UNIVERSITÉ LUMIÈRE LYON 2
MASTER STATISTIQUE ET INFORMATIQUE POUR LA SCIENCE DES DONNÉES (SISE)

ClustR

Package R pour le Clustering de Variables

Riad SAHRANE, Aya MECHERI, Thibaud LECOMTE
11/12/2025

Introduction

Clustering de Variables : Principes Généraux

Aspect	Observations	Variables
Espace	$\mathbb{R}^p$	$\mathbb{R}^n$
Distance typique	Euclidienne	Corrélation
Objectif	Grouper individus	Grouper features
Application	Segmentation	Réduction dim.

Distance entre Variables

Pour des variables numériques, la distance la plus courante est basée sur la corrélation : $d(X_j, X_k) = 1 - |\rho(X_j, X_k)|^2$

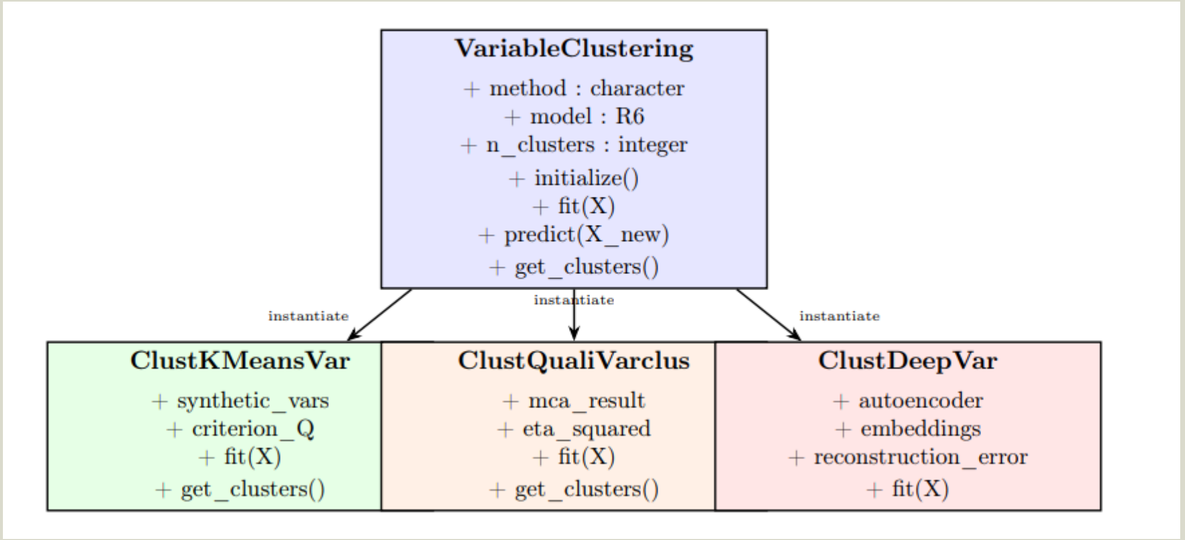
Pour des variables catégorielles, on utilise plutôt :

- **Chi-carré** : $\chi^2(V_1, V_2)$
- **Cramér's V** : $V_C = \sqrt{\frac{\chi^2}{n \cdot \min(r-1, c-1)}}$
- **Rapport de corrélation** : $\eta^2(V, Y) = \frac{\mathrm{Var}(\mathbb{E}[Y|V])}{\mathrm{Var}(Y)}$

Philosophie & Approche Globale

Philosophie de conception (inspirée de scikit-learn)

- *Interface unifiée via R6.*
- *Tous les modèles : fit(), predict(), get_clusters().*
- *Modularité / extensibilité.*
- *Reproductibilité + structure claire.*



1

linéaire
(K-means réallocatif)
ClustKMeansVar

2

qualitatif
(MCA + η^2)
ClustQualiVarclus

3

non linéaire
(deep autoencoder)
ClustDeepVar

K-means Réallocatif

Principe Vigneau & Qannari (2003)

- *centre = PC1 (synthetic variable)*
- *distance = 1 − corr²*

Ce que nous avons développé “from scratch” :

- Implémentation complète de la boucle réallocative.
- Calcul automatique de Q (B/T).
- Standardisation (optionnelle).
- Initialisation aléatoire.
- Gestion de la convergence.

