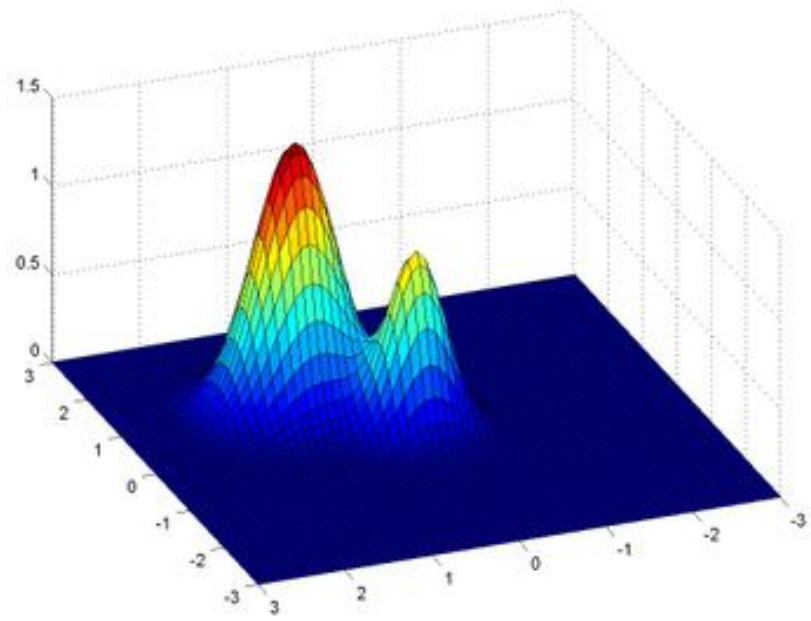
Machine-Learning - Challenges

- Ongoing intense research activity, motivated by
  - Need for increasing performance
  - Evolution of computing environments (data centers, clouds, HDFS)
  - New applications/problems: recommending systems, search engines, medical imagery, yield management *etc.*
  - The Big Data era

- Mathematical/computational challenges
  - **Volume** (data deluge): ubiquity of sensors, high dimension, distributed storage/processing systems
  - **Variety** (of data structures): text, graphs, images, signals
  - **Velocity** (real-time): on-line prediction, evolutionary environment, reinforcement learning strategies (exploration vs exploitation)

Statistical Learning - Milestones

- The 30's - Fisher's (parametric/Gaussian) statistics
  - Linear Discriminant Analysis
  - Linear (logistic) regression
  - PCA, ...



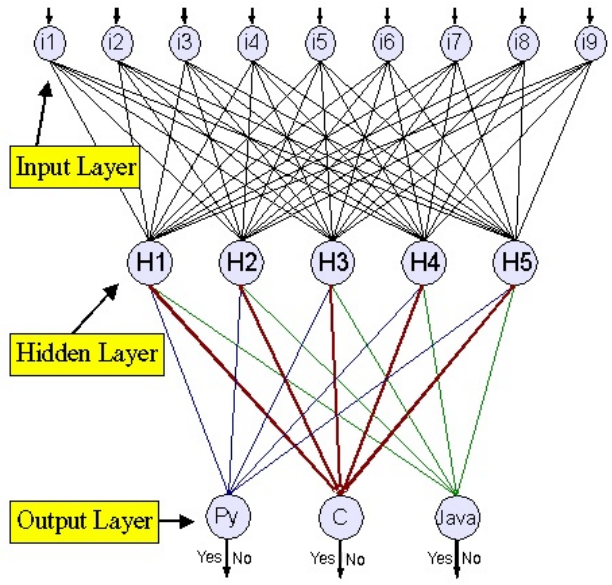
Statistical Learning - Milestones

- The 60's and 70's - F. Rosenblatt's perceptron & VC theory
  - First "machine-learning" algorithm (linear binary classification)
  - Inspired by congnitive sciences
  - Convexification, one-pass/on-line (stochastic gradient descent)
  - Relaxation, large margin linear classifiers, structural ERM

A. Chervonenkis & V. Vapnik

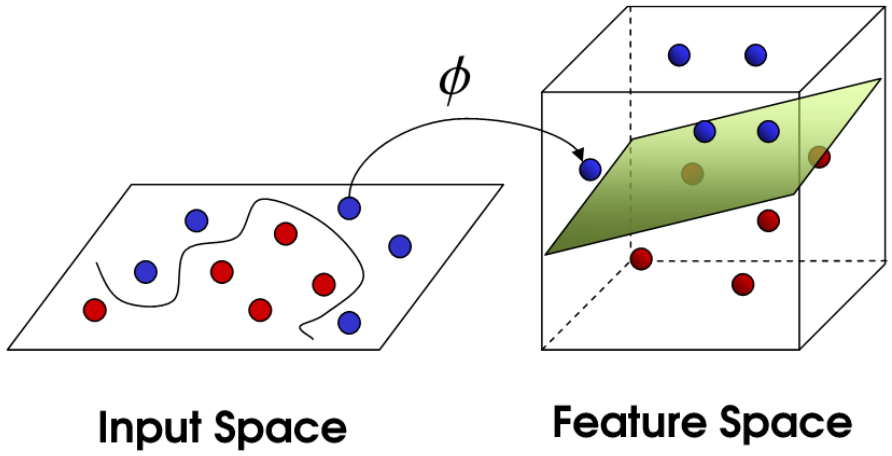
Statistical Learning - Milestones

- The 80's - Neural Networks & Decision Trees
  - Artificial Intelligence "A theory of learnability" Valiant '84
  - The Backpropagation algorithm
  - The CART algorithm ('84)



Statistical Learning - Milestones

- From the 90's - Kernels & Boosting
  - Kernel trick: SVM, nonlinear PCA, ...

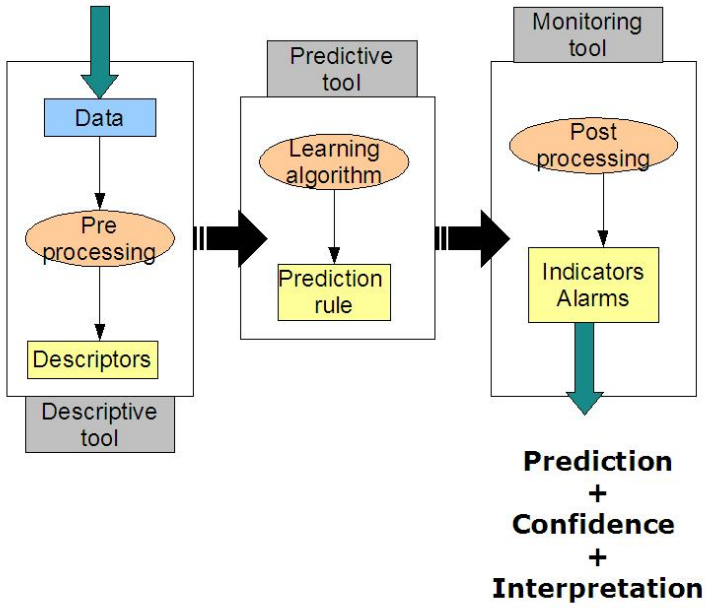


  - AdaBoost ('95)
  - Lasso, compressed sensing
  - A comprehensive theory beyond VC concepts
  - Rebirth of Q-learning

Applications

Supervised Learning - Pattern Recognition/Regression

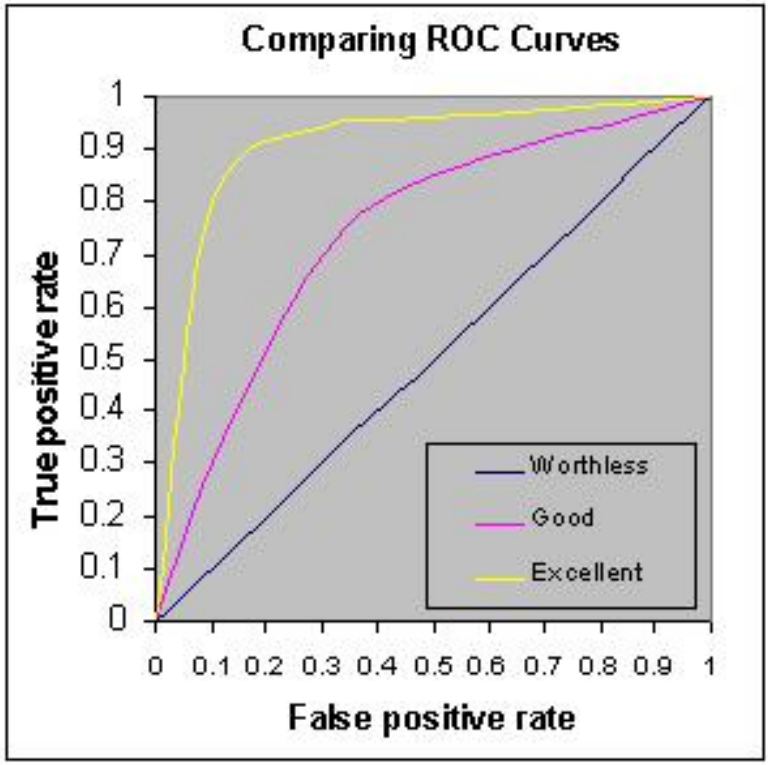
- Data with labels, *e.g.* $(X_i, Y_i) \in \mathbb{R}^d \times \{-1, +1\}$, $i = 1, \ldots, n$. Learn to **predict** Y **based on** X
- Example: in **Quality Control**, X features of the product and/or production factors, $Y = +1$ if "defect" and $Y = -1$ otherwise. Build a decision rule C minimizing $L(C) = \mathbb{P}\{Y \neq C(X)\}$



Applications

Supervised Learning - Scoring

- Data with labels, *e.g.* $(X_i, Y_i) \in \mathbb{R}^d \times \{-1, +1\}$, $i = 1, \ldots, n$. Learn to **rank all possible observations X in the same order as that induced by** $\mathbb{P}\{Y = +1 \mid X\}$ through a **scoring function** $s(X)$



Applications

Supervised Learning - Image Recognition

- Objects are assigned to data (pixels), *e.g.* biometrics
- Goal: learn to **assign objects to new data**

Empirical Risk Minimization and Stochastic Approximation

- Most learning problems consists of minimizing a functional

$$L(f) = \mathbb{E}[\psi(Z, f)]$$

where Z is the observation, f a decision rule candidate

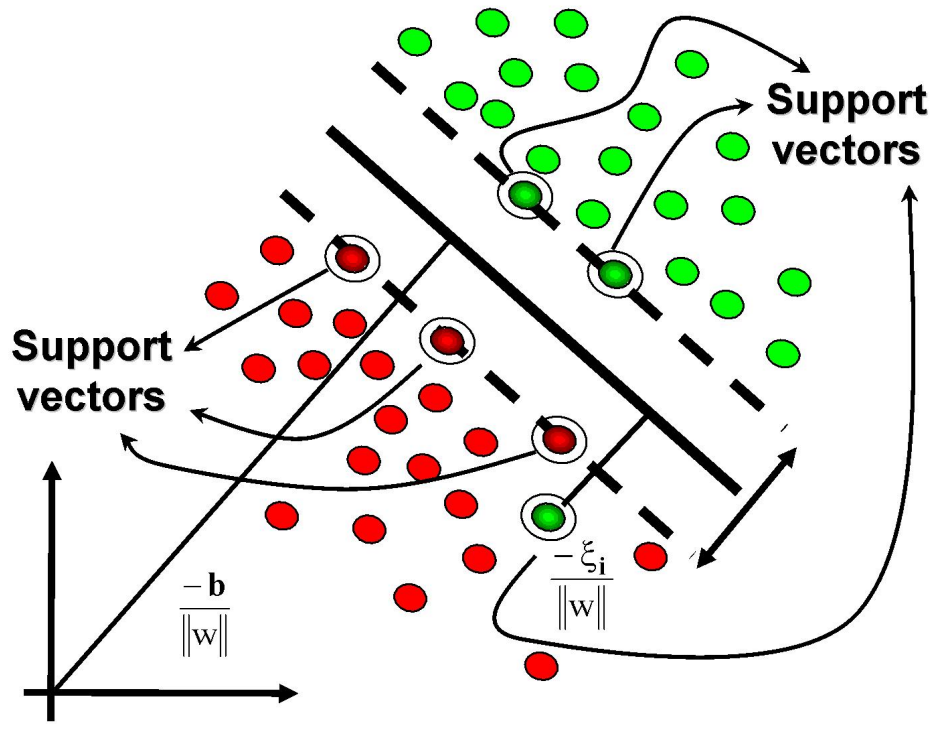
- In general, a stochastic approximation inductive method must be implemented

$$f_{t+1} = f_t - \rho_t \widehat{\nabla_f L}(f_t),$$

where $\widehat{\nabla_f L}$ is a statistical estimate of L's gradient based on training data $Z_1, \ldots, Z_n$

