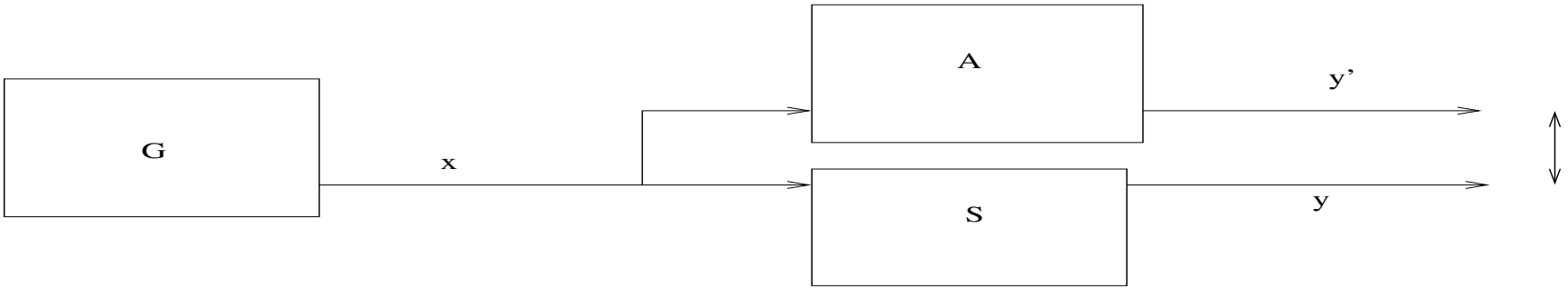
# Apprentissage pour la RI

Eric Gaussier

AMA/LIG
Université Grenoble Alpes
Eric.Gaussier@imag.fr

# Table des matières

# Qu'est-ce que l'apprentissage ?




1. Ensemble *fini* de couples $(x, y)$, $\mathcal{D} = ((x^{(1)}, y^{(1)}), \cdots, (x^{(n)}, y^{(n)}))$ : *ensemble d'apprentissage*
2. $x \in \mathbb{R}^p$ (document représenté par un vecteur), $y \in \mathcal{Y}$ (catégorisation binaire : $\mathcal{Y} = \{0, 1\}$)
3. **Apprenant A** sélectionne, parmi un ensemble donné $\mathcal{F}$, la fonction $f$ qui lui semble la plus appropriée : $y' = f(x)$

Comment sélectionner $f$ ?

# Mesure de la qualité d'une fonction apprise

1. Fonction de coût (*loss function*) :

$$L : \mathcal{Y}x\mathcal{Y} \rightarrow \mathbb{R}^+, \text{telle que } L(y, y') > 0 \text{ pour } y \neq y'$$

Coût 0-1 : $L(y^{(i)}, f(x^{(i)})) = 0$ si $y^{(i)} = f(x^{(i)}), 1$ sinon

2. Risque, risque fonctionnel, erreur de généralisation :
$R(f) = \mathbb{E}_{P(x,y)} L(y, f(x))$

$$\min_{f \in \mathcal{F}} R(f)$$

3. Risque empirique : $\text{Remp}(f; \mathcal{D}) = \frac{1}{n} \sum_{i=1}^{n} L(y^{(i)}, f(x^{(i)}))$

$$\min_{f \in \mathcal{F}} \text{Remp}(f) ?$$

# Mesure de la qualité d'une fonction apprise

1. Fonction de coût (*loss function*) :

$$L : \mathcal{Y}x\mathcal{Y} \rightarrow \mathbb{R}^+, \text{telle que } L(y, y') > 0 \text{ pour } y \neq y'$$

Coût 0-1 : $L(y^{(i)}, f(x^{(i)})) = 0$ si $y^{(i)} = f(x^{(i)}), 1$ sinon

2. Risque, risque fonctionnel, erreur de généralisation :
$R(f) = \mathbb{E}_{P(x,y)} L(y, f(x))$

$$\min_{f \in \mathcal{F}} R(f)$$

3. Risque empirique : $\text{Remp}(f; \mathcal{D}) = \frac{1}{n} \sum_{i=1}^{n} L(y^{(i)}, f(x^{(i)}))$

$$\min_{f \in \mathcal{F}} \text{Remp}(f) ?$$

# Mesure de la qualité d'une fonction apprise

1. Fonction de coût (*loss function*) :

$$L : \mathcal{Y}x\mathcal{Y} \rightarrow \mathbb{R}^+, \text{telle que } L(y, y') > 0 \text{ pour } y \neq y'$$

Coût 0-1 : $L(y^{(i)}, f(x^{(i)})) = 0$ si $y^{(i)} = f(x^{(i)}), 1$ sinon

2. Risque, risque fonctionnel, erreur de généralisation :
$R(f) = \mathbb{E}_{P(x,y)} L(y, f(x))$

$$\min_{f \in \mathcal{F}} R(f)$$

3. Risque empirique : $\text{Remp}(f; \mathcal{D}) = \frac{1}{n} \sum_{i=1}^{n} L(y^{(i)}, f(x^{(i)}))$

$$\min_{f \in \mathcal{F}} \text{Remp}(f)?$$

# Mesure de la qualité d'une fonction apprise

1. Fonction de coût (*loss function*) :

$$L : \mathcal{Y}x\mathcal{Y} \rightarrow \mathbb{R}^+, \text{telle que } L(y, y') > 0 \text{ pour } y \neq y'$$