- Synthetic variable = PC1
- Critère Q
- Réallocation selon corr²

Formalisation Mathématique Fit

Étape 1 Standardisation (optionnelle):
Chaque variable est centrée-réduite avant l'apprentissage

$$Z_j = \frac{X_j - \mu_j}{\sigma_j}$$

Étape 2 Initialisation de la partition :
Répartition aléatoire des variables dans K clusters

$$\mathcal{C}^{(0)} = \{C_1^{(0)}, \ldots, C_K^{(0)}\}$$

Étape 3 Calcul du centroïde d'un cluster (PC1) :
Le centroïde du cluster est la première composante principale

$$\mu_k = \mathrm{PC1}(Z_{C_k})$$

Étape 4 Fonction de distance : $1-r^2$:
On veut mesurer la proximité d'une variable Zj avec le centroïde μk

$$d(j,k) = 1 - \mathrm{cor}(Z_j, \mu_k)^2$$

Cette distance est équivalente à mesurer le manque de variance expliquée par le cluster car μk est définie comme le vecteur unitaire maximisant la variance projetée

Formalisation Mathématique

Étape 5 Réallocation des variables :
Chaque variable est affectée au cluster qui maximise r^2_{jk}

$$j \in C_k^{(\text{new})} \quad \Longleftrightarrow \quad k = \arg \max_{\ell \in \{1,\ldots,K\}} r^2_{j\ell}$$

Étape 6 Critère d'optimisation : inertie intra et inter :
Inertie intra-cluster (W), Inertie totale (T), Inertie inter (B) et le Critère de qualité de partition (Q)

$$W = \sum (1 - r^2), \quad B = T - W, \quad Q = B/T$$

Étape 7 Boucle K-means réallocatif :

Répéter jusqu'à convergence :

recalcul des PC1 (centroïdes) ⟶ réallocation des variables (maximisation de r^2) ⟶ vérification

Si non ⟵ $C^{(t+1)} = C^{(t)}$

Formalisation Mathématique Predict

Étape 1 Standardisation (optionnelle):
On standardise avec les paramètres du jeu d'apprentissage

$$Z_{\text{new}} = \frac{X_{\text{new}} - \mu_{\text{train}}}{\sigma_{\text{train}}}$$

Étape 2 Utilisation des centroïdes appris (PC1) :
Les μk sont fixés aucune mise à jour en prédiction

$$\mu_k = \text{PC1}(Z_{C_k})$$

Étape 3 Calcul des corrélations entre la nouvelle variable et chaque PC1 :
Pour chaque cluster k, on calcule

$$r_k = \text{cor}(Z_{\text{new}}, \mu_k)$$

Étape 4 Calcul des distances :

$$d_k = 1 - r_k^2$$

Étape 3 Attribution au cluster qui maximise la corrélation :

$$\widehat{k} = \arg\max_{1 \le k \le K} r_k^2$$

Quali VARCLUS

Ce que nous avons récupéré :

- *Idée générale de VARCLUS (SAS)*
- *MCA via FactoMineR comme moteur*

Ce que nous avons développé :

- Réallocation via η^2
- Gestion des clusters multiples
- Système générique compatible avec l'interface R6

- MCA synthétise un cluster.
- Rapport η^2 measure la qualité d'appartenance.
- Algorithme similaire au k-means mais adapté au qualitatif.

Quali VARCLUS

Étape 1 : Initialisation aléatoire des clusters :

Choix du nombre de clusters

Affecter chaque variable qualitative à un cluster choisi aléatoirement (répartition équilibrée).

Étape 2 : MCA par Cluster

MCA par Cluster Pour un cluster de variables, on réalise une MCA sur les colonnes associées.

