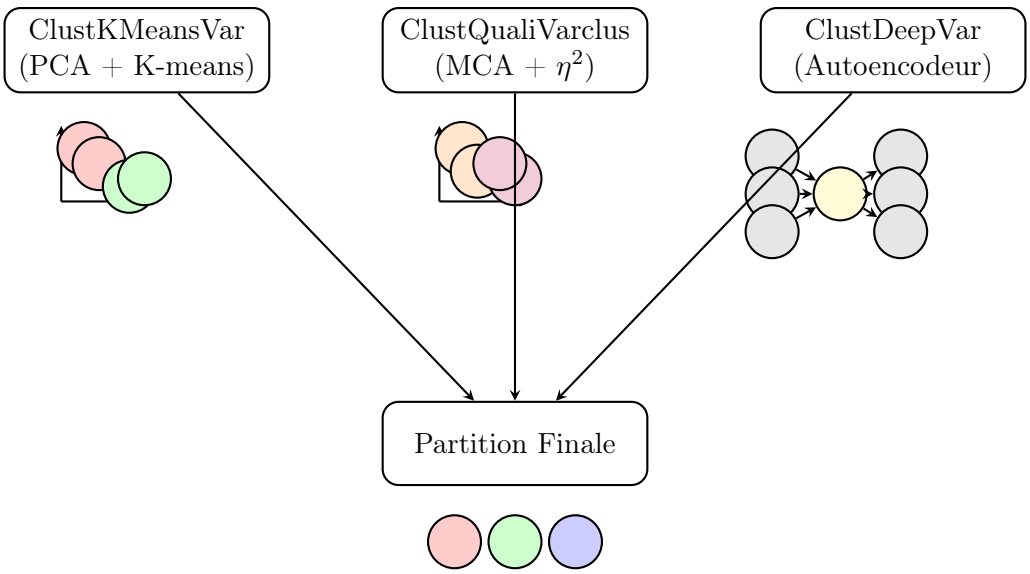
**Université Lumière Lyon 2**
Master Statistique et Informatique pour la Science des Données (SISE)

# ClustR

**Package R pour le Clustering de Variables**

Trois Approches Complémentaires : K-means, Qualitatif et Deep Learning




**Auteurs :**

Riad SAHRANE
Aya MECHERI
Thibaud LECOMTE

**Encadrant :** Ricco Rakotomalala

Année Universitaire 2024–2025
30 novembre 2025

**Résumé**

Ce rapport présente **ClustR**, un package R innovant dédié au **clustering de variables**, dont l'objectif est d'identifier des groupes homogènes de variables afin de faciliter l'interprétation, réduire la dimensionnalité et améliorer les analyses statistiques et prédictives. Contrairement au clustering classique des observations, le clustering de variables s'intéresse aux relations entre *features*, nécessitant des outils adaptés capables de capturer aussi bien les dépendances linéaires que non linéaires, et de traiter des données mixtes (numériques et qualitatives).

ClustR propose une **interface unifiée basée sur R6**, inspirée de la philosophie de *scikit-learn*, permettant à l'utilisateur de sélectionner et comparer facilement plusieurs algorithmes complémentaires. Le package intègre également une application **Shiny** pour une exploration interactive, rendant la méthodologie accessible aussi bien aux utilisateurs avancés qu'aux non-programmeurs.

Les trois algorithmes principaux implémentés sont :

— **ClustKMeansVar** : une méthode réallocative fondée sur les composantes principales, inspirée des travaux de Vigneau & Qannari. Cette approche permet de résumer chaque cluster par une variable synthétique (PC1) et maximise un critère global de séparation ($Q = B/T$).

— **ClustQualiVarclus** : une méthode dédiée aux variables qualitatives, reposant sur l'Analyse des Correspondances Multiples (MCA) et l'utilisation du rapport de corrélation $\eta^2$. Elle permet de regrouper des modalités présentant des structures similaires dans l'espace factoriel.

— **ClustDeepVar** : un algorithme de **deep clustering** utilisant un autoencodeur pour projeter chaque variable dans un espace latent non linéaire. Cette approche capture des relations complexes entre variables et propose un regroupement fondé sur les embeddings appris.

L'ensemble constitue un outil cohérent permettant de traiter des jeux de données réels variés, incluant des variables corrélées, catégorielles ou possédant des relations non linéaires. Des tests sur des jeux de données simulés et réels démontrent la robustesse et la complémentarité des trois méthodes.

**Mots-clés :** Clustering de variables, R6, Autoencodeur, MCA, PCA, K-means réallocatif, Analyse factorielle, Deep Learning, Shiny, Package R.