Empirical Risk Minimization and Stochastic Approximation

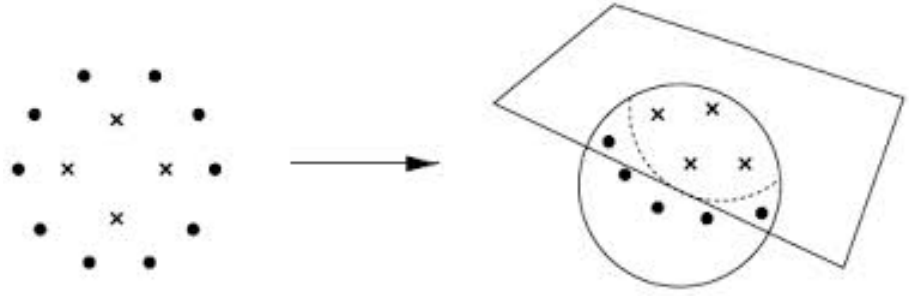
- Popular algorithms are based on these principles
- Examples: Logit, Neural Networks, linear SVM, *etc.*
- Computational advantages but too rigid (underfitting)



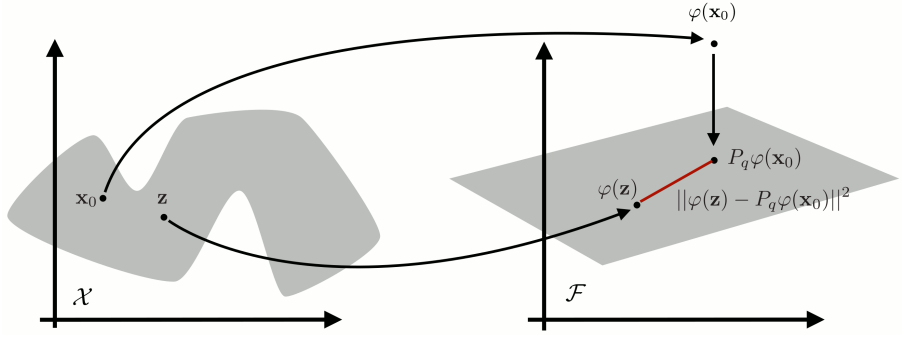
Kernel Trick

- Apply a simple algorithm but... in a transformed space

$$K(x, x') = \langle \phi(x), \phi(x') \rangle$$



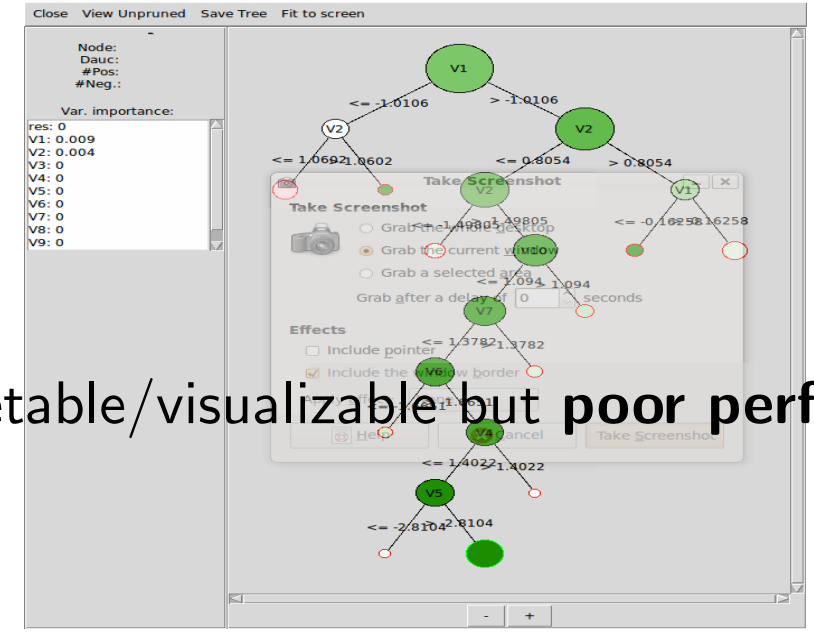
- Examples: Nonlinear SVM, Kernel PCA, SVR



- Kernels for images, text data, biological sequences, *etc.*

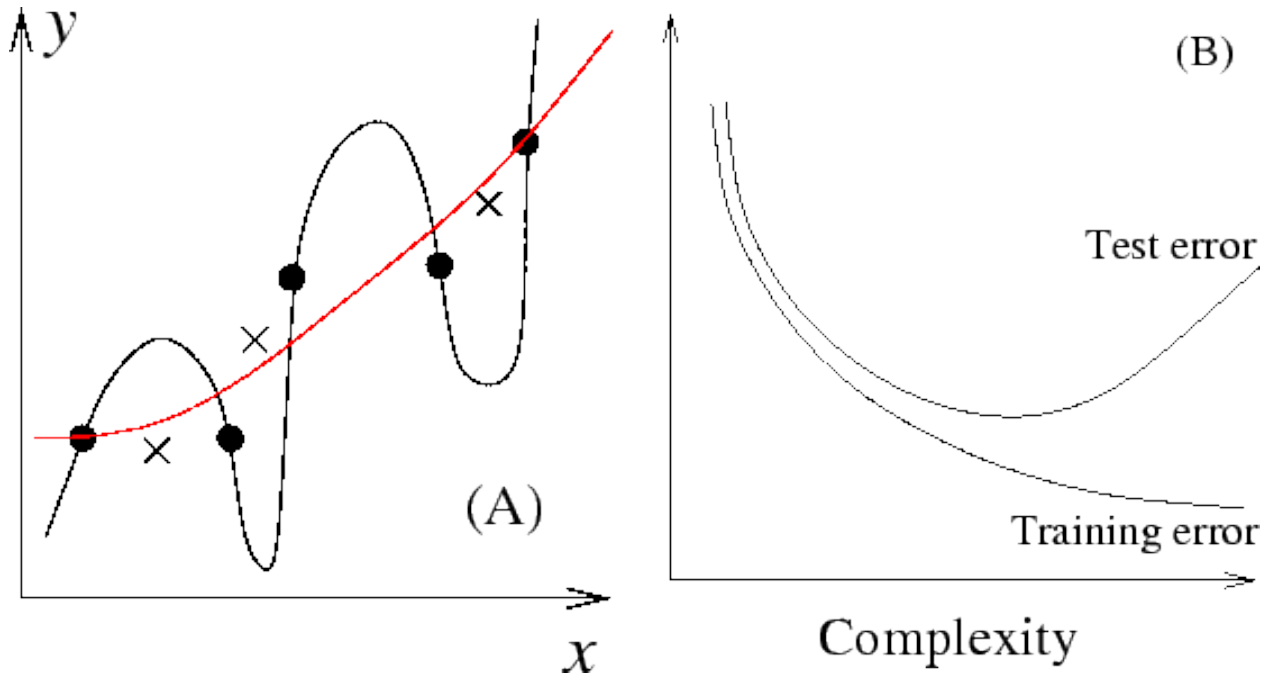
Greedy Algorithms

- **Recursive** methods exploring exhaustively a **structured space** at each step
- Examples: CART, projection pursuit, matching pursuit, *etc.*



- Highly interpretable/visualizable but **poor performance**

No Free Lunch



Ensemble learning

Heuristic: combine predictions output by weak decision rules
Amit & Geman ('97) for image recognition

- **Example** committee-based binary classification:
  predict $Y \in \{-1, +1\}$ based on X

$$C_{agg}(X) = sgn\left(\sum_{m=1}^{M} \omega_m C_m(X)\right),$$

  where ω_m controls the impact of the vote of weak rule C_m

- The **B**ootstrap **Agg**regat**ing** method - Breiman ('96)

  The C_m's re learnt from bootstrap versions of the training data and $\omega_m \equiv 1$
  $\Rightarrow$ Bagging reduces instability of prediction rules

Ensemble learning

- The Adaptive Boosting algorithm for binary classification Freund & Shapire ('95) - Slow learning
- AdaBoost can be interpreted as a **forward additive stagewise modelling** strategy to minimize a convexified version of the risk

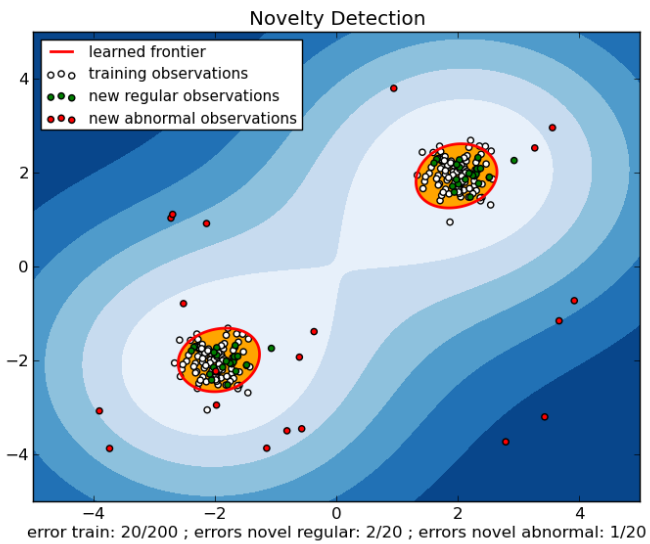
$$\mathbb{E}[\exp(-Y \sum_{m=1}^{M} \alpha_m C_m(X))]$$

- A serious competitor: Random Forest, Breiman ('01) Bagging applied to randomized decision trees
- Boosting methods and Random Forests outperform older methods in most cases

Applications

Unsupervised Learning - Anomaly/Novelty Detection

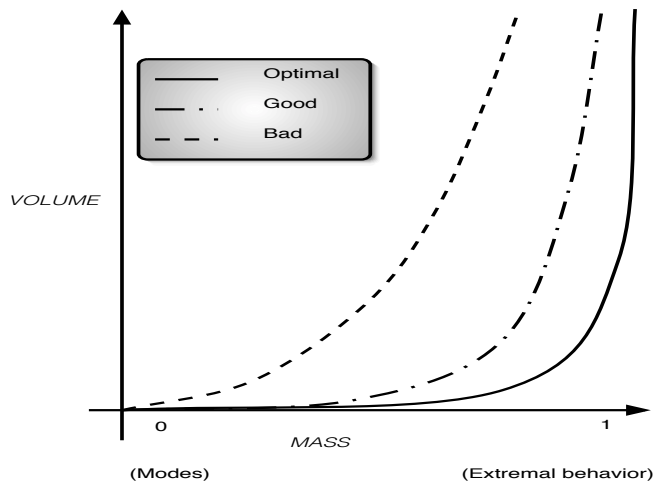
- Data with **no labels**, *e.g.* $X_i \in \mathbb{R}^d$, $i = 1, \ldots, n$
- Example: **monitoring of complex systems**, *e.g.* aircraft systems, fraud detection, predictive maintenance, cybersecurity
- Detect abnormal observations - **Rarity replaces labeling**
- 1-class SVM: $\widehat{G}_\alpha = \{x \in \mathcal{X} : \sum_{i=1}^n \alpha_i K(x, X_i) \geq t_\mu\}$



Applications

Unsupervised Learning - Anomaly/Novelty Ranking

- Data with **no labels**, *e.g.* $X_i \in \mathbb{R}^d$, $i = 1, \ldots, n$
- **Rank** data by degree of novelty/abnormality
- **Distributed Fleet Monitoring**: check the 5 % the most abnormal, then the next 5 %, *etc.*



