Apprentissage non-supervisé

Boi Faltings

Laboratoire d'Intelligence Artificielle
boi.faltings@epfl.ch
http://moodle.epfl.ch/

Apprentissage non-supervisé

- Souvent on a beaucoup d'exemples, mais il manque une classification $\Rightarrow$ impossible d'appliquer l'apprentissage supervisé.
- Apprentissage non-supervisé: remplace le critère d'un modèle *correct* par un critère de performance plus général:
  - apprentissage de sous-classes (clustering) d'un jeu d'exemples: critère d'homogénéité.
  - apprentissage d'un modèle mixte (mixture models): degré d'approximation des exemples.
- Apprentissage semi-supervisé: utiliser les données supplémentaires pour améliorer la performance d'un apprentissage supervisé.

Applications de l'apprentissage non-supervisé

Analyse de données et découverte de nouvelles connaissances:

- site web: trouver des classes d'utilisateurs similaires pour leur fournir des chemins d'accès spécifiques.
- bioinformatique: classer des segments d'ADN similaires pour reconnaître la structure.
- donner des recommandations de produits dans le commerce électronique.

Apprentissage de sous-classes

- Un ensemble d'exemples peut être groupé en sous-classes naturelles:

  A: grand,allongé,rouge
  B: grand,rond,rouge
  C: petit,rond,vert
  D: petit,rond,rouge

  $\Rightarrow$ on groupe {A,B} et {C,D}, mais pas {A,C} et {B,D}.
- "Naturel" dépend d'un critère, par exemple la taille de la description de la classe.

Clustering

- Apprendre une sous-division implique une *recherche* parmi des alternatives possibles pour sélectionner celle qui couvre les exemples de la meilleure manière.
- "Clustering": regrouper les exemples en *aggrégats* de façon à ce que chaque cluster contienne des exemples similaires.
- Base: mesure de distance $d(\underline{x}_1, \underline{x}_2)$ (ou similarité) entre exemples.

n exemples
$\Rightarrow$ k^n manières de les diviser en k aggrégats
$\Rightarrow$ trop complexe pour une recherche exhaustive

Etapes du clustering

1. Représentation des exemples
2. Mesure de similarité: définition de la proximité (distance) de deux exemples suivant le domaine d'application (par ex. la distance euclidienne)
3. Regroupement d'exemples (clustering à proprement dit)

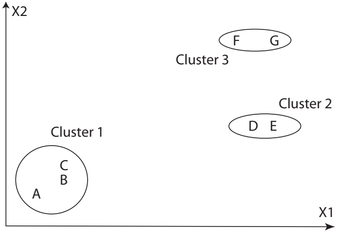
Algorithmes de clustering

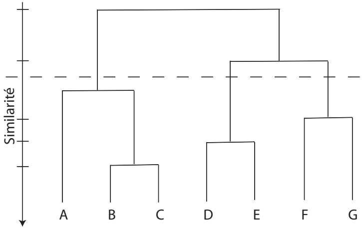
- Le *clustering hiérarchique* crée une structure de regroupement de clusters à différents niveaux:
  - single-link
  - complete-link
- Le *clustering de partitionnement* obtient une partition unique des exemples (au lieu d'une structure hiérarchique):
  - k-means clustering
  - clustering sur la base de la théorie des graphes
- Le *clustering flou* définit une probabilité d'appartenance à un cluster.

Clustering hiérarchique (1)

- Crée un *dendrogramme* qui représente les regroupements d'exemples et des niveaux de similarité.
- Le dendrogramme est coupé à différents niveaux, donnant lieu à différents aggrégats.

Clustering hiérarchique (2)



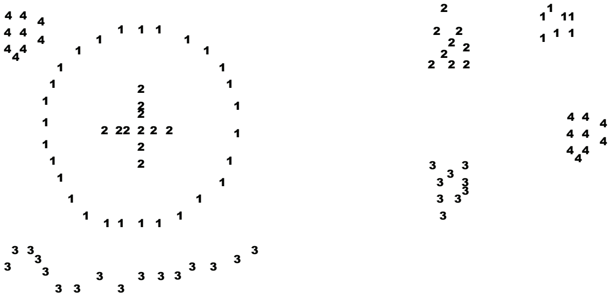


Méthode par agglomération:

Etapes:

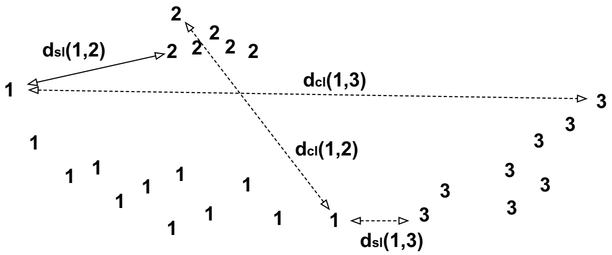
1. Placer chaque exemple dans son propre cluster (singleton).
2. Trouver la paire de clusters la plus similaire (suivant une définition de distance entre clusters) et la fusionner en un seul cluster.
3. Calculer la distance entre ce nouveau cluster et tous les autres clusters.
4. Répéter 2 et 3 jusqu'à ce qu'il n'existe qu'un seul cluster contenant tous les exemples.

Mesures de similarité



- à gauche: similarité transitive
- à droite: similarité non-transitive

Mesures de similarité entre clusters

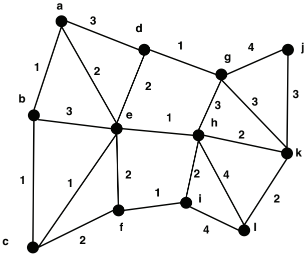


- *single-link*: le minimum des distances entre toutes les paires d'exemples de ces deux clusters. Implique une similarité *transitive*: connexion de 1 et 3.
- *complete-link*: le maximum des distances. Implique une similarité *non-transitive*: connexion de 1 et 2.

Clustering hiérarchique par division

Considérer un graphe de similarité:

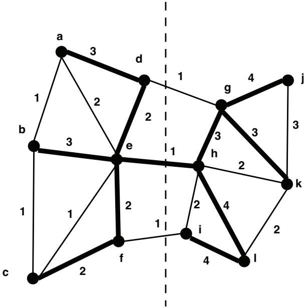
- noeuds = instances
- arcs = paires de noeuds dont la similarité dépasse un certain seuil (suffisant pour que le graphe soit connecté).
- poids d'un arc = similarité entre instances.



Division

- Principe: découper le graphe en composantes connexes.
- couper les arcs dont la similarité est la plus faible.
- clusters = composantes connexes du graphe.
- implique une similarité *transitive*.

Utilisant l'arbre couvrant maximal



- construire un arbre couvrant à poids maximal (minimum spanning tree avec poids inversés, MST).
- couper l'arc de poids minimal pour séparer une composante à la fois.
- correspond au critère single link: mesure de distance transitive.

Faiblesses

- Ne considère pas l'ensemble des arcs qui séparent les composantes: sensible à des erreurs induites par quelques instances.
- Ne permet pas d'équilibrer la taille des composantes et donc de la hierarchie.

Souvent mieux: utiliser des coupes à poids minimal.

Coupes à poids minimal

- Algorithme exact pour minimiser la somme des poids: quadratique en nombre des arcs, complexité prohibitive.
- Approximation: coupe spectrale par le vecteur de Fiedler
- Minimise le critère NCut:

$$\left(\sum_{a \in cut} w(a)\right) \cdot \left(\frac{1}{\sum_{a \in C_1} w(a)} + \frac{1}{\sum_{a \in C_2} w(a)}\right)$$

$\Rightarrow$ équilibrage de la taille des composantes en même temps.

Coupe spectrale

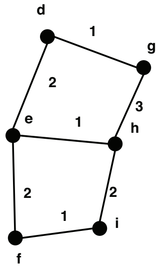
- Matrice $W =$ poids des arcs entre instances.
- Matrice diagonale $D =$ donne la somme des poids des arcs incidents à chaque instance.
- Construire la *matrice Laplacien*:

$$L = I - D^{-1}W$$

- Si graphe connexe, la plus petite valeur propre $e_1 = 0$.
- Le vecteur propre associé à la *deuxième* plus petite valeur propre est le vecteur de Fiedler:

  composantes positives pour instances dans C_1,
  négatives pour instances dans C_2.

Exemple (sous-graphe d-i)



W =

	d	e	f	g	h	i
d	5	2	0	1	0	0
e	2	5	2	0	1	0
f	0	2	5	0	0	1
g	1	0	0	5	3	0
h	0	1	0	3	5	2
i	0	0	1	0	2	5

D =

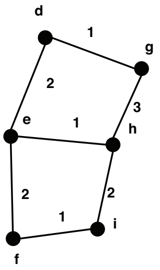
	d	e	f	g	h	i
d	8	0	0	0	0	0
e	0	10	0	0	0	0
f	0	0	8	0	0	0
g	0	0	0	9	0	0
h	0	0	0	0	11	0
i	0	0	0	0	0	8

⇒ L =

	d	e	f	g	h	i
d	1-5/8	-2/8	0	-1/8	0	0
e	-0.2	1-5/10	-0.2	0	-0.1	0
f	0	-2/8	1-5/8	0	0	-1/8
g	-1/9	0	0	1-5/9	-3/9	0
h	0	-1/11	0	-3/11	1-5/11	-2/11
i	0	0	-1/8	0	-2/8	1-5/8

Exemple (2)

valeurs propres:
-0.02384543627974877 $\approx$ 0
0.9002625341080476
0.4880620765867691
0.6825500054465383
0.2565102964256015
0.1914605237127924 3 *



- vecteur propre (0.19):

$$[-0.2397, -0.4451, -0.6003, 0.5383, 0.3066, 0.0087]$$

- $\Rightarrow$ regrouper (d,e,f) et (g,h,i)

Clustering de partitionnement: k-means

Initialisation: choisir k *noyaux*:

$\underline{c}_j \leftarrow$ exemple qui est au coeur d'un aggrégat C_j.

Méthode itérative:

- associer chaque exemple $\underline{x}_i$ à l'aggrégat dont le noyau est le plus proche, c'est-à-dire C_j tel que $d(\underline{c}_j, \underline{x}_i)$ est minimal.
- pour chaque aggrégat C_j, remplacer le noyau par l'exemple qui est le plus au centre des exemples de l'aggrégat, c'est-à-dire $\underline{x_c} \in C_j$ tel que

$$\sum_{\underline{x}_i \in C_j} d(\underline{x}_c, \underline{x}_i)^2$$

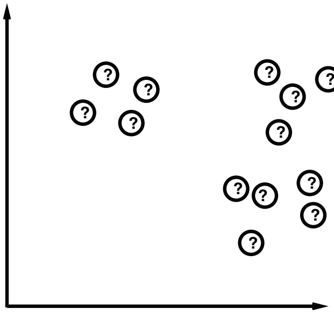
  est minimal.

Faiblesse: résultat dépend du choix initial des noyaux!

Exemple (k-means)

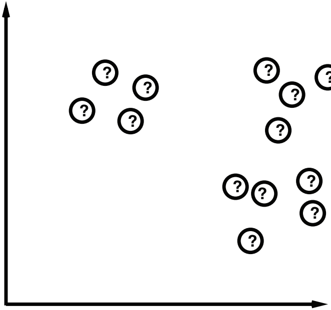
Situation initiale:

exemples non-classés:

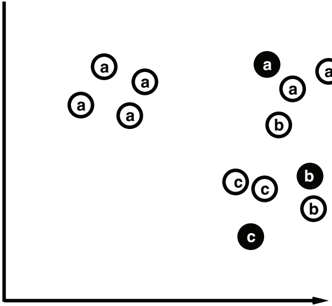


Exemple (k-means)

Situation initiale:
exemples non-classés:

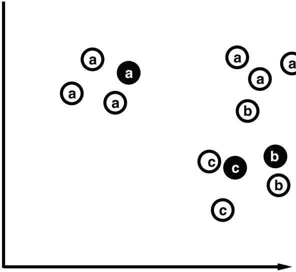


Initialisation: choisir 3 noyaux
et classer les exemples:

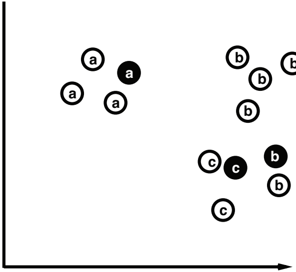


Exemple (2)

Recalculer les noyaux:

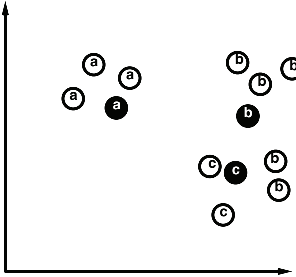


classer les exemples à nouveau:

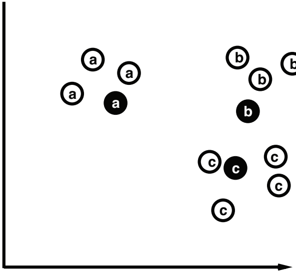


Exemple (3)

Recalculer les noyaux:



classer les exemples:



⇒ aggrégats raisonnables!

Propriétés du k-means

- Repéter l'aggrégation jusqu'à ce qu'il y ait un regroupement stable.
- Aucune garantie de convergence!
- Difficile de choisir le bon k.
- Mesure de distance pour des exemples formalisés par des attributs logiques: nombre d'attributs différents.
- Implique une mesure de similarité *non-transitive*.

Complexité

- n objets, m attributs
- clustering hiérarchique: au moins $O(n^2 * m)$
- clustering k-means: $O(l * k * n * m)$, $l =$ nombre d'itérations

$\Rightarrow$ si bonne convergence, k-means peut être beaucoup plus efficace.

Versions

- k-means proprement dit: les noyaux sont les *moyennes* des coordonnées des exemples.

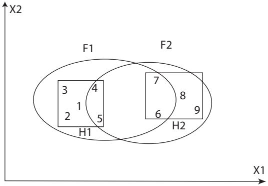
$\Rightarrow$ ne s'applique qu'aux attributs continus.

- La version que nous avons vu s'appelle *k-médoïdes*, mais est souvent appellé k-means.
- Plus générale car elle admet tout type d'attribut.

Clustering probabiliste

- Les méthodes traditionnelles de clustering génèrent des partitions où chaque exemple n'appartient à une et une seule partition à la fois.
- Le *clustering flou* permet à un exemple d'appartenir à plusieurs clusters à la fois.
- Appartenance à un cluster défini par une distribution de probabilité.

Exemple de clustering probabiliste



Deux clusters probabilistes F1 et F2 sont créés. A chaque exemple, on associe une probabilité d'appartenance:

- F1 = (1,0.9), (2,0.8), (3,0.7), (4,0.6), (5,0.55), (6,0.2), (7,0.2), (8,0.0), (9,0.0)
- F2 = (1,0.0), (2,0.0), (3,0.0), (4,0.1), (5,0.15), (6,0.4), (7,0.35), (8,1.0), (9,0.9)

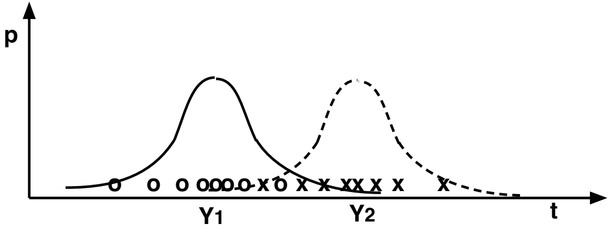
Modèle génératif

- Critère de succès: le modèle doit *générer* les exemples observés.
- Supposons que les exemples sont des clusters avec chacun sa propre distribution.
- Pour générer un exemple:
  - d'abord on choisit le cluster de l'exemple
  - ensuite on génère ses attributs selon la distribution associée au cluster.

$\Rightarrow$ modèle statistique très général.

Mélange de Gaussiennes

Supposons que les exemples sont generés par un un mélange de k processus à distribution Gaussienne:



2 distributions:

- "o" centrées autour de Y_1.
- "x" centrées autour de Y_2.

mais on ne sait pas quels sont les "o" et les "x".

Modèle

- Moyenne Y_j = instance prototypique
- Variance σ_j
- Probabilité de générer l'instance X_i:

$$P(X_i|j) = \frac{1}{\sigma_j\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{d(X_i,Y_j)}{\sigma_j}\right)^2}$$

avec $d(X, Y)$ = distance entre instances X et Y

Exemple: chaque type j de visiteur d'un site web a un prototype Y_j avec des variations aléatoires.

Estimer les paramètres

- Nous cherchons les paramètres Y_j et σ_j de chaque cluster, et en plus la probabilité d'appartenance $p(z_i = j)$
- ...de façon à maximiser les probabilités des instances:

$$p(X_i) = \sum_{j=1}^{k} P(X_i|j)p(z_i = j)$$

- Supposons qu'on connaît pour chaque instance i le cluster z_i auquel elle appartient.

$\Rightarrow$ Pour les Gaussiennes, on maximise la probabilité des instances par Y_j = moyenne des X_i avec $z_i = j$.

- ...mais comment savoir les z_i?

Algorithme EM (Expectation Maximization)

Initialiser aléatoirement pour k clusters Y_j^0, σ_j^0, $p^0(z_i = j)$.
Itération:

- Espérance: pour chaque exemple et cluster, estimer la probabilité $L^t(i,j)$ d'appartenance de l'instance i au cluster j:

$$L^t(i,j) = \frac{p^t(z_i = j)g(X_i, \sigma_j^t, Y_j^t)}{\sum_{l=1}^k p^t(z_i = l)g(X_i, \sigma_l^t, Y_l^t)}$$

  $g(X, \sigma, Y) =$ fonction de distribution Gaussienne centrée à Y avec variance σ, appliquée à X.

- Maximisation: ...

Algorithme EM (2)

- Maximisation: pour chaque cluster, recalculer le centre et la variance de la distribution pour maximiser la probabilité des exemples:

$$Y_j^{t+1} = \frac{\sum_{i=1}^{n} L^t(i,j) X_i}{\sum_{i=1}^{n} L^t(i,j)}$$

et

$$\sigma_j^{t+1} = \frac{\sum_{i=1}^{n} L(i,j) d^2(X_i, Y_j)}{\sum_{i=1}^{n} L(i,j)}$$

et la probabilité d'appartenance:

$$p_j^{t+1} = \frac{1}{n} \sum_{i=1}^{n} L^t(i,j)$$

Algorithme EM (3)

- Itération des pas E et M jusqu'à ce que les changements sont suffisement petits (comme k-means).
- Les clusters pourraient suivre d'autres distributions que des Gaussiennes.
- Convergence pas garantie mais non-convergence est rare.
- Beaucoup utilisé (non seulement pour le clustering).

Généralisation du clustering spectral

Pour obtenir une coupe en $k > 2$ composantes:

- trier les valeurs propres dans l'ordre croissant:

$$e_1 \leq e_2 \leq e_3 \leq \ldots$$

- considérer les vecteurs propres associées aux valeurs 2 à $k + 1$ comme système de coordonnées.
- construire un clustering de partitionnement dans cet espace (par k-means ou EM).

Application

Modélisation d'une population mixte, par exemple

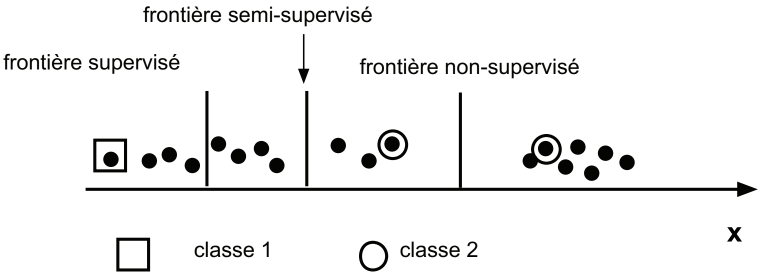
- les programmes TV regardés par les membres d'une famille, où un programme peut être aimé par plusieurs membres.
- les mots qui caractérisent un sujet, où chaque mot peut appartenir à plusieurs sujets.

Apprentissage semi-supervisé

Souvent, beaucoup de données mais que peu d'exemples classifiés.

⇒ pour l'apprentissage supervisé, ajuster les frontières pour correspondre à la distribution des instances.

⇒ pour l'apprentissage non-supervisé, utiliser les classifications pour identifier les clusters distincts.



Application: Recommandation de produits

Personnes similaires achètent des produits similaires:

1. $\Rightarrow$ recommander produits achetés par personnes similaires
   personne similaire $\Leftrightarrow$ achète les mêmes produits
   Amazon: "clients qui ont achetés x ont aussi achetés y"
2. $\Rightarrow$ regrouper des produits par leur clientèle
   Google: actualités personalisées

Calcul sur demande trop coûteux: utiliser un clustering de produits, d'articles de presse, et d'utilisateurs pour réduire la complexité.
Le clustering est appris sur les données d'achat.

Résumé

- L'apprentissage non-supervisé n'a pas besoin d'exemples classifiés préalablement.
- Clustering: déterminer une sous-classification efficace d'un ensemble d'exemples.
  - hiérarchique, par agglomeration
  - spectral, par arbre couvrant ou min-cut
  - k-means: partitionnement
  - EM: clustering probabiliste