

Pense-bête VIP : Apprentissage non-supervisé

Afshine AMIDI et Shervine AMIDI

6 octobre 2018

Introduction à l'apprentissage non-supervisé

□ **Motivation** – Le but de l'apprentissage non-supervisé est de trouver des formes cachées dans un jeu de données non-labelées $\{x^{(1)}, \dots, x^{(m)}\}$.

□ **Inégalité de Jensen** – Soit f une fonction convexe et X une variable aléatoire. On a l'inégalité suivante :

$$E[f(X)] \geq f(E[X])$$

Espérance-Maximisation

□ **Variables latentes** – Les variables latentes sont des variables cachées/non-observées qui posent des difficultés aux problèmes d'estimation, et sont souvent notées z . Voici les cadres dans lesquelles les variables latentes sont le plus fréquemment utilisées :

Cadre	Variance latente z	$x z$	Commentaires
Mixture de k gaussiennes	Multinomial(ϕ)	$\mathcal{N}(\mu_j, \Sigma_j)$	$\mu_j \in \mathbb{R}^n, \phi \in \mathbb{R}^k$
Analyse factorielle	$\mathcal{N}(0, I)$	$\mathcal{N}(\mu + \Lambda z, \psi)$	$\mu_j \in \mathbb{R}^n$

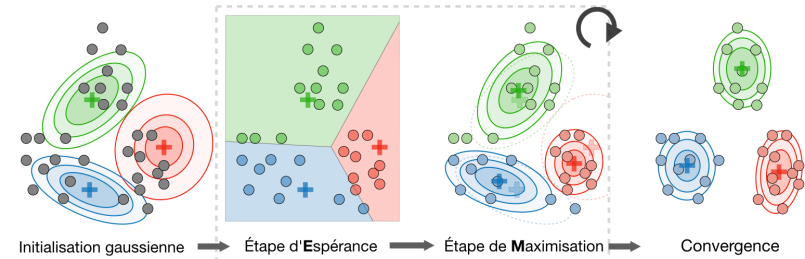
□ **Algorithme** – L'algorithme d'espérance-maximisation (EM) est une méthode efficace pour estimer le paramètre θ . Elle passe par le maximum de vraisemblance en construisant une borne inférieure sur la vraisemblance (E-step) et optimisant cette borne inférieure (M-step) de manière successive :

- E-step : Évaluer la probabilité postérieure $Q_i(z^{(i)})$ que chaque point $x^{(i)}$ provienne d'une partition particulière $z^{(i)}$ de la manière suivante :

$$Q_i(z^{(i)}) = P(z^{(i)} | x^{(i)}; \theta)$$

- M-step : Utiliser les probabilités postérieures $Q_i(z^{(i)})$ en tant que coefficients propres aux partitions sur les points $x^{(i)}$ pour ré-estimer séparément chaque modèle de partition de la manière suivante :

$$\theta_i = \operatorname{argmax}_{\theta} \sum_i \int_{z^{(i)}} Q_i(z^{(i)}) \log \left(\frac{P(x^{(i)}, z^{(i)}; \theta)}{Q_i(z^{(i)})} \right) dz^{(i)}$$



Partitionnement k -means

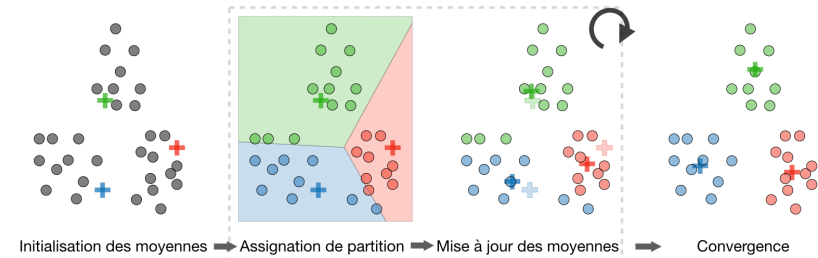
On note $c^{(i)}$ la partition du point i et μ_j le centre de la partition j .

□ **Algorithme** – Après avoir aléatoirement initialisé les centroïdes de partitions $\mu_1, \mu_2, \dots, \mu_k \in \mathbb{R}^n$, l'algorithme k -means répète l'étape suivante jusqu'à convergence :

$$c^{(i)} = \operatorname{argmin}_j \|x^{(i)} - \mu_j\|^2$$

et

$$\mu_j = \frac{\sum_{i=1}^m 1_{\{c^{(i)}=j\}} x^{(i)}}{\sum_{i=1}^m 1_{\{c^{(i)}=j\}}}$$



□ **Fonction de distortion** – Pour voir si l'algorithme converge, on regarde la fonction de distortion définie de la manière suivante :

$$J(c, \mu) = \sum_{i=1}^m \|x^{(i)} - \mu_{c^{(i)}}\|^2$$

Regroupement hiérarchique

□ **Algorithme** – C'est un algorithme de partitionnement avec une approche hiérarchique qui construit des partitions intriquées de manière successive.

□ **Types** – Il y a différents types d'algorithme de regroupement hiérarchique qui ont pour but d'optimiser différents fonctions objectif, récapitulés dans le tableau ci-dessous :

Ward linkage	Average linkage	Complete linkage
Minimize within cluster distance	Minimize average distance between cluster pairs	Minimize maximum distance of between cluster pairs

Indicateurs d'évaluation de clustering

Dans le cadre de l'apprentissage non-supervisé, il est souvent difficile d'évaluer la performance d'un modèle vu que les vrais labels ne sont pas connus (contrairement à l'apprentissage supervisé).

□ **Coefficient silhouette** – En notant a et b la distance moyenne entre un échantillon et tous les autres points d'une même classe, et entre un échantillon et tous les autres points de la prochaine partition la plus proche, le coefficient silhouette s d'un échantillon donné est défini de la manière suivante :

$$s = \frac{b - a}{\max(a, b)}$$

□ **Index de Calinski-Harabaz** – En notant k le nombre de partitions, B_k et W_k les matrices de dispersion entre-partitions et au sein d'une même partition sont définis respectivement par :

$$B_k = \sum_{j=1}^k n_{c(i)} (\mu_{c(i)} - \mu)(\mu_{c(i)} - \mu)^T, \quad W_k = \sum_{i=1}^m (x^{(i)} - \mu_{c(i)})(x^{(i)} - \mu_{c(i)})^T$$

l'index de Calinski-Harabaz $s(k)$ renseigne sur la qualité des partitions, de sorte à ce qu'un score plus élevé indique des partitions plus denses et mieux séparées entre elles. Il est défini par :

$$s(k) = \frac{\text{Tr}(B_k)}{\text{Tr}(W_k)} \times \frac{N - k}{k - 1}$$

Analyse des composantes principales

C'est une technique de réduction de dimension qui trouve les directions maximisant la variance, vers lesquelles les données sont projetées.

□ **Valeur propre, vecteur propre** – Soit une matrice $A \in \mathbb{R}^{n \times n}$, λ est dit être une valeur propre de A s'il existe un vecteur $z \in \mathbb{R}^n \setminus \{0\}$, appelé vecteur propre, tel que l'on a :

$$Az = \lambda z$$

□ **Théorème spectral** – Soit $A \in \mathbb{R}^{n \times n}$. Si A est symétrique, alors A est diagonalisable par une matrice réelle orthogonale $U \in \mathbb{R}^{n \times n}$. En notant $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$, on a :

$$\exists \Lambda \text{ diagonal, } A = U\Lambda U^T$$

Remarque : le vecteur propre associé à la plus grande valeur propre est appelé le vecteur propre principal de la matrice A .

□ **Algorithme** – La procédure d'analyse des composantes principales (en anglais *PCA - Principal Component Analysis*) est une technique de réduction de dimension qui projette les données sur k dimensions en maximisant la variance des données de la manière suivante :

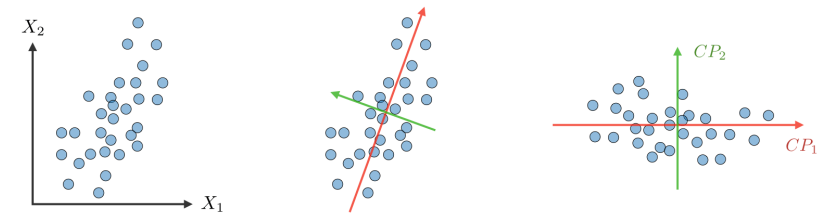
— Étape 1 : Normaliser les données pour avoir une moyenne de 0 et un écart-type de 1.

$$x_j^{(i)} \leftarrow \frac{x_j^{(i)} - \mu_j}{\sigma_j} \quad \text{où} \quad \mu_j = \frac{1}{m} \sum_{i=1}^m x_j^{(i)} \quad \text{et} \quad \sigma_j^2 = \frac{1}{m} \sum_{i=1}^m (x_j^{(i)} - \mu_j)^2$$

— Étape 2 : Calculer $\Sigma = \frac{1}{m} \sum_{i=1}^m x^{(i)} x^{(i)T} \in \mathbb{R}^{n \times n}$, qui est symétrique et aux valeurs propres réelles.

— Étape 3 : Calculer $u_1, \dots, u_k \in \mathbb{R}^n$ les k valeurs propres principales orthogonales de Σ , i.e. les vecteurs propres orthogonaux des k valeurs propres les plus grandes.

— Étape 4 : Projeter les données sur $\text{span}_{\mathbb{R}}(u_1, \dots, u_k)$. Cette procédure maximise la variance sur tous les espaces à k dimensions.



Données dans l'espace initial ➡ Trouve les composantes principales ➡ Données dans l'espace des CP

Analyse en composantes indépendantes

C'est une technique qui vise à trouver les sources génératrices sous-jacentes.

□ **Hypothèses** – On suppose que nos données x ont été générées par un vecteur source à n dimensions $s = (s_1, \dots, s_n)$, où les s_i sont des variables aléatoires indépendantes, par le biais d'une matrice de mélange et inversible A de la manière suivante :

$$x = As$$

Le but est de trouver la matrice de démixage $W = A^{-1}$.

□ **Algorithme d'ICA de Bell and Sejnowski** – Cet algorithme trouve la matrice de démixage W en suivant les étapes ci-dessous :

— Écrire la probabilité de $x = As = W^{-1}s$ par :

$$p(x) = \prod_{i=1}^n p_s(w_i^T x) \cdot |W|$$

— Écrire la log vraisemblance de notre jeu de données d'entraînement $\{x^{(i)}, i \in [1, m]\}$ et en notant g la fonction sigmoïde par :

$$l(W) = \sum_{i=1}^m \left(\sum_{j=1}^n \log \left(g'(w_j^T x^{(i)}) \right) + \log |W| \right)$$

Par conséquent, l'algorithme du gradient stochastique est tel que pour chaque exemple du jeu d'entraînement $x^{(i)}$, on met à jour W de la manière suivante :

$$W \leftarrow W + \alpha \left(\begin{pmatrix} 1 - 2g(w_1^T x^{(i)}) \\ 1 - 2g(w_2^T x^{(i)}) \\ \vdots \\ 1 - 2g(w_n^T x^{(i)}) \end{pmatrix} x^{(i)T} + (W^T)^{-1} \right)$$