Feature Selection

A quick algorithm has been proposed by Efron *et al* (2002) to compute

$$\widehat{\beta}_n(\lambda) = \underset{\phi}{\text{Argmin}} \left[\frac{1}{n} \sum_{i=1}^{n} (y_i - \mathbf{x}_i^T \phi)^2 + \lambda \sum_{i=1}^{p} |\phi_i| \right],$$

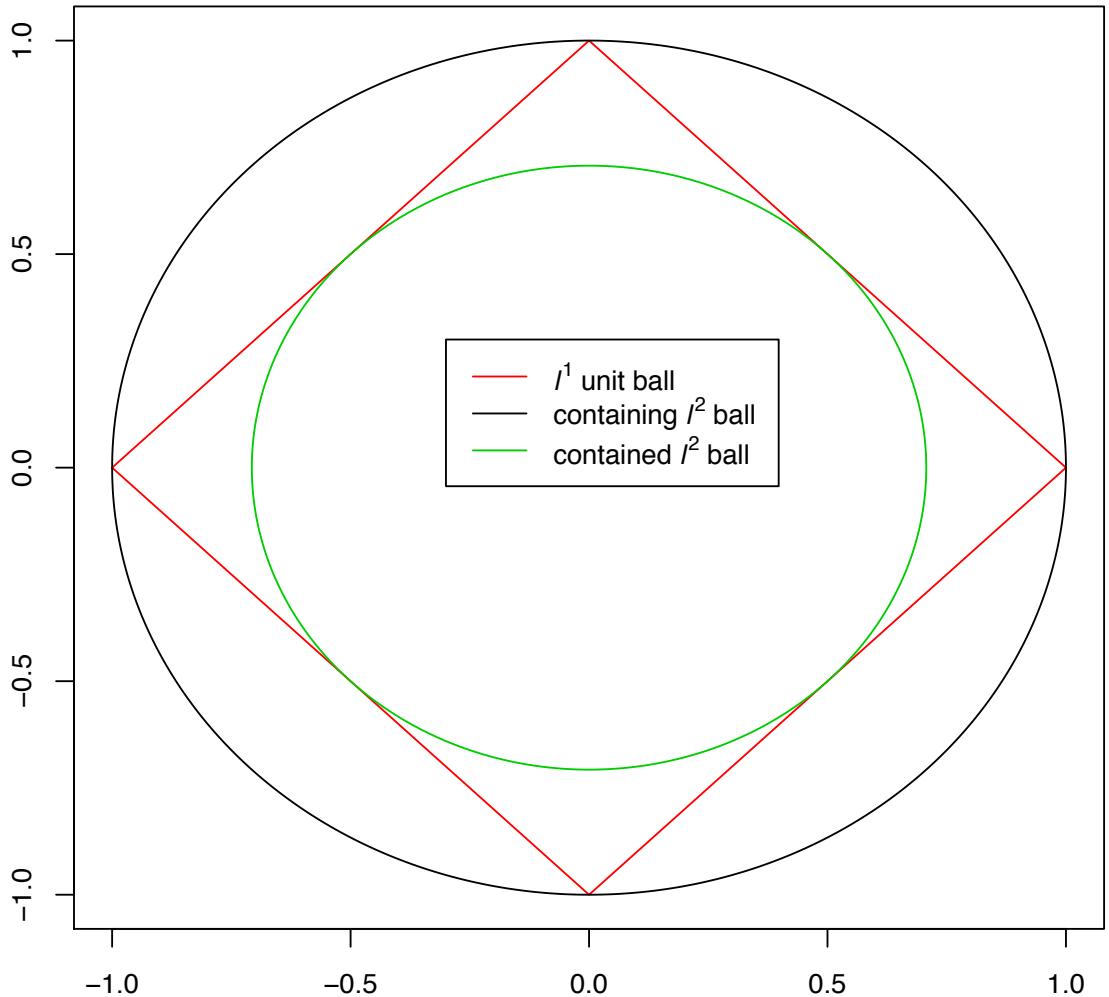
for all $\lambda > 0$.
As λ decreases one obtains more and more active (i.e. non zero) coefficients. The regularization path

$$\lambda \mapsto \widehat{\beta}_n(\lambda)$$

thus defines as $\lambda \downarrow 0$ a sequence of models with increasing dimension.

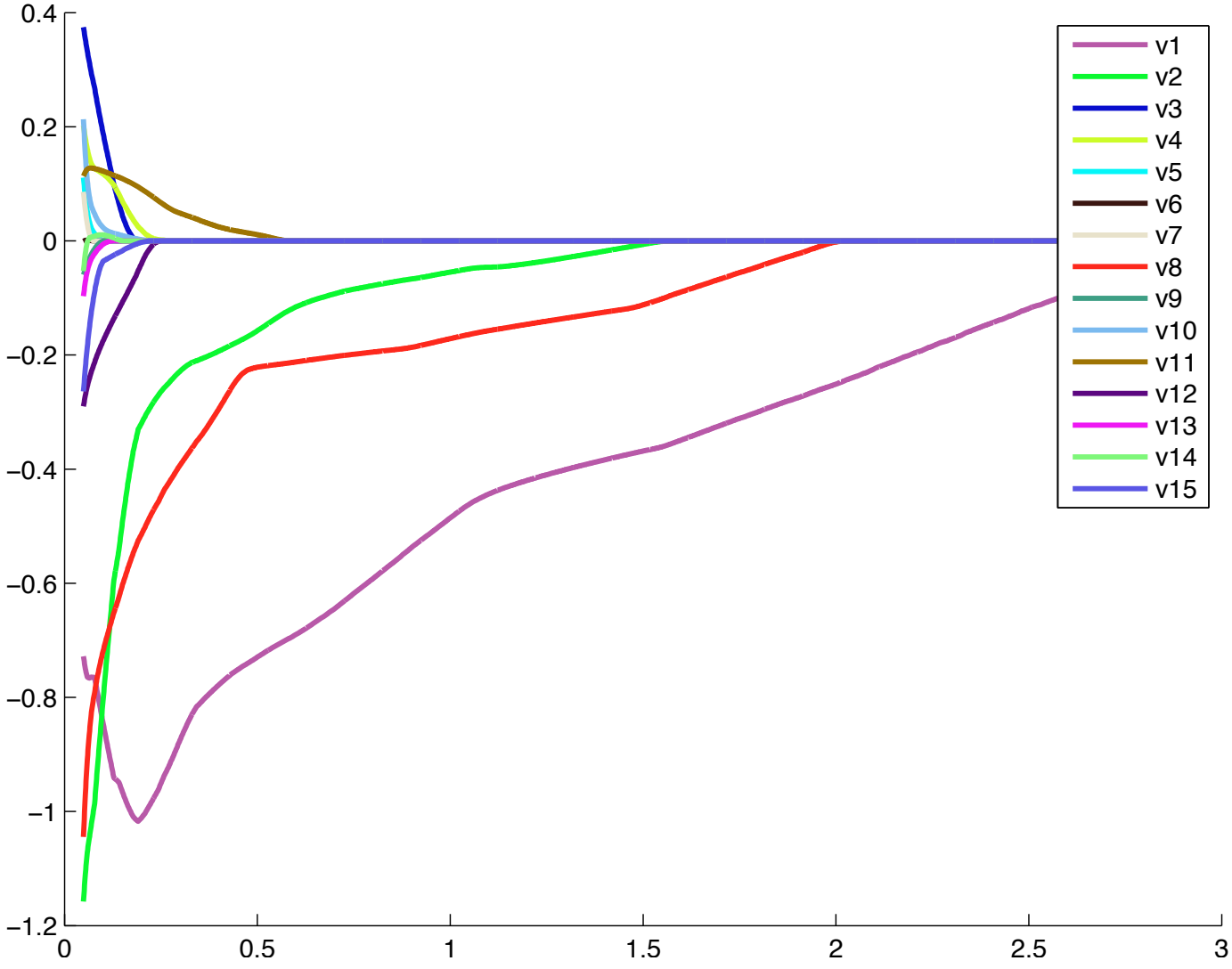
Variable selection

Under ℓ^1 constraints, the points that are the ℓ^2-furthest away from the origin are on the axes (zero coefficient):



Industrial example (Renault Technocentre)

Regularization path: $\boldsymbol{\beta}_{1,2}, \ldots, \boldsymbol{\beta}_{1,p}$ VS λ



Lasso - L_1 penalty

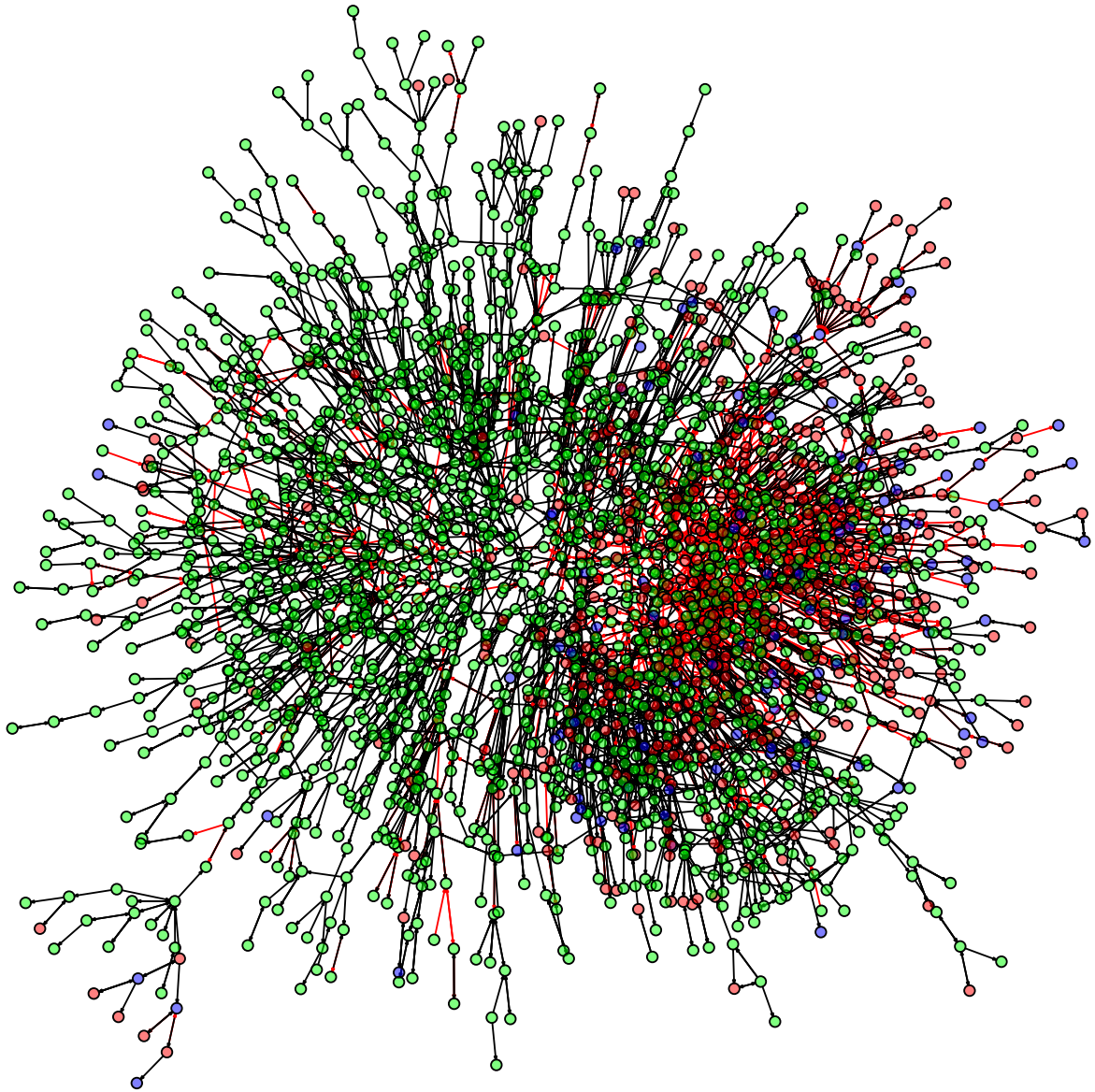
- Many variants: group Lasso, lasso and elastic net
- L_1 penalty ensures sparsity
- Compressed sensing: Candès & Tao ('04), Donoho ('04)
- Numerous applications, *e.g.* matrix completion, recommender systems