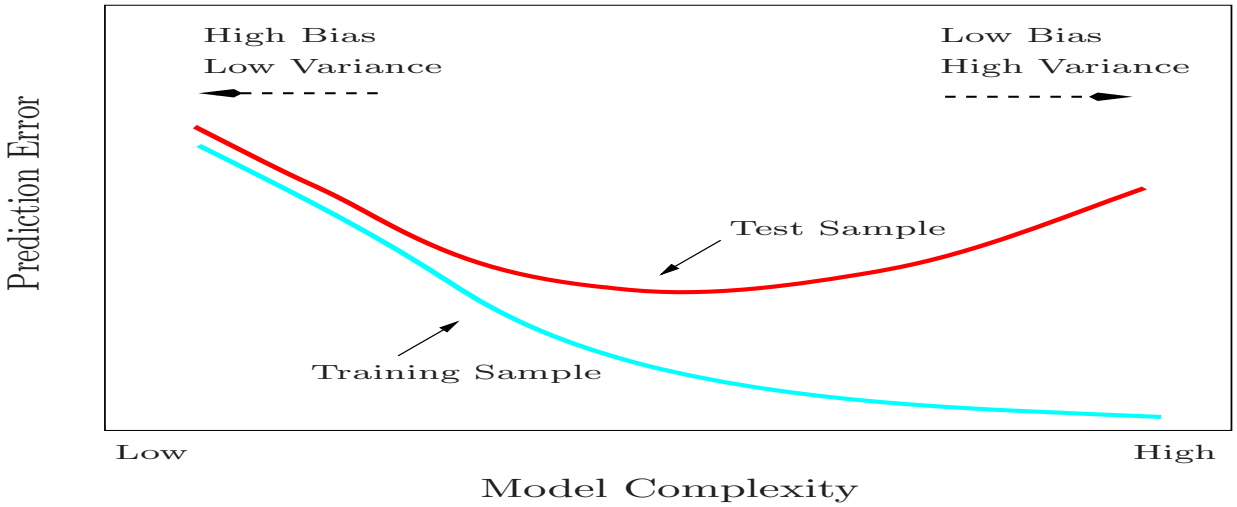
Coût 0-1 : $L(y^{(i)}, f(x^{(i)})) = 0$ si $y^{(i)} = f(x^{(i)}), 1$ sinon

2. Risque, risque fonctionnel, erreur de généralisation :
$R(f) = \mathbb{E}_{P(x,y)} L(y, f(x))$

$$\min_{f \in \mathcal{F}} R(f)$$

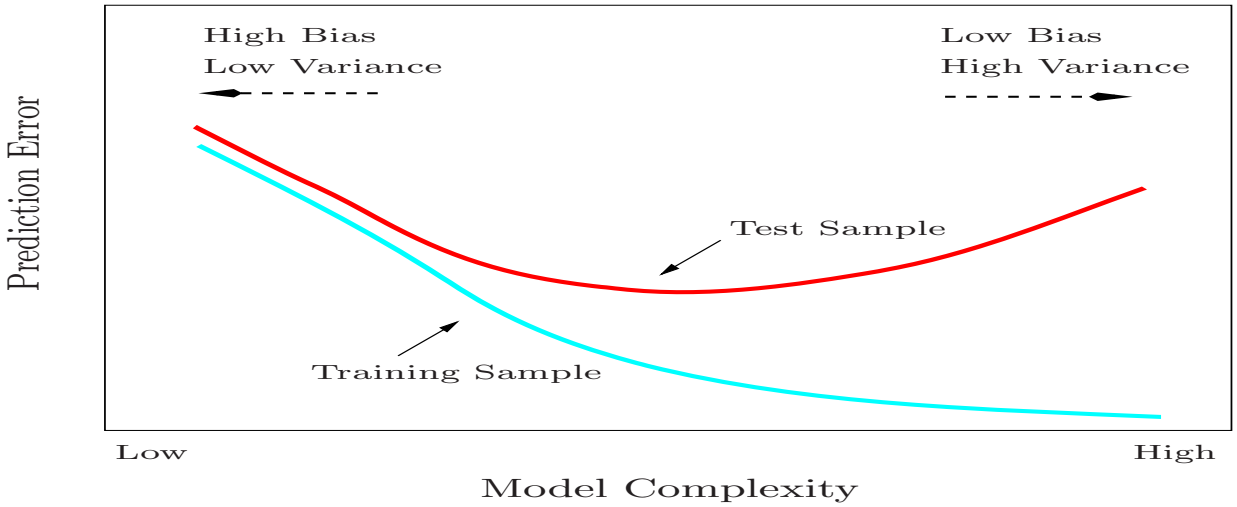
3. Risque empirique : $\text{Remp}(f; \mathcal{D}) = \frac{1}{n} \sum_{i=1}^{n} L(y^{(i)}, f(x^{(i)}))$

$$\min_{f \in \mathcal{F}} \text{Remp}(f)?$$



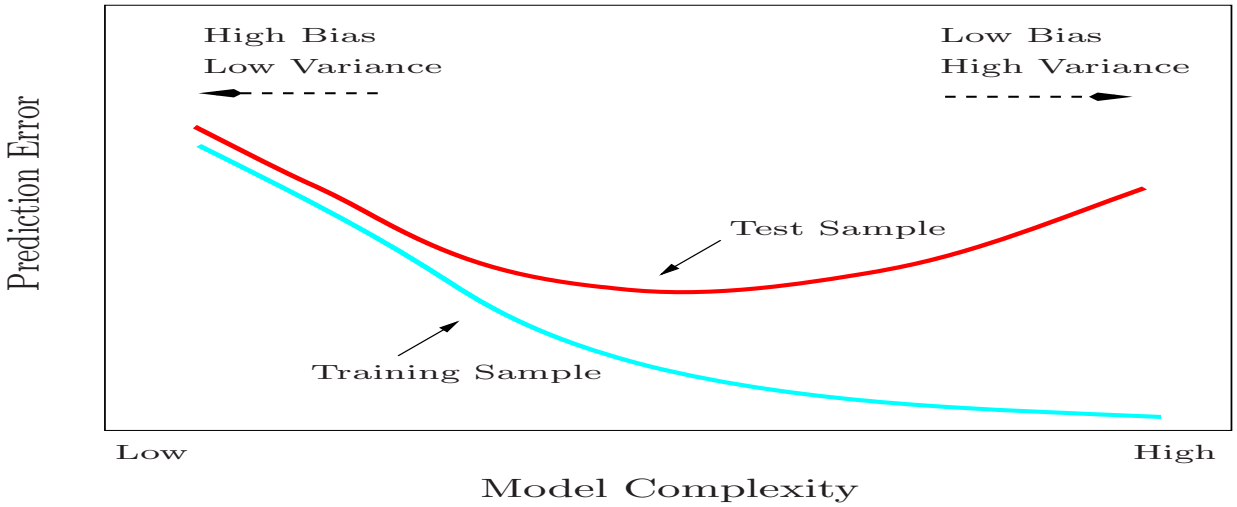

Soution : $\min_{f \in \mathcal{F}} \text{Remp}(f) + \lambda \Omega(f)$

*Image tirée de Elements of statistical learning. Hastie, Tibshirani, Friedman. Springer*




Soution : $\min_{f \in \mathcal{F}} \text{Remp}(f) + \lambda \Omega(f)$

*Image tirée de Elements of statistical learning. Hastie, Tibshirani, Friedman. Springer*




Soution : $\min_{f \in \mathcal{F}} \text{Remp}(f) + \lambda \Omega(f)$

*Image tirée de Elements of statistical learning. Hastie, Tibshirani, Friedman. Springer*

# Algorithmes d'apprentissage

Un algorithme d'apprentissage :

- a accès à un ensemble de fonctions $\mathcal{F}$ (par ex. perceptrons multi-couches) ;
- sélectionne la fonction la plus approprié à partir de l'ens. d'apprentissage et de la fonction objectif définie par l'utilisateur ;
- réalise cette sélection suivant des méthodes propres (par ex. descente de gradient stochastique (*stochastic gradient descent (SGD)*).

L'utilisateur définit :

- la fonction de coût qui l'intéresse (un grand nombre de fonctions de coût existent et peuvent être utilisées - attention à la dérivabilité) ;
- la fonction de régularisation $\Omega$ qui lui semble la plus appropriée (là aussi, beaucoup de fonctions existent - régularisation $L_1$, $L_2$, ...).

Un choix crucial : la représentation des exemples

# Algorithmes d'apprentissage

Un algorithme d'apprentissage :

- a accès à un ensemble de fonctions $\mathcal{F}$ (par ex. perceptrons multi-couches) ;
- sélectionne la fonction la plus approprié à partir de l'ens. d'apprentissage et de la fonction objectif définie par l'utilisateur ;
- réalise cette sélection suivant des méthodes propres (par ex. descente de gradient stochastique (*stochastic gradient descent (SGD)*).

L'utilisateur définit :

- la fonction de coût qui l'intéresse (un grand nombre de fonctions de coût existent et peuvent être utilisées - attention à la dérivabilité) ;
- la fonction de régularisation $\Omega$ qui lui semble la plus appropriée (là aussi, beaucoup de fonctions existent - régularisation $L_1$, $L_2$, ...).

Un choix crucial : la représentation des exemples

# Algorithmes d'apprentissage

Un algorithme d'apprentissage :

- a accès à un ensemble de fonctions $\mathcal{F}$ (par ex. perceptrons multi-couches) ;
- sélectionne la fonction la plus approprié à partir de l'ens. d'apprentissage et de la fonction objectif définie par l'utilisateur ;
- réalise cette sélection suivant des méthodes propres (par ex. descente de gradient stochastique (*stochastic gradient descent (SGD)*).