Extraction de l'axe 1

Étape 3 : Rapport de Corrélation η2

Calculer η² entre la variable et chaque axe 1 du cluster.

Réaffecter chaque variable dans le cluster maximisant son η².

$$\eta^2(V, Y) = \frac{\mathrm{Var}(\mathbb{E}[Y \mid V])}{\mathrm{Var}(Y)} \in [0, 1]$$

Étape 4 : Gérer les clusters vides

Déplacer la variable la moins bien représentée dans le cluster vide.

Étape 5 : Tester la convergence

Deep Clustering

Autoencodeurs pour le Clustering

Plusieurs approches existent :

— DEC (Xie et al., 2016) : Deep Embedded Clustering

— DCEC (Guo et al., 2017) : Deep Convolutional Embedded Clustering

— VaDE (Jiang et al., 2017) : Variational Deep Embedding

Principe général :

1. Pré-entraînement de l'autoencodeur (reconstruction)
2. Extraction des embeddings latents
3. Clustering dans l'espace latent (k-means, GMM)

Application aux Variables :

Innovation de ClustR :

transposer la matrice de données pour traiter les variables comme des observations.

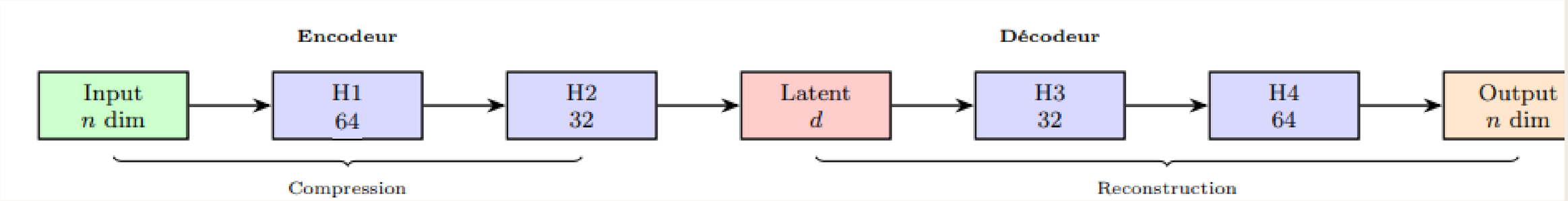
$$X \in \mathbb{R}^{n \times p} \quad \Rightarrow \quad X^T \in \mathbb{R}^{p \times n}$$

Chaque variable devient une "observation" de dimension n (les valeurs sur les individus).

Algorithme Deep Clustering

*ClustDeepVar utilise **un autoencodeur** profond pour **apprendre** des embeddings **non-linéaires** des variables, puis cluster ces embeddings avec k-means.*

***Objectif :** clustering de variables en apprenant pour chacune un embedding latent non-linéaire via un autoencodeur entraîné sur XT .*



Formalisation Mathématique

Étape 1 : Standardisation
Chaque variable est centrée-réduite avant l'apprentissage :

$$X_{\text{std}} = \frac{X - \mu}{\sigma}$$

Étape 2 : Transposition
On entraîne l'autoencodeur sur XT : chaque variable devient une "observation" composée de n valeurs

$$X^T \in \mathbb{R}^{p \times n}$$

Étape 3 : Encodeur Embeddings
L'encodeur réduit chaque variable à un vecteur latent de dimension d = latent_dim :

$$Z = f_{\text{enc}}(X^T), \quad Z \in \mathbb{R}^{p \times d}$$

Étape 4 : Décodeur — Reconstruction
L'autoencodeur reconstruit chaque variable à partir de son embedding :

$$\hat{X}^T = f_{\text{dec}}(Z)$$

Formalisation Mathématique

Étape 5 : Fonction de Perte
L'autoencodeur minimise l'erreur de reconstruction avec régularisation L2 :

$$\min_{\theta} \frac{1}{p} \sum_{j=1}^{p} \mathcal{L}(x_j, \hat{x}_j) + \lambda \|\theta\|_2^2$$