Spectral clustering - Ng *et al.* ('01)

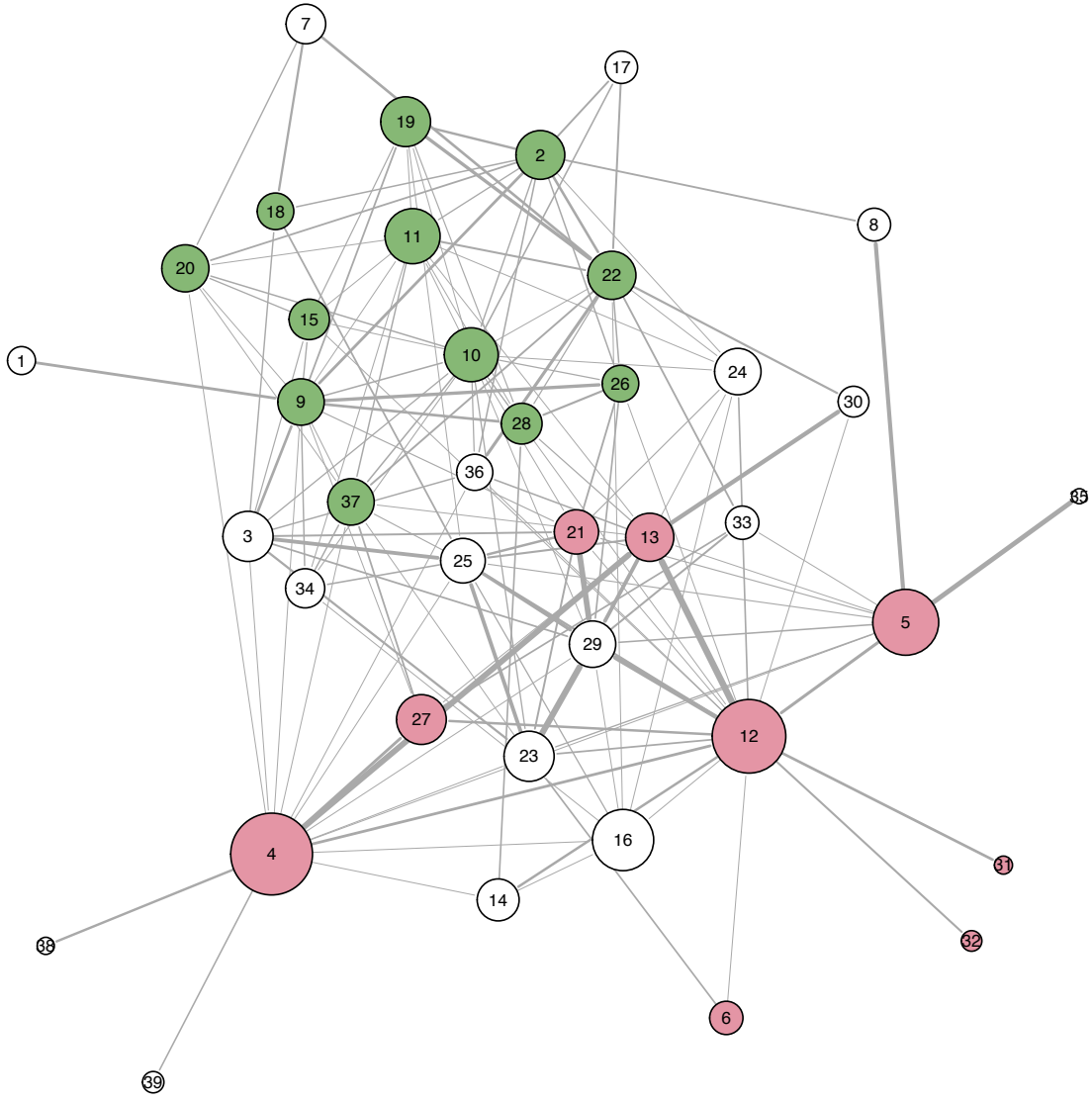
- Partition the vertices of a graph, clustering
- Graph Laplacian $L = D - W$, D is the degree matrix and W is the adjacency/weight matrix
- Spectral Clustering using the normalised version $\tilde{L} = D^{-1/2} L D^{-1/2}$

  (i) Find k-smallest eigenvectors $V_k = (v_1, \ldots, v_k)$ of $\tilde{L}$
  (ii) Normalise V_k's rows: $V_k \leftarrow diag(V_k V_k^t)^{-1/2} V_k$
  (iii) Cluster rows of V_k with the k-means algorithm

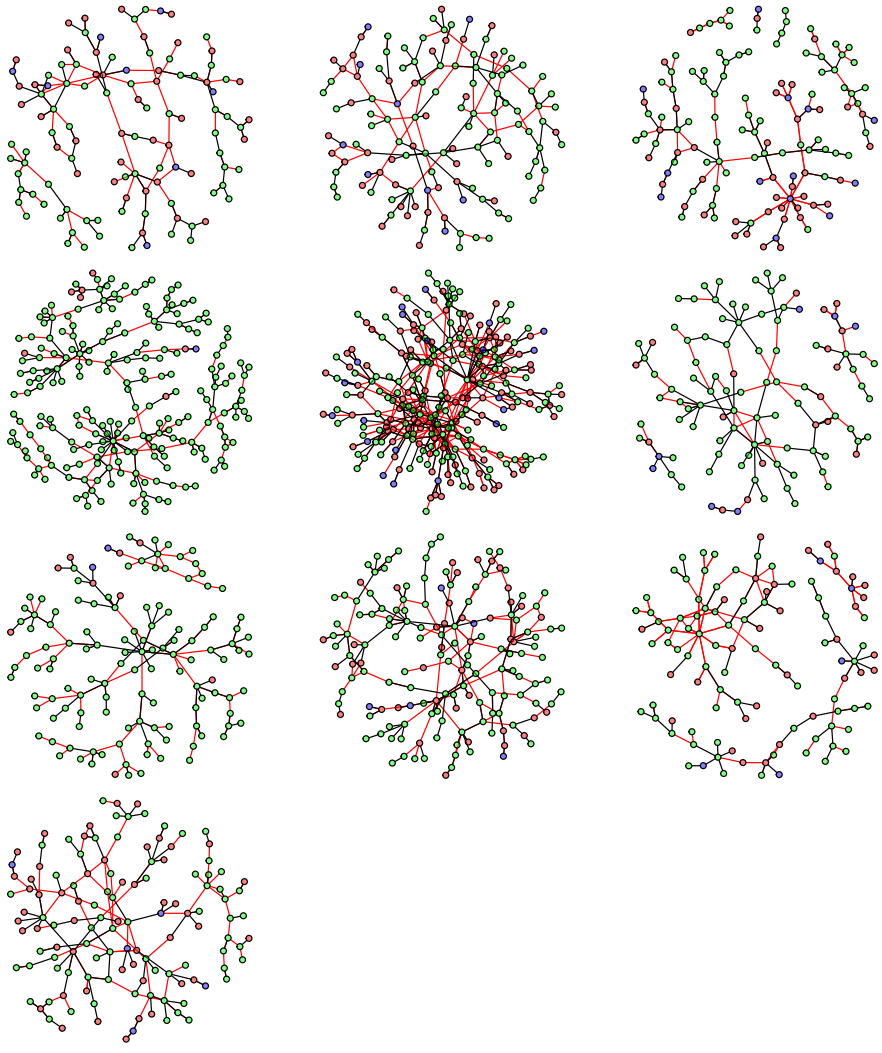
Spectral clustering - Ng *et al.* ('01)



Spectral clustering - Ng *et al.* ('01)



Community Detection



Spectral clustering - Ng *et al.* ('01)

- Partition the vertices of a graph, clustering
- Graph Laplacian $L = D - W$, D is the degree matrix and W is the adjacency/weight matrix
- Spectral Clustering using the normalised version $\tilde{L} = D^{-1/2} L D^{-1/2}$

  (i) Find k-smallest eigenvectors $V_k = (v_1, \ldots, v_k)$ of $\tilde{L}$
  (ii) Normalise V_k's rows: $V_k \leftarrow diag(V_k V_k^t)^{-1/2} V_k$
  (iii) Cluster rows of V_k with the k-means algorithm

Machine-Learning Applied to Finance: Main Challenges

Financial data

- are massive
- are of explosive dimensionality
- exhibit high autocorrelation (paths $\neq$ vectors), are not i.i.d.
- are very heterogeneous (*e.g.* return series, economic series, rates, ratings, tweets)
- must be analyzed in quasi-real time

Machine-Learning Applied to Finance: Main Challenges

Many issues

- Financial risks may not be well described by sample means *e.g.* Clémençon, Goix & Sabourin (2015, 16)
- Prediction problems in Finance should be adressed in a multitask setup, *e.g.* Argyriou, Clémençon & Zhang (2014)
- Certain prediction problems should be formulated in a reinforcement learning setup
- How to represent efficiently financial data?

Scaling-up Machine-Learning Algorithms

- **"Smart Randomization/Sampling"**
- **Massive Parallelization**: break a large optimization problem into smaller problems
  e.g. Cascade SVM, parallel large-scale feature selection, parallel clustering
- **Distributed Optimization**
- **Many frameworks** are available:
  MapReduce (+In Memory=PLANET, IBM PML, Mahout), DryadLINQ, MADlib, Storm, *etc.*

How to apply the ERM paradigm to Massive Data?

- Suppose that n is too large to evaluate the empirical risk $L_n(g)$
- Common sense: run your preferred learning algorithm using a subsample of "reasonable" size $B << n$, *e.g.* by drawing with replacement in the original training data set...

How to apply the ERM paradigm to Massive Data?

- Suppose that n is too large to evaluate the empirical risk $L_n(g)$
- Common sense: run your preferred learning algorithm using a subsample of "reasonable" size $B << n$, *e.g.* by drawing with replacement in the original training data set...
- ... but of course, statistical performance is **downgraded**

$$1/\sqrt{n} \; << \; 1/\sqrt{B}$$

"Smart sampling"

- Use side information and implement your **mini-batch SGD** with a Horvitz-Thompson estimate of the local gradient Bertail, Chautru & Clémençon (2014), Bertail, Chautru, Clémençon & Papa (2016)
- In various situations, the performance criterion is not a basic sample mean statistic any more but a U-statistic
- **Examples:**
  - Clustering: within cluster point scatter related to a partition $\mathcal{P}$

$$\frac{2}{n(n-1)} \sum_{i<j} D(X_i, X_j) \sum_{C \in \mathcal{P}} \mathbb{I}\{(X_i, X_j) \in C^2\}$$

  - Graph inference (link prediction)
  - Ranking
  - ...

Example: Ranking

- **Data with ordinal label:**
  $(X_1, Y_1), \ldots, (X_n, Y_n) \in \left(X \times \{1, \ldots, K\}\right)^{\otimes n}$

Example: Ranking

- **Data with ordinal label:**
  $(X_1, Y_1), \ldots, (X_n, Y_n) \in \left(X \times \{1, \ldots, K\}\right)^{\otimes n}$
- **Want to:** rank $X_1, \ldots, X_n$ through a scoring function
  $s : X \to$ s.t.

  $s(X)$ and Y tend to increase/decrease together with high probability

Example: Ranking

- **Data with ordinal label:**
  $(X_1, Y_1), \ldots, (X_n, Y_n) \in \left(\mathcal{X} \times \{1, \ldots, K\}\right)^{\otimes n}$
- **Want to:** rank $X_1, \ldots, X_n$ through a scoring function $s : \mathcal{X} \to$ s.t.

  $s(X)$ and Y tend to increase/decrease together with high probability

- **Quantitative formulation:** maximize the criterion

$$L(s) = \mathbb{P}\{s(X^{(1)}) < \ldots < s(X^{(k)}) \mid Y^{(1)} = 1, \ldots, Y^{(K)} = K\}$$

Example: Ranking

- **Data with ordinal label:**
  $(X_1, Y_1), \ldots, (X_n, Y_n) \in \left(\mathcal{X} \times \{1, \ldots, K\}\right)^{\otimes n}$
- **Want to:** rank $X_1, \ldots, X_n$ through a scoring function $s : \mathcal{X} \to$ s.t.

  $s(X)$ and Y tend to increase/decrease together with high probability

- **Quantitative formulation:** maximize the criterion

$$L(s) = \mathbb{P}\{s(X^{(1)}) < \ldots < s(X^{(k)}) \mid Y^{(1)} = 1, \ldots, Y^{(K)} = K\}$$

- **Observations:** n_k i.i.d. copies of X given $Y = k$, $X_1^{(k)}, \ldots, X_{n_k}^{(k)}$

$$n = n_1 + \ldots + n_K$$

Example: Ranking

- A natural empirical counterpart of $L(s)$ is

$$\widehat{L}_{\mathbf{n}}(s) = \frac{\sum_{i_1=1}^{n_1} \cdots \sum_{i_K=1}^{n_K} \mathbb{I}\left\{ s(X_{i_1}^{(1)}) < \ldots < s(X_{i_K}^{(K)}) \right\}}{n_1 \times \cdots \times n_K},$$

Example: Ranking

- A natural empirical counterpart of $L(s)$ is

$$\widehat{L}_{\mathbf{n}}(s) = \frac{\sum_{i_1=1}^{n_1} \cdots \sum_{i_K=1}^{n_K} \mathbb{I}\left\{s(X_{i_1}^{(1)}) < \ldots < s(X_{i_K}^{(K)})\right\}}{n_1 \times \cdots \times n_K},$$

- But the number of terms to be summed is **prohibitive**!

$$n_1 \times \ldots \times n_K$$

Example: Ranking

- A natural empirical counterpart of $L(s)$ is

$$\widehat{L}_{\mathbf{n}}(s) = \frac{\sum_{i_1=1}^{n_1} \cdots \sum_{i_K=1}^{n_K} \mathbb{I}\left\{s(X_{i_1}^{(1)}) < \ldots < s(X_{i_K}^{(K)})\right\}}{n_1 \times \cdots \times n_K},$$

- But the number of terms to be summed is **prohibitive**!

$$n_1 \times \ldots \times n_K$$

- Maximization of $\widehat{L}_{\mathbf{n}}(s)$ is **computationally unfeasible...**

Generalized U-statistics

- $K \geq 1$ samples and degrees $(d_1, \ldots, d_K) \in \mathbb{N}^{*K}$
- $(X_1^{(k)}, \ldots, X_{n_k}^{(k)})$, $1 \leq k \leq K$, K independent i.i.d. samples drawn from $F_k(dx)$ on $\mathcal{X}_k$ respectively
- **Kernel** $H : \mathcal{X}_1^{d_1} \times \cdots \times \mathcal{X}_K^{d_K} \to \mathbb{R}$, square integrable w.r.t. $\mu = F_1^{\otimes d_1} \otimes \cdots \otimes F_K^{\otimes d_K}$

Generalized U-statistics

Definition

The K-sample U-statistic of degrees $(d_1, \ldots, d_K)$ with kernel H is

$$U_{\mathbf{n}}(H) = \frac{\sum_{I_1} \cdots \sum_{I_K} H(\mathbf{X}_{I_1}^{(1)}; \mathbf{X}_{I_2}^{(2)}; \ldots; \mathbf{X}_{I_K}^{(K)})}{\binom{n_1}{d_1} \times \cdots \binom{n_K}{d_K}},$$

where $\sum_{I_k}$ refers to summation over all $\binom{n_k}{d_k}$ subsets
$\mathbf{X}_{I_k}^{(k)} = (X_{i_1}^{(k)}, \ldots, X_{i_{d_k}}^{(k)})$ related to a set I_k of d_k indexes
$1 \leq i_1 < \ldots < i_{d_k} \leq n_k$

It is said symmetric when H is permutation symmetric in each set of d_k arguments $\mathbf{X}_{I_k}^{(k)}$.

References: Lee (1990)

Generalized U-statistics

- **Unbiased estimator** of

$$\theta(H) = \mathbb{E}[H(X_1^{(1)}, \ldots, X_{d_1}^{(1)}, \ldots, X_1^{(K)}, \ldots, X_{d_k}^{(K)})]$$

  with **minimum variance**

- **Asymptotically Gaussian** as $n_k/n \to \lambda_k > 0$ for $k = 1, \ldots, K$

- Its computation requires the summation of

$$\prod_{k=1}^{K} \binom{n_k}{d_k} \text{ terms}$$

- K-partite ranking: $d_k = 1$ for $1 \leq k \leq K$

$$\mathrm{H}_s(x_1, \ldots, x_K) = \mathbb{I}\{s(x_1) < s(x_2) < \cdots < s(x_K)\}$$

Incomplete U-statistics

- Replace $U_{\mathbf{n}}(H)$ by an **incomplete** version, involving much less terms
- Build a set $\mathcal{D}_B$ of cardinality B built by **sampling with replacement** in the set Λ of indexes

$$((i_1^{(1)}, \ldots, i_{d_1}^{(1)}), \ldots, (i_1^{(K)}, \ldots, i_{d_K}^{(K)}))$$

with $1 \leq i_1^{(k)} < \ldots < i_{d_k}^{(k)} \leq n_k,\ 1 \leq k \leq K$

- Compute the Monte-Carlo version based on B terms

$$\widetilde{U}_B(H) = \frac{1}{B} \sum_{(I_1, \ldots, I_K) \in \mathcal{D}_B} H(X_{I_1}^{(1)}, \ldots, X_{I_K}^{(K)})$$

- An incomplete U-statistic is **NOT a U-statistic**

M-Estimation based on incomplete U-statistics

- Replace the criterion by a tractable incomplete version based on $B = O(n)$ terms

$$\min_{H \in \mathcal{H}} \widetilde{U}_B(H)$$

- This leads to investigate the maximal deviations

$$\sup_{H \in \mathcal{H}} \left| \widetilde{U}_B(H) - U_{\mathbf{n}}(H) \right|$$

Main Result

Theorem

Let $\mathcal{H}$ be a VC *major class of bounded symmetric kernels of finite* VC *dimension $\mathcal{V} < +\infty$. Set $\mathcal{M}_{\mathcal{H}} = \sup_{(H,x)\in\mathcal{H}\times\mathcal{X}} |H(x)|$. Then,*

(i) $\mathbb{P}\left\{\sup_{H\in\mathcal{H}} \left|\widetilde{U}_B(H) - U_{\mathbf{n}}(H)\right| > \eta\right\} \leq$
$2(1+\#\Lambda)^V \times e^{-B\eta^2/\mathcal{M}_{\mathcal{H}}^2}$

(ii) *for all $\delta \in (0,1)$, with probability at least $1-\delta$, we have:*

$$\frac{1}{\mathcal{M}_{\mathcal{H}}} \sup_{H\in\mathcal{H}} \left|\widetilde{U}_B(H) - \mathbb{E}\left[\widetilde{U}_B(H)\right]\right| \leq 2\sqrt{\frac{2\mathcal{V}\log(1+\kappa)}{\kappa}} + \sqrt{\frac{\log(2/\delta)}{\kappa}} + \sqrt{\frac{\mathcal{V}\log(1+\#\Lambda)+\log(4/\delta)}{B}},$$

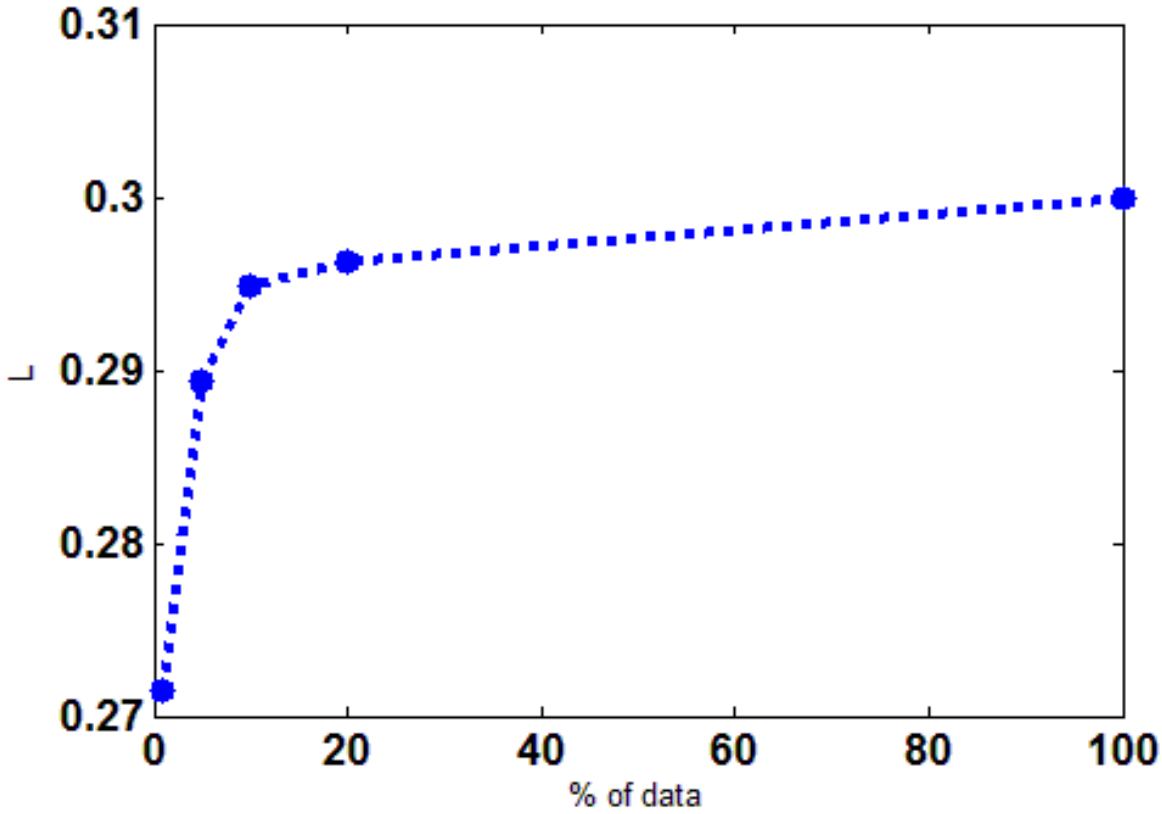
where $\kappa = \min\{\lfloor n_1/d_1\rfloor, \ldots, \lfloor n_K/d_K\rfloor\}$

Consequences

- Empirical risk sampling with $B = O(n)$ yields a rate bound of the order $O(\sqrt{\log n / n})$
- One suffers **no loss** in terms of learning rate, while **drastically reducing computational cost**

Example: Ranking

Empirical ranking performance for SVMRANK based on 1%, 5%, 10%, 20% and 100% of the "LETOR 2007" dataset.