L'utilisateur définit :

- la fonction de coût qui l'intéresse (un grand nombre de fonctions de coût existent et peuvent être utilisées - attention à la dérivabilité) ;
- la fonction de régularisation $\Omega$ qui lui semble la plus appropriée (là aussi, beaucoup de fonctions existent - régularisation $L_1$, $L_2$, ...).

Un choix crucial : la représentation des exemples

# *Feature engineering vs representation learning*

1. Effort mis dans la recherche de la meilleure représentation

2. Effort mis dans l'architecture qui permet d'apprendre une représentation adaptée (nécessité néanmoins d'une première représentation)

# Modéliser la RI comme un problème de catégorisation

Quelle est la difficulté ?

Choisir une représentation des exemples ? Choisir le nombre de classes ? Choisir un algorithme d'apprentissage ?

- Nombre de classes : 2 (pertinent, non-pertinent) - le score peut de catégorisation peut être utilisé pour trier les documents
- N'importe quel algorithme de catégorisation !
- Quelle représentation des exemples ?

# Modéliser la RI comme un problème de catégorisation

## Quelle est la difficulté ?

Choisir une représentation des exemples ? Choisir le nombre de classes ? Choisir un algorithme d'apprentissage ?

- Nombre de classes : 2 (pertinent, non-pertinent) - le score peut de catégorisation peut être utilisé pour trier les documents
- N'importe quel algorithme de catégorisation !
- Quelle représentation des exemples ?

# Représentation des exemples

Problème crucial : quels types d'exemples considérer ? Docs ? ...

Représentation standard, $x = (q, d) \in \mathbb{R}^m$. Les coordonnées $(f_i(q, d), i = 1, \cdots, p)$ sont très générales. On essaye de se reposer sur un maximum d'information :

- $f_1(q,d) = \sum_{t \in q \bigcap d} \log(t^d)$, $f_2(q,d) = \sum_{t \in q} \log(1 + \frac{t^d}{|\mathcal{C}|})$
- $f_3(q,d) = \sum_{t \in q \bigcap d} \log(\text{idf}(t))$, $f_4(q,d) = \sum_{t \in q \bigcap d} \log(\frac{|\mathcal{C}|}{t^{\mathcal{C}}})$
- $f_5(q,d) = \sum_{t \in q} \log(1 + \frac{t^d}{|\mathcal{C}|}\text{idf}(t))$, $f_6(q,d) = \sum_{t \in q} \log(1 + \frac{t^d}{|\mathcal{C}|}\frac{|\mathcal{C}|}{t^{\mathcal{C}}})$
- $f_7(q,d) = \text{RSV}_{\text{vect}}(q,d)$, ...

# Représentation des exemples

Problème crucial : quels types d'exemples considérer ? Docs ? ...
Représentation standard, $x = (q, d) \in \mathbb{R}^m$. Les coordonnées $(f_i(q, d), i = 1, \cdots, p)$ sont très générales. On essaye de se reposer sur un maximum d'information :

- $f_1(q,d) = \sum_{t \in q \bigcap d} \log(t^d)$, $f_2(q,d) = \sum_{t \in q} \log(1 + \frac{t^d}{|\mathcal{C}|})$
- $f_3(q,d) = \sum_{t \in q \bigcap d} \log(\text{idf}(t))$, $f_4(q,d) = \sum_{t \in q \bigcap d} \log(\frac{|\mathcal{C}|}{t^{\mathcal{C}}})$
- $f_5(q,d) = \sum_{t \in q} \log(1 + \frac{t^d}{|\mathcal{C}|}\text{idf}(t))$, $f_6(q,d) = \sum_{t \in q} \log(1 + \frac{t^d}{|\mathcal{C}|}\frac{|\mathcal{C}|}{t^{\mathcal{C}}})$
- $f_7(q,d) = \text{RSV}_{\text{vect}}(q,d)$, ...

# Remarques sur cette approche

1. Cas d'une pertinence multivaluée : catégorisation multi-classes
2. Méthode qui permet d'attribuer un score, pour une requête donnée, à un document, indépendamment des autres (méthode dite *pointwise*)
3. Résultats comparables à ceux des modèles probabilistes dans le cas de collections "classiques", meilleurs dans le cas du Web (espace d'attributs plus riches)
4. Méthode qui repose sur une notion "absolue" de pertinence
5. La fonction objectif est "éloignée" de la fonction d'évaluation
6. Disponibilité des annotations ?

# Table des matières

# Les paires de préférence

• La notion de pertinence n'est pas une notion absolue. Il est souvent plus facile de juger de la pertinence relative de deux documents

• Les jugements par paires constituent en fait les jugements les plus généraux

# Représentation des données

On cherche ici une fonction $f$ qui respecte l'ordre :
$d^{(i)} \prec_q d^{(j)} \iff f(d^{(i)}) <_q f(d^{(j)})$
Transformer une information d'ordre en une information de catégorie On forme la différence entre deux éléments fondamentaux $x'_i = (d_i, q)\ (1 \leq i \leq n)$ :

$$\{(x^{(1)} = (x'^{(1)}_1 - x'^{(1)}_2), z^{(1)}), \cdots, (x^{(p)} = (x'^{(p)}_1 - x'^{(p)}_2), z^{(p)})\}$$

avec :

$$z^{(i)} = \begin{cases} +1 \text{ si } x'^{(i)}_2 \prec x'^{(i)}_1 \\ -1 \text{ si } x'^{(i)}_1 \prec x'^{(i)}_2 \end{cases}$$

# Ordonnancement = catégorisation

On peut utiliser (presque) n'importe quel algorithme de catégorisation sur l'ensemble d'apprentissage
$\{(x^{(1)} = (x'^{(1)}_1 - x'^{(1)}_2), z^{(1)}), \cdots, (x^{(p)} = (x'^{(p)}_1 - x'^{(p)}_2), z^{(p)})\}$

Comment trier les documents pour des nouvelles requêtes ?

Pour chaque requête, on considère $x'_2 = \vec{0}$ et on trie les documents sur le score obtenu

Propriété de *consistance* : $score(x'_1 - x'_2) > 0$ ssi $score(x'_1 - \vec{0}) > score(x'_1 - \vec{0})$

# Ordonnancement = catégorisation

On peut utiliser (presque) n'importe quel algorithme de catégorisation sur l'ensemble d'apprentissage
$\{(x^{(1)} = (x'^{(1)}_1 - x'^{(1)}_2), z^{(1)}), \cdots, (x^{(p)} = (x'^{(p)}_1 - x'^{(p)}_2), z^{(p)})\}$

Comment trier les documents pour des nouvelles requêtes ?

Pour chaque requête, on considère $x'_2 = \vec{0}$ et on trie les documents sur le score obtenu
Propriété de *consistance* : $score(x'_1 - x'_2) > 0$ ssi
$score(x'_1 - \vec{0}) > score(x'_1 - \vec{0})$

# Extensions

Approche *listwise*

- Traiter directement les listes triées comme des exemples d'apprentissage
- Deux grands types d'approche
  - Fonction objectif liée aux mesures d'évaluation
  - Fonction objectif définie sur des listes de documents
- Mais les mesures d'évaluation sont en général non continues

# Fonction objectif et mesures d'évaluation

- Approche standard, pour des problèmes continus et dérivables
  - Fonction objectif borne supérieure de la mesure d'évaluation (SVM-MAP)
  - Lissage des fonctions objectifs (SoftRank)
- Méthodes d'optimisation pour problèmes non continus
  - Algorithmes génétiques (RankGP)

# Table des matières

# Constituer des données d'apprentissage

- On dispose de données annotées pour plusieurs collections
  - TREC (TREC-vidéo)
  - CLEF
  - NTCIR
- Pour les entreprises (intranets), de telles données n'exitent pas en général $\rightarrow$ modèles standard, parfois faiblement supervisés
- Qu'en est-il du web ?

# Données d'apprentissages sur le web

- Une source importante d'information : les clics des utilisateurs
  - Utiliser les clics pour inférer des préférences entre documents (paires de préférence)
  - Compléter éventuellement par le temps passé sur le résumé d'un document (*eye-tracking*)
- Que peut-on déduire des clics ?

# Exploiter les clics (1)

Les clicks ne fournissent pas des jugements de pertinence absolus, mais relatifs. Soit un ordre $(d_1, d_2, d_3, \cdots)$ et $C$ l'ensemble des documents cliqués. Les stratégies suivantes peuvent être utilisées pour construre un ordre de pertinence entre documents :

1. Si $d_i \in C$ et $d_j \notin C$, $d_i \succ_{pert-q} d_j$
2. Si $d_i$ est le dernier doc cliqué, $\forall j < i$, $d_j \notin C$, $d_i \succ_{pert-q} d_j$
3. $\forall i \geq 2, d_i \in C, d_{i-1} \notin C, d_i \succ_{pert-q} d_{i-1}$
4. $\forall i, d_i \in C, d_{i+1} \notin C, d_i \succ_{pert-q} d_{i+1}$

# Exploiter les clics (2)

- Ces différentes stratégies permettent d'inférer un ordre partiel entre documents
- La collecte de ces données fournit un ensemble d'apprentissage très large, sur lequel on peut déployer les techniques vues précédemment
- La RI sur le web est en partie caractérisée par une course aux données :
  - Indexer le maximum de pages
  - Récupérer le maximum de données de clics

# Letor

http ://research.microsoft.com/en-us/um/beijing/projects/letor/

Tao Qin, Tie-Yan Liu, Jun Xu, and Hang Li. *LETOR : A Benchmark Collection for Research on Learning to Rank for Information Retrieval*, Information Retrieval Journal, 2010

Contient des jeux de données standard pour l'évaluation d'algorithmes d'ordonnancement

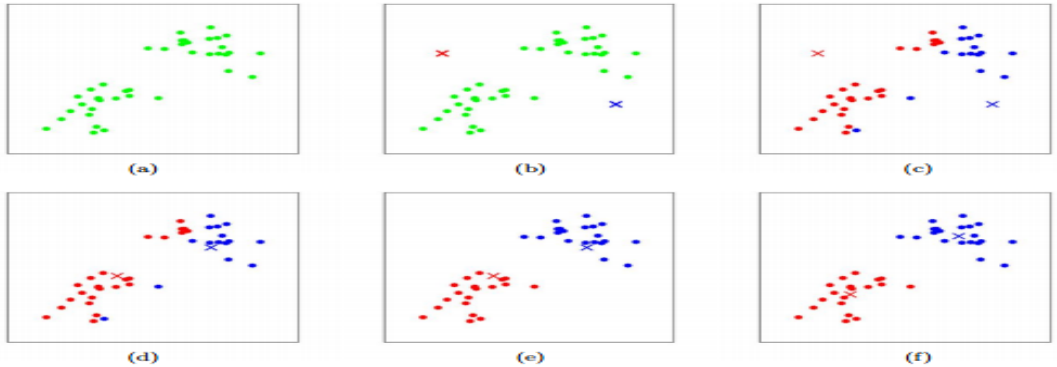
# Table des matières

# Apprentissage de représentations et clustering

Clustering est le processus visant à regrouper, au sein d'une même classe, les documents qui sont similaires

- Mesure de similarité/dissimilarité
- Nombre de classes ou seuil sur la mesure (sélection de modèles)
- À partir d'une représentation donnée

# Les $k$-moyennes : un algorithme standard de clustering



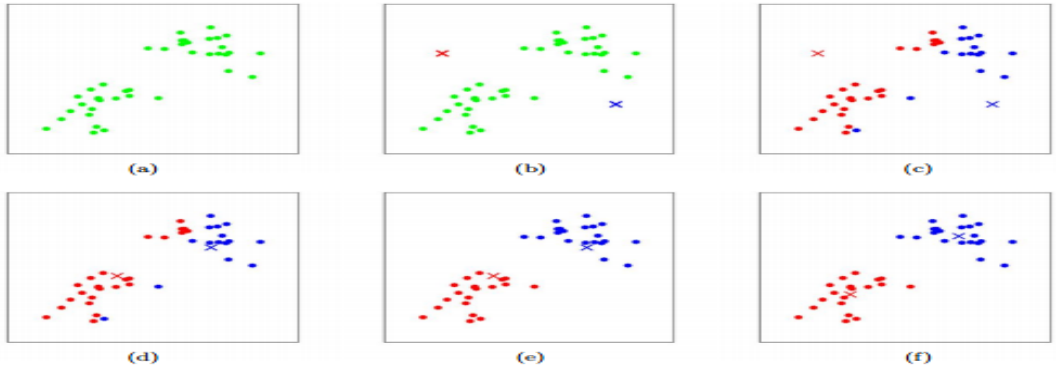

*Emprunté à C. Piech*

Step 1 Affecter les documents aux classes dont les représentants sont les plus proches

Step 2 Mettre à jour les représentants (moyenne des documents de la classe)

Prob. optim. : $\min_{\mathcal{R}} \sum_{x \in \mathcal{X}} ||\mathbf{x} - c(\mathbf{x}; \mathcal{R})||^2$ (avec $c(\mathbf{x}; \mathcal{R}) = \arg\min_{\mathbf{r} \in \mathcal{R}} ||\mathbf{x} - \mathbf{r}||^2$)

# Les $k$-moyennes : un algorithme standard de clustering



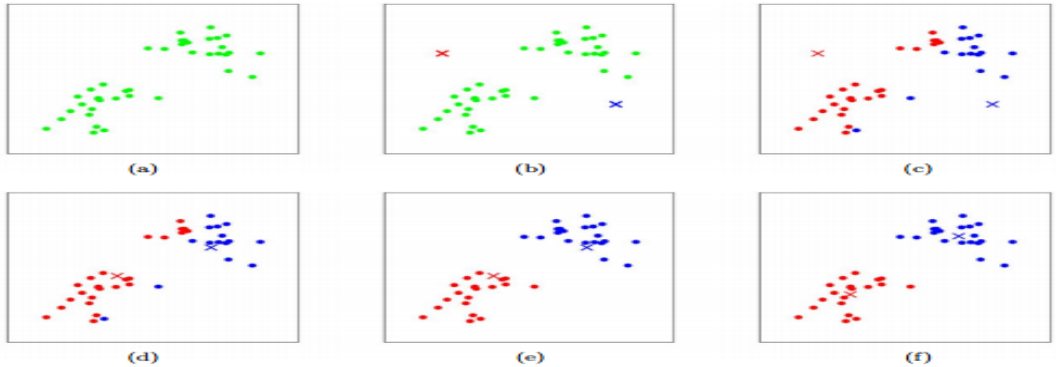

*Emprunté à C. Piech*

Step 1 Affecter les documents aux classes dont les représentants sont les plus proches

Step 2 Mettre à jour les représentants (moyenne des documents de la classe)

Prob. optim. : $\min_{\mathcal{R}} \sum_{x \in \mathcal{X}} ||\mathbf{x} - c(\mathbf{x}; \mathcal{R})||^2$ (avec $c(\mathbf{x}; \mathcal{R}) = \arg\min_{\mathbf{r} \in \mathcal{R}} ||\mathbf{x} - \mathbf{r}||^2$)

# Les $k$-moyennes : un algorithme standard de clustering




*Emprunté à C. Piech*

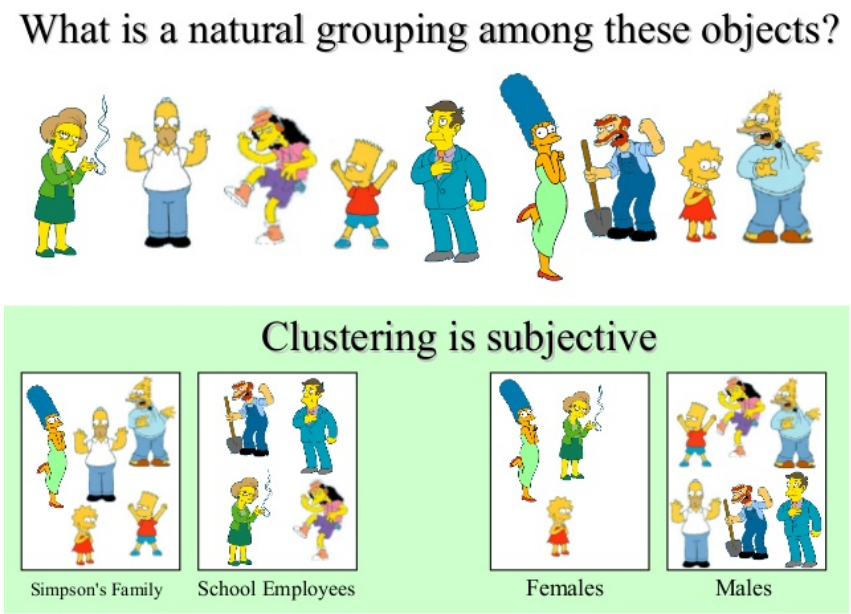
Step 1 Affecter les documents aux classes dont les représentants sont les plus proches

Step 2 Mettre à jour les représentants (moyenne des documents de la classe)

Prob. optim. : $\min_{\mathcal{R}} \sum_{x \in \mathcal{X}} ||\mathbf{x} - c(\mathbf{x}; \mathcal{R})||^2$ (avec $c(\mathbf{x}; \mathcal{R}) = \arg\min_{\mathbf{r} \in \mathcal{R}} ||\mathbf{x} - \mathbf{r}||^2$)

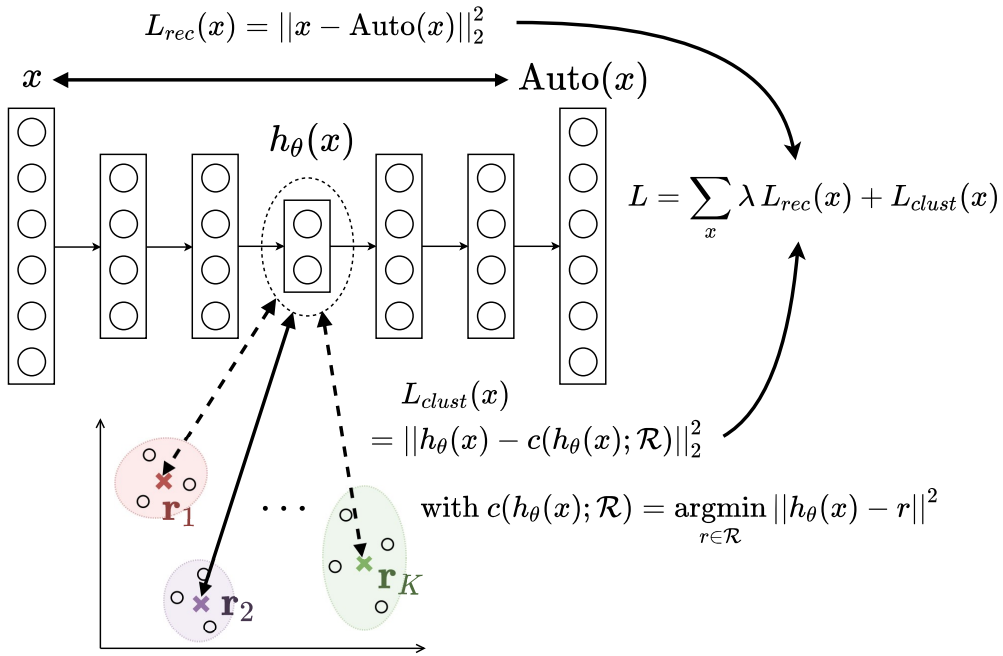
# Mais quelle représentation choisir ?

## Subjectivité de la solution



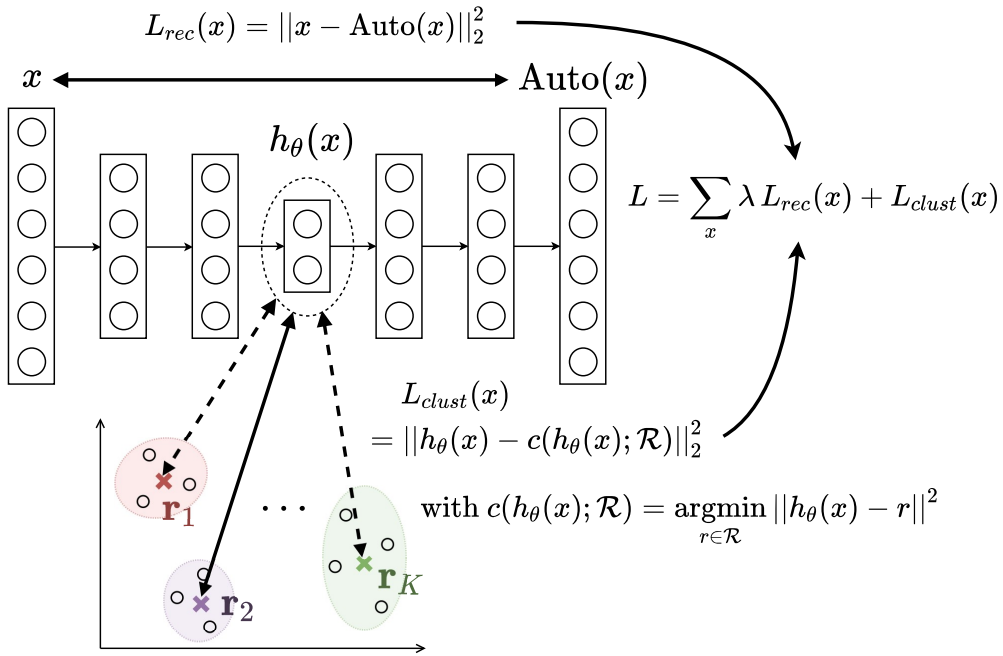
# Apprentissage de représentation et clustering

- Plusieurs façons de projeter des documents dans des espaces latents (factorisation, *latent probabilistic models*, ...)
- Solution élégante offerte pour les autoencodeurs (texte, images, séries temporelles, ...)



# Apprentissage de représentation et clustering

- Plusieurs façons de projeter des documents dans des espaces latents (factorisation, *latent probabilistic models*, ...)
- Solution élégante offerte pour les autoencodeurs (texte, images, séries temporelles, ...)



# Fonction objectif

Optimisation jointe sur une représentation plus adéquate

Combinaison des coûts de représentation et des $k$-moyennes :

$$P_{\text{DKM}}: \min_{\mathcal{R},\theta} \sum_{x \in \mathcal{X}} \underbrace{||x - \text{Auto}(x;\theta)||^2}_{L_{\text{rec}}(x)} + \lambda \underbrace{||\mathbf{h}_\theta(x) - c(\mathbf{h}_\theta(x);\mathcal{R})||^2}_{L_{\text{clust}}(x)}$$

avec $c(\mathbf{h}_\theta(x);\mathcal{R}) = \arg\min_{\mathbf{r} \in \mathcal{R}} ||\mathbf{h}_\theta(x) - \mathbf{r}||^2$ (plus proche représ. de $\mathbf{h}_\theta(x)$)

Regularisation ? Code "simple" sous TensorFlow ; optimisation par SGD

*Deep k-Means : Jointly clustering with k-Means and learning representations*. Moradi Fard, Thonet, Gaussier (arXiv :1806.10069)

# Table des matières

# Conclusion - Apprentissage et RI

## Ordonnancement

- Des approches qui tentent d'exploiter toutes les informations à disposition (60 attributs pour la collection *gov* par exemple, y compris les scores des modèles *ad hoc*) et qui visent directement à ordonner les documents (*pairwise*, *listwise*)
- Beaucoup de propositions (réseaux neuronaux (Bing), *boosting*, méthodes à ensemble)
- Comptent parmi les méthodes les plus performantes à l'heure actuelle

## Apprentissage profond

- Devenu extrêmement populaire – plongements de mots ou de documents (B. Piwowarski), sessions de recherches (Sordoni *et al.*)

# Quelques Références (1)

Burges et al. *Learning to Rank with Nonsmooth Cost Functions*, NIPS 2006

Cao et al. *Learning to Rank : From Pairwise to Listwise Approach*, ICML 2007

Joachims et al. *Accurately Interpreting Clickthrough Data as Implicit Feedback*, SIGIR 2005

Liu *Learning to Rank for Information Retrieval*, 2009.

Mitra and Craswell *An* Introduction to Neural Information Retrieval, 2018

# Quelques Références (2)

Nallapati *Discriminative model for Information Retrieval*, SIGIR 2004

Sordoni et al. *A hierarchical recurrent encoder-decoder for generative context-aware query suggestion*, CIKM 2015

Yue et al. *A Support Vector Method for Optimizing Average Precision*, SIGIR 2007

Workshop LR4IR, 2007 (Learning to Rank for Information Retrieval).