Étape 6 : Régularisation et Robustesse
Techniques de régularisation pour éviter le surapprentissage(biais/variance) :

Dropout : désactivation aléatoire de neurones
Régularisation L2 : pénalité sur les poids
Early Stopping : arrêt automatique de l'entraînement

Étape 7 : Clustering des Embeddings
Les embeddings des variables sont clusterisés via k-means :

$$\mathcal{C} = \text{k-means}(Z, k)$$

Étape 8 : Soft-Clustering (probabilités)
Les distances aux centres sont converties en probabilités via un softmax :

$$P_{jk} = \frac{\exp(-d_{jk}/T)}{\sum_{c=1}^{k} \exp(-d_{jc}/T)}$$

Étape 9 : Projection de Variables Illustratives
Une variable illustrative v (numérique ou factorielle) est projetée dans l'espace latent via :

$$z_{\text{illu}} = \frac{\sum_{j=1}^{p} \text{cor}(x_j, v)\, z_j}{\left\| \sum_{j=1}^{p} \text{cor}(x_j, v)\, z_j \right\|}$$

Algorithm 4 ClustDeepVar — Deep Variable Clustering

Require: $X \in \mathbb{R}^{n \times p}$, k clusters, d latent_dim, hyperparamètres
Ensure: Clusters $\mathcal{C}$, embeddings Z

Étape 1 : Standardisation
$X \leftarrow (X - \mu)/\sigma$
Étape 2 : Transposition
$X^T \leftarrow X^T \in \mathbb{R}^{p \times n}$
Étape 3 : Construction de l'autoencodeur
Encodeur : Input(n) $\rightarrow$ Hidden_layers $\rightarrow$ Latent(d)
Décodeur : Latent(d) $\rightarrow$ Hidden_layers (inversé) $\rightarrow$ Output(n)
Étape 4 : Entraînement
for epoch = 1 to epochs **do**
    $Z \leftarrow f_{\text{enc}}(X^T)$
    $\hat{X}^T \leftarrow f_{\text{dec}}(Z)$
    loss $\leftarrow \text{MSE}(X^T, \hat{X}^T) + \lambda \|\theta\|_2^2$
    Backpropagation & mise à jour des poids
    if early stopping criterion **then**
        break
    end if
end for
Étape 5 : Extraction des embeddings
$Z \leftarrow f_{\text{enc}}(X^T)$
Étape 6 : Clustering k-means
$\mathcal{C} \leftarrow \text{k-means}(Z, k)$
Étape 7 : Soft-clustering (optionnel)
Calculer P_{jk} via softmax des distances
return $\mathcal{C}$, Z, P

Avantages Spécifiques

Relations non-linéaires : capture des structures complexes inaccessibles aux méthodes linéaires

Embeddings robustes : goulot d’étranglement + régularisation (L2, dropout, early stopping)

Soft-clustering : probabilités d’appartenance via softmax (analyse d’incertitude)

Projection illustratives : intégration de variables externes pour interprétation

Prédiction : possibilité de projeter de nouvelles variables dans l’espace latent

Scalabilité : adapté aux gros datasets (p > 200) avec architecture profonde

Métriques de Validation

Inertie Intra-Cluster (W)

$$W = \sum_{g=1}^{k} \sum_{j \in C_g} d(X_j, y_g)$$

Mesure la compacité des clusters. Plus W est faible, meilleurs sont les clusters.

Inertie Inter-Clusters (B)

$$B = \sum_{g=1}^{k} |C_g| \cdot d(\bar{y}_g, \bar{y})$$

Mesure la séparation entre clusters. Plus B est élevé, meilleure est la séparation.

Critère Q (K-means uniquement)

$$Q = \frac{B}{T}, \quad T = W + B$$

$Q \in [0, 1]$, optimal proche de 1.

Métriques de Validation

Silhouette

Pour une variable j dans le cluster C_g :

$$a_j = \text{distance moyenne intra-cluster}$$
$$b_j = \text{distance moyenne au cluster le plus proche}$$
$$s_j = \frac{b_j - a_j}{\max(a_j, b_j)}$$

Silhouette moyenne :

$$\bar{s} = \frac{1}{p}\sum_{j=1}^{p} s_j \in [-1, 1]$$

Interprétation :

— $s_j \approx 1$: bien clusterisé
— $s_j \approx 0$: entre deux clusters
— $s_j < 0$: mal clusterisé

Reconstruction Error (Deep uniquement)

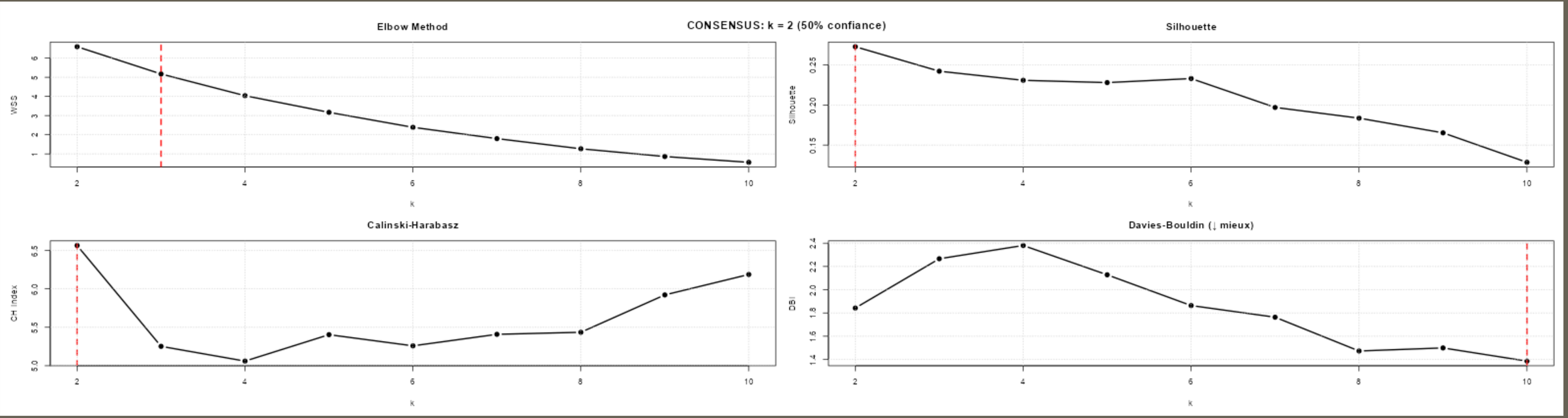
$$\text{MSE} = \frac{1}{p \cdot n}\sum_{j=1}^{p}\sum_{i=1}^{n}(x_{ij} - \hat{x}_{ij})^2$$

Plus l'erreur est faible, meilleur est l'autoencodeur.

Sélection Automatique du Nombre de Clusters

Auto-k par consensus (Elbow + Silhouette + CH + DBI)

“Notre package ne se contente pas de clusteriser :
il propose automatiquement le meilleur k avec un score de confiance.”



Comparaison des Algorithmes

Critère	K-means	Quali	Deep
Type de données	Numériques	Catégorielles	Numériques
Relations	Linéaires	χ^2	Non-linéaires
Interprétabilité	Haute (PC1)	Moyenne (MCA)	Faible (boîte noire)
Complexité	$O(np^2/k)$	$O(np^2)$	$O(epochs \cdot p \cdot d^2)$
Scalabilité	Moyenne	Faible	Haute
Hyperparamètres	k, standardize	k, auto_k	k, d, epochs, dropout
Soft-clustering	Non	Non	Oui
Projection	Non	Oui	Oui

Limitations :

K-means : sensibilité à l'initialisation, relations linéaires seulement
Quali : ne gère que les variables catégorielles, complexité
Deep : temps d'entraînement long, hyperparamètres nombreux, interprétabilité limitée

Merci !