Recommender systems are everywhere

E-commerce
Products to buy

Entertainment
Movies, music, books

Tourism
Hotels, restaurants

Social networks
Contacts

Targeted advertising
Ads

Objective: Make the best recommendations for each user

General principle

1. Estimate the taste of a user for a recommendation from the analysis of her feedback data
2. Recommend the items **she should like**

Possible additional criteria

- Diversity of recommendations
- Prices of recommended items
- Life cycle of recommended items

Possible methods

Item-based recommendation

- Recommend to a user items with similar features to the ones she likes

Collaborative filtering

- Recommend to a user items that other users with similar tastes also like

Hybrid

- Combination of both

General approach: Model user feedback by quantitative values

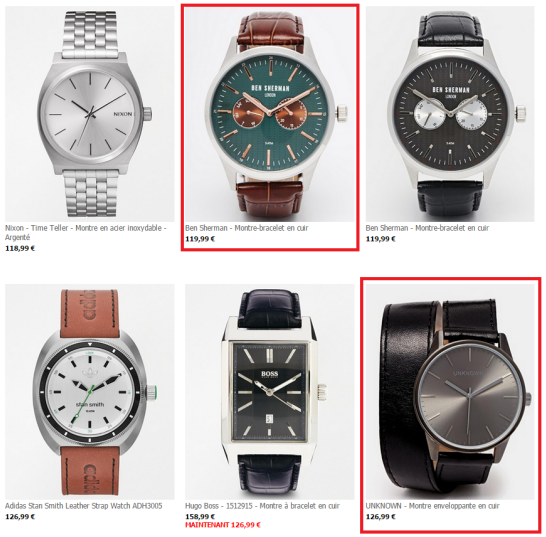
3

4

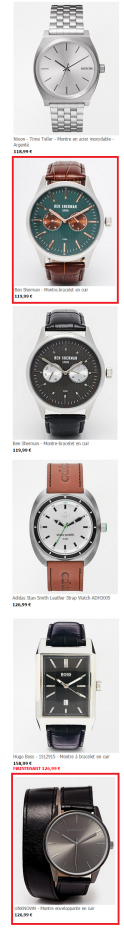
5

4

General approach: Model user feedback by quantitative values



$\Rightarrow$



0

1

0

0

0

1

General approach: Model user feedback by quantitative values

Advantages

- Results are satisfying enough
- Mathematically simple

Drawbacks

- Need to deal with rating scale bias
- Only exploits partial information from user feedback

Feedback data also contain information about relative preferences

3

4

5

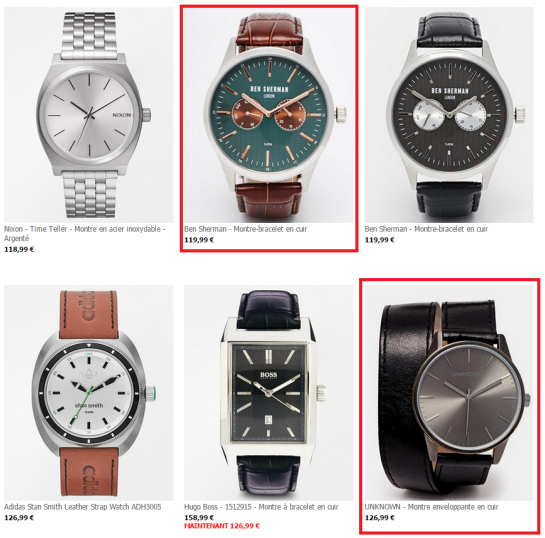
4

$\Rightarrow$

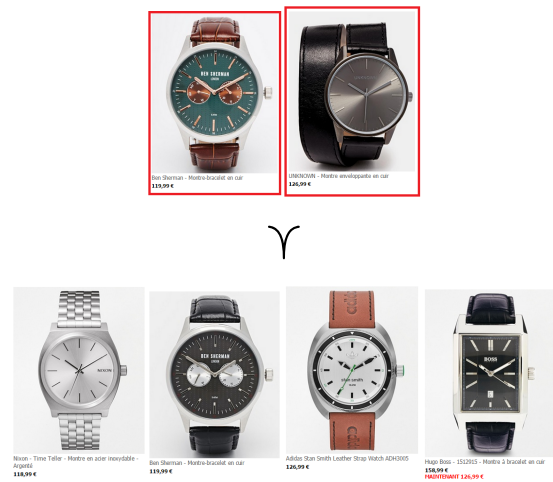
$\succ$

$\succ$

Feedback data also contain information about relative preferences



$\Rightarrow$



Outline

Modeling preference data

Set of items: $[\![n]\!] := \{1, \ldots, n\}$

We consider preferences without ties: $a_1 \succ \cdots \succ a_k$

- $k = n$: Preference is **complete** over $[\![n]\!]$
- $k < n$: Preference is **incomplete** over $[\![n]\!]$

A complete preference induces an incomplete preference on a subset of items $A \subset [\![n]\!]$. Example for $n = 5$:

$$2 \succ 4 \succ 1 \succ 5 \succ 3 \qquad \Longrightarrow \qquad 4 \succ 1 \succ 3 \quad \text{over } A = \{1, 3, 4\}$$

Modeling preference data

The global variability of the data is modeled by a probability distribution p over the set of complete preferences on $\{1, \ldots, n\}$ with:

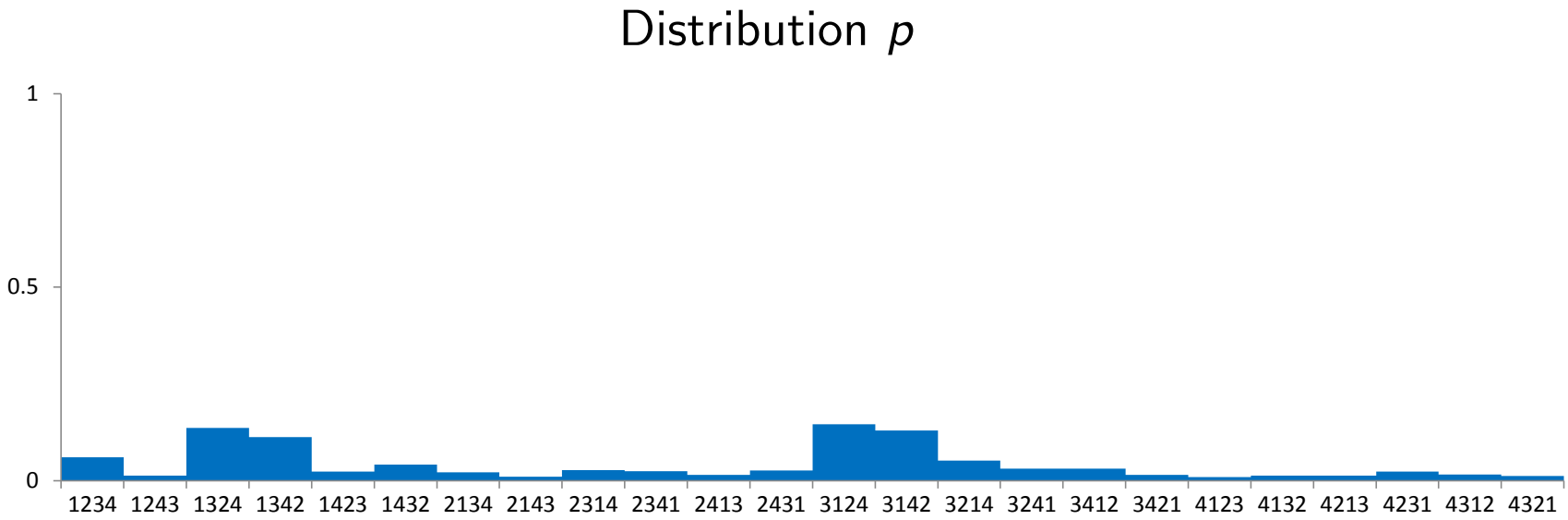
$$\mathbb{P}[a_1 \succ \cdots \succ a_k] = \sum_{\substack{\pi \text{ complete preference on } [\![n]\!] \\ \pi \text{ induces } a_1 \succ \cdots \succ a_k \text{ on } \{a_1, \ldots, a_k\}}} p(\pi)$$

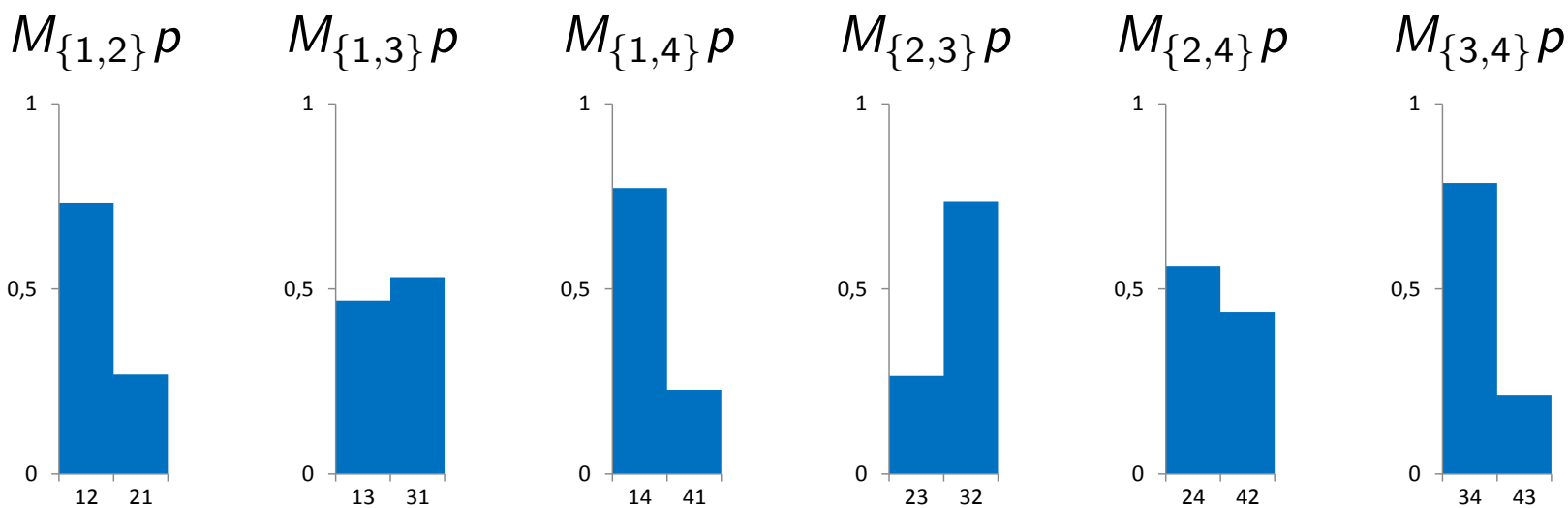
$$:= M_{\{a_1,\ldots,a_k\}} p(a_1 \succ \cdots \succ a_k) \qquad \textit{(notation)}$$

Example (for $n = 3$)

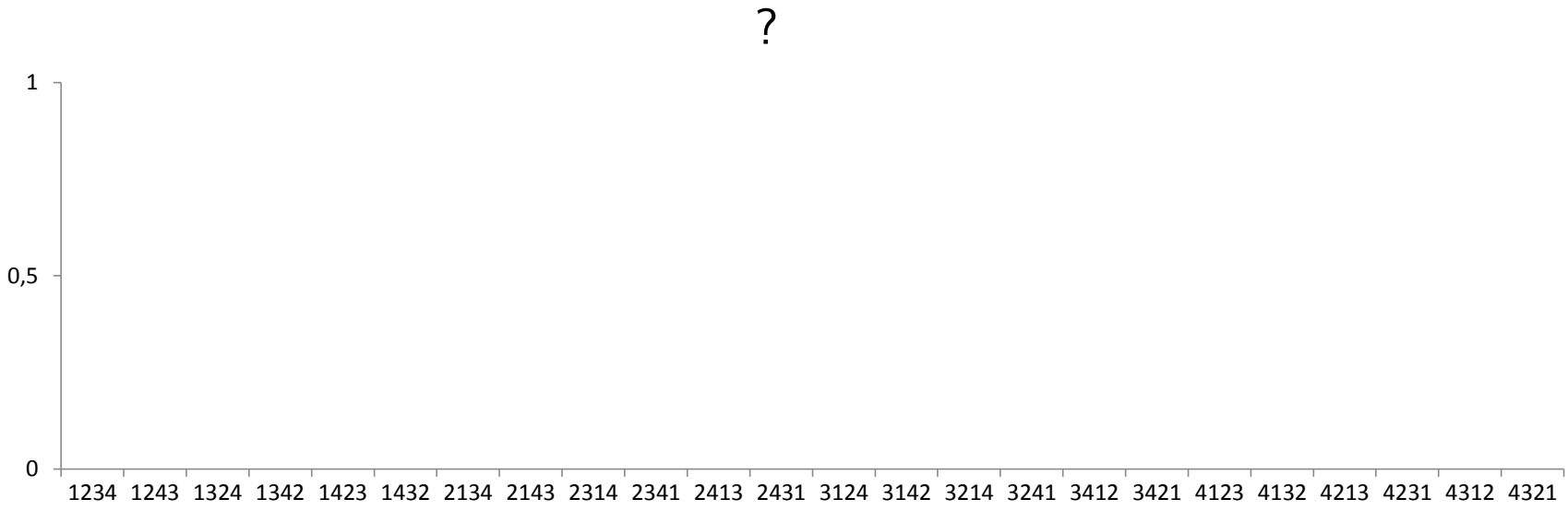
$$\mathbb{P}[1 \succ 3] = p(2 \succ 1 \succ 3) + p(1 \succ 2 \succ 3) + p(1 \succ 3 \succ 2)$$

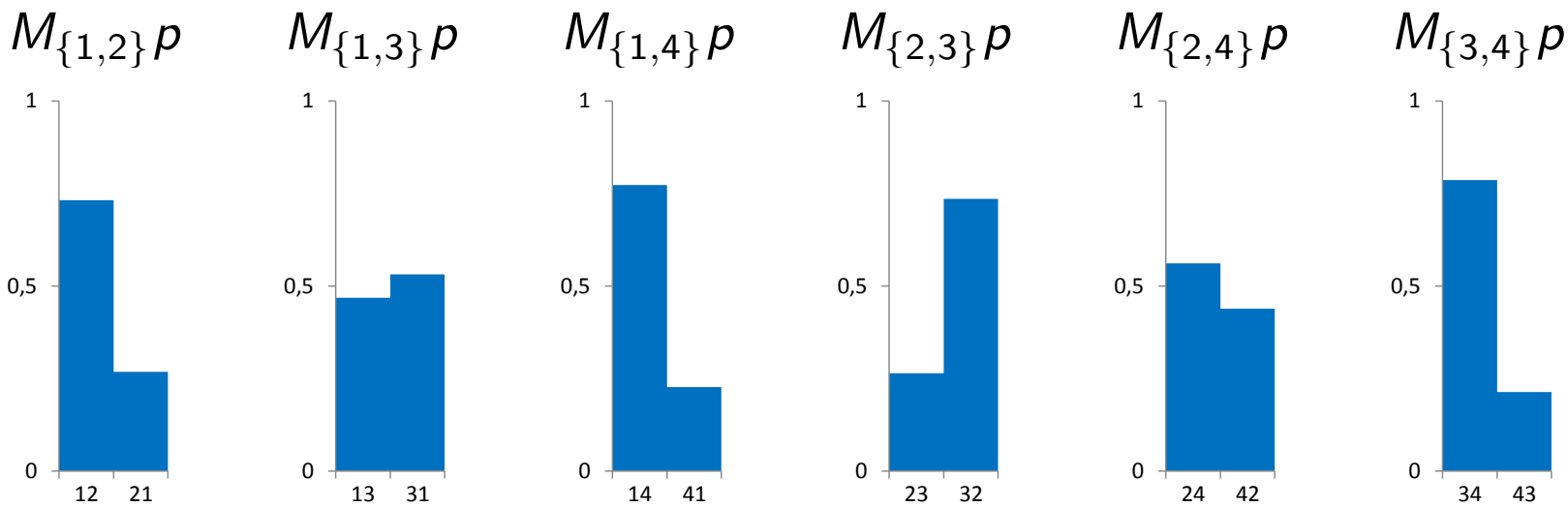
Modeling preference data





Challenge: Recover p from incomplete observations





Challenge: Recover p from incomplete observations

Example

$$\left.\begin{array}{l}\mathbb{P}[1 \prec 2] = 0,7 \\ \mathbb{P}[1 \prec 3] = 0,4 \\ \mathbb{P}[2 \prec 3] = 0,9\end{array}\right\} \quad \Rightarrow \quad \mathbb{P}[2 \prec 3 \prec 1] = ?$$

Problem

- p has $n! - 1$ parameters $\Rightarrow$ more than $10^{10\,000}$ parameters for a catalog of 10 000 items
- Very few applicable approaches from the literature

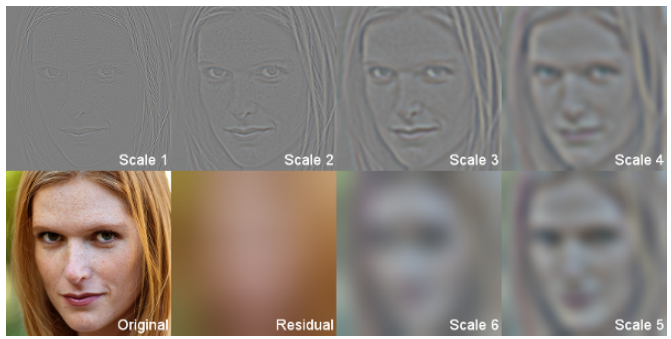
Efficient representation of the data

General principle

The real world has much less parameters.
$\Rightarrow$ **The great dimensionality of the problem is due to a bad representation of the data.**

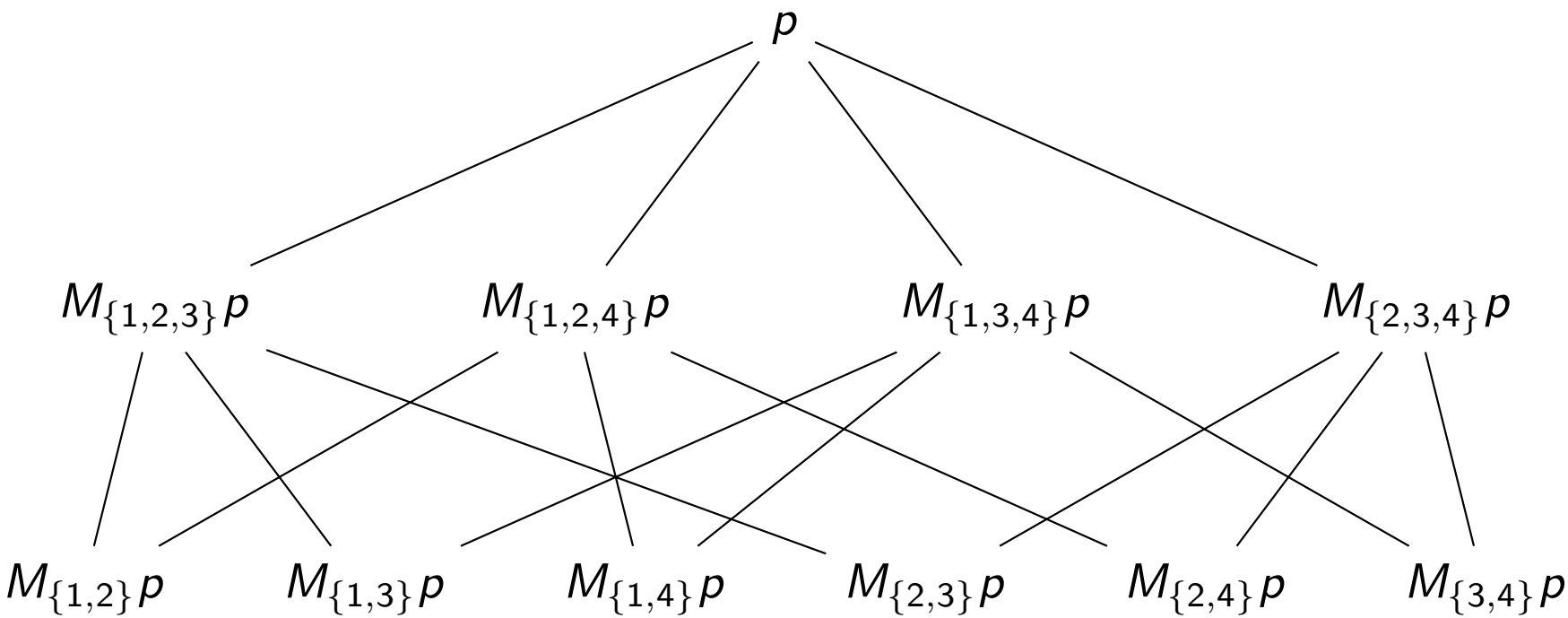
Analogy with images

- An image of 10 megapixels : 10^7 parameters $\Rightarrow$ Not efficient representation
- Wavelet decomposition: selection of the most relevant components $\Rightarrow$ Efficient representation

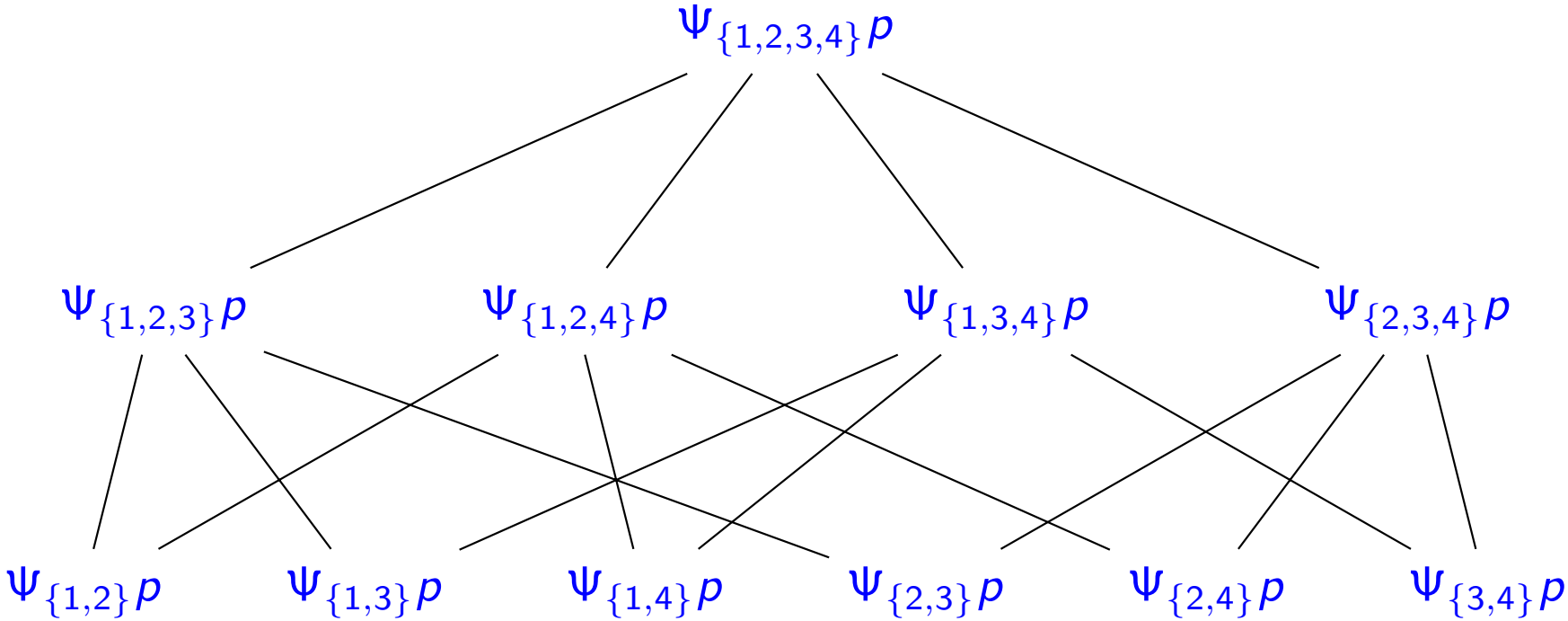


Outline

Our solution: a wavelet decomposition of preference data



The MRA representation allows to localize the part of information of each relative marginal



Our solution: a wavelet decomposition of preference data

Theorem

Any probability distribution p over complete preferences decomposes as

$$p = \frac{1}{n!} + \phi_{[\![n]\!]} \sum_{k=2}^{n} \sum_{|B|=k} \Psi_B p$$

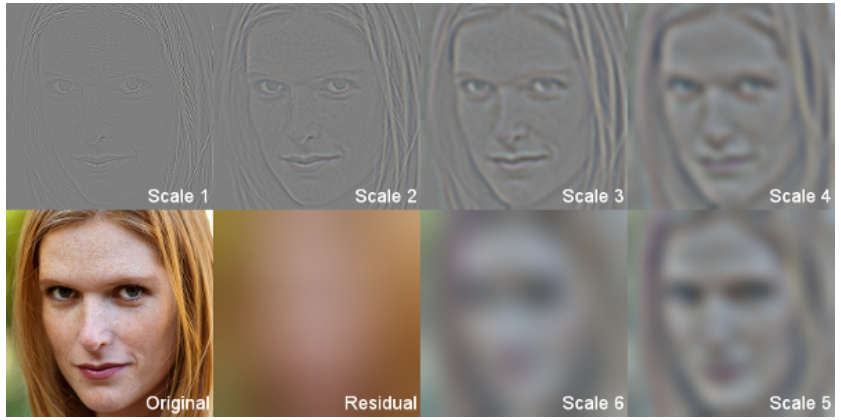
where $\Psi_B p$ "localizes the part of information specific to B":

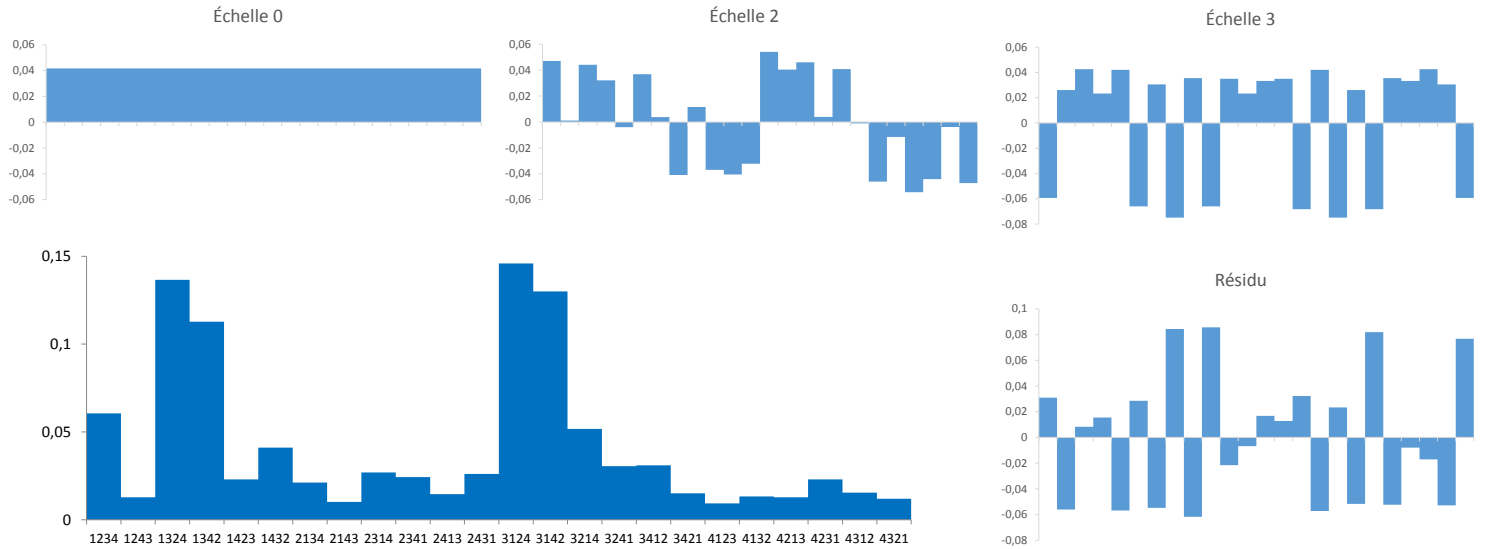
$$M_A p = \frac{1}{|A|!} + \phi_A \sum_{k=2}^{n} \sum_{\substack{|B|=k \\ B \subset A}} \Psi_B p$$

(the ϕ_A's for $A \subset [\![n]\!]$ are easily computable embedding operators).

$\Rightarrow$ **Drastic reduction of the dimensionality**

Analogy with images

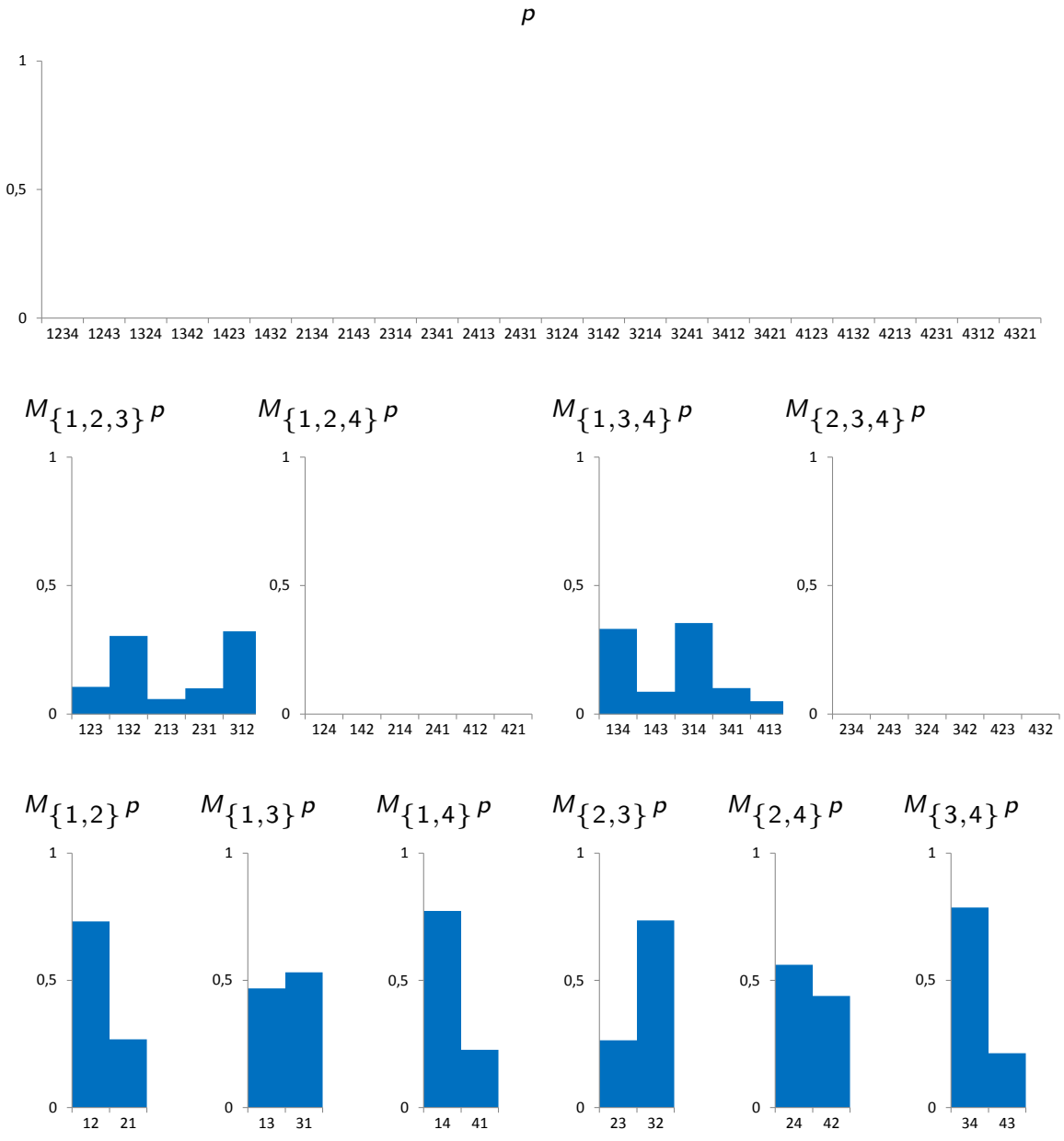




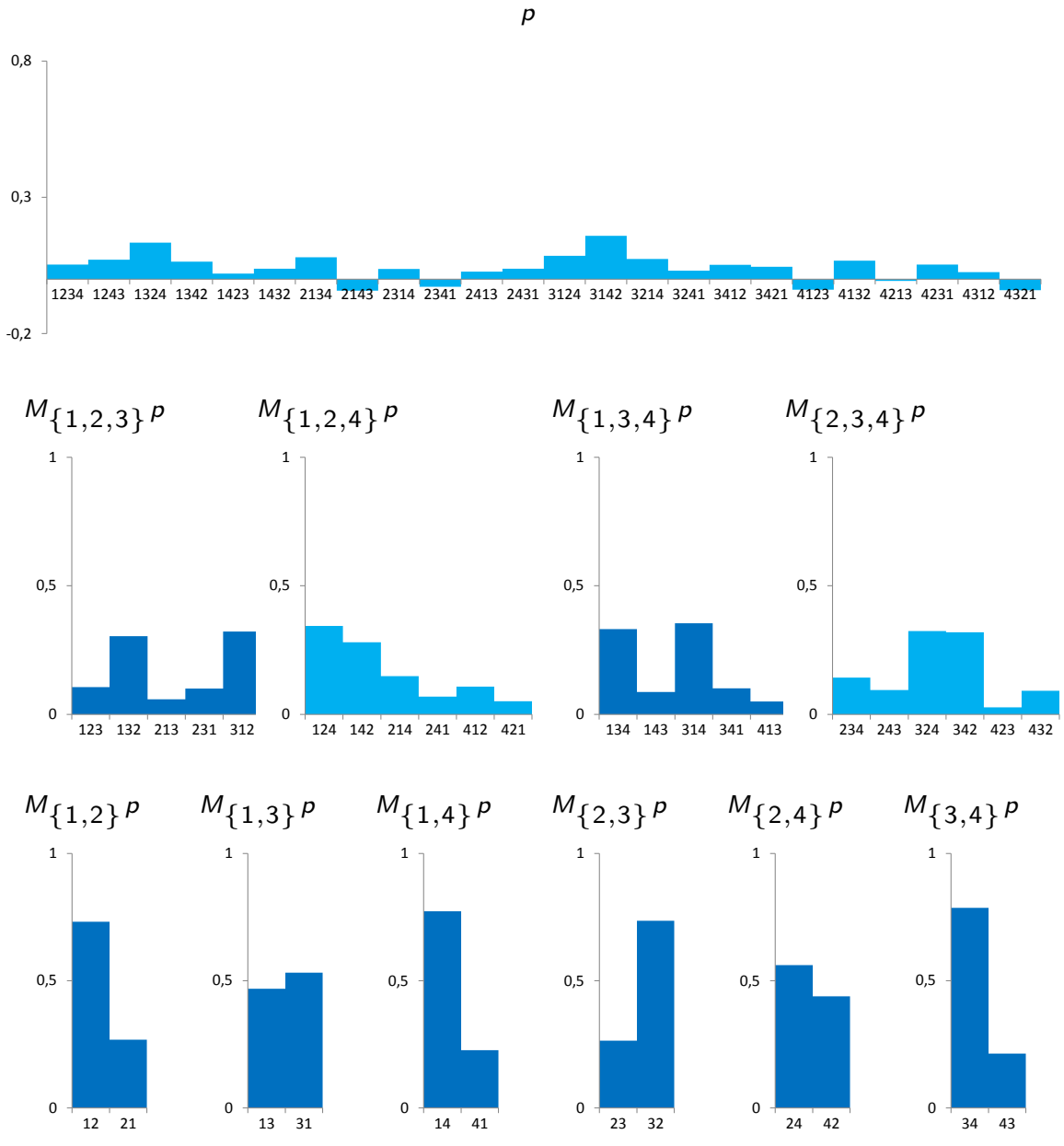
Ingredients of the proof

- Linear algebra
- Combinatorics of words
- Recent result in algebraic topology

Application: recovering p



Application: recovering p



Ongoing work

Applications to recommender systems

- Clustering of users
- Ranking of recommendations
- Active learning of preferences for cold-start recommendation

Theoretical developments

- Regularization procedures
- Combination with similarity measures on items

Development of a library

- In python
- In a map/reduce framework

Thank you

References

- Sibony, E., Clémençon, S. & Jakubowicz, J. A Multiresolution Framework for the Statistical Analysis of Incomplete Rankings. *ArXiv e-prints*, 2016.

- Sibony, E., Clémençon, S. & Jakubowicz, J. MRA-based Statistical Learning From Incomplete Rankings. In *Proceedings of ICML*, 2015.

- Sibony, E., Clémençon, S. & Jakubowicz, J. Multiresolution analysis of incomplete rankings with applications to prediction. In *Proceedings of IEEE Big Data*, 2014.

- Clémençon, S., Jakubowicz, J. & Sibony, E. Multiresolution analysis of incomplete rankings. *ArXiv e-prints*, 2014.