# Table des matières

# 0.1 Introduction

## 0.1.1 Contexte et Motivation

Le clustering de variables est une technique d'analyse multivariée visant à regrouper des variables **similaires** en clusters homogènes, contrairement au clustering d'observations classique. Cette approche est particulièrement pertinente dans les contextes suivants :

- **Réduction de dimensionnalité** : identifier des groupes de variables redondantes pour simplifier l'analyse
- **Sélection de variables** : choisir un représentant par cluster pour éviter la multicolinéarité
- **Interprétation** : comprendre la structure latente des données via des groupes cohérents
- **Feature engineering** : créer des variables synthétiques par cluster

Malgré l'importance de cette tâche, les solutions existantes en R présentent plusieurs limitations :

- Manque d'interface unifiée (packages dispersés)
- Absence de méthodes deep learning
- Incompatibilité entre variables numériques et catégorielles
- Pas de sélection automatique du nombre de clusters

## 0.1.2 Objectifs du Projet

Ce projet vise à développer **ClustR**, un package R moderne offrant :

1. Une **interface unifiée** de type scikit-learn (classe R6)
2. **Trois algorithmes complémentaires** :
   - K-means réallocatif (Vigneau & Qannari, 2003)
   - VARCLUS qualitatif (MCA + $\eta^2$)
   - Deep clustering par autoencodeur
3. Des **métriques de validation** robustes (inertie, silhouette, stabilité)
4. Une **sélection automatique du nombre de clusters** via méthodes consensus
5. Une **documentation complète** avec vignettes et exemples

## 0.1.3 Contributions

Les contributions principales de ce travail sont :

- **Première implémentation R** d'un autoencodeur pour le clustering de variables
- **Interface unifiée** permettant de comparer facilement les trois méthodes
- **Régularisation avancée** dans ClustDeepVar (dropout, L2, early stopping)
- **Projection de variables illustratives** dans l'espace latent
- **Sélection automatique de k** via consensus de 4 métriques

## 0.2 État de l'Art

### 0.2.1 Clustering de Variables : Principes Généraux

Le clustering de variables diffère fondamentalement du clustering d'observations :

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

TABLE 1 – Comparaison clustering d'observations vs. variables

#### Distance entre Variables

Pour des variables numériques, la distance la plus courante est basée sur la corrélation :

$$d(X_j, X_k) = 1 - |\rho(X_j, X_k)|^2 \tag{1}$$

où $\rho(X_j, X_k)$ est le coefficient de corrélation de Pearson.
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** : $\eta^2(V, Y) = \frac{\mathrm{Var}(\mathbb{E}[Y|V])}{\mathrm{Var}(Y)}$

### 0.2.2 Méthodes Existantes

#### ClustOfVar (Chavent et al., 2012)

Package R de référence utilisant :
— Classification ascendante hiérarchique (CAH)
— Distance basée sur $1 - \rho^2$
— Critère de Ward pour l'agrégation
— Support mixte (numér. + catég.) via FAMD
**Limitations** :
— Pas de soft-clustering
— Complexité $O(p^2 \log p)$
— Pas de projection de nouvelles variables

#### VARCLUS (SAS)

Algorithme propriétaire SAS :
— Divisive clustering (top-down)
— Basé sur PCA oblique
— Critère $1 - R^2$ ratio

**Vigneau & Qannari (2003)**

Méthode K-means adaptée aux variables :
— Centre de cluster = PC1 (synthetic variable)
— Distance = $1 - \text{cor}(X_j, y_g)^2$
— Critère global = $Q = B/T$

### 0.2.3 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)
4. (Optionnel) Fine-tuning joint clustering + reconstruction

**Application aux Variables**

**Innovation de ClustR** : transposer la matrice de données pour traiter les variables comme des observations.

$$X \in \mathbb{R}^{n \times p} \quad \Rightarrow \quad X^T \in \mathbb{R}^{p \times n} \tag{2}$$

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

### 0.2.4 Positionnement de ClustR

| **Package** | **Méthodes** | **Interface** | **Deep** |
|---|---|---|---|
| ClustOfVar | Hiérarchique, mixte | Classique R | Non |
| VARCLUS (SAS) | Divisive, PCA oblique | Propriétaire | Non |
| **ClustR** | K-means, Quali, Deep | R6 + Auto-k | **Oui** |

TABLE 2 – Comparaison des packages de clustering de variables

## 0.3 Architecture du Package ClustR

### 0.3.1 Philosophie de Conception

ClustR adopte une architecture orientée objet basée sur le système **R6** (classes mutables), inspirée de la philosophie de `scikit-learn` en Python :

- **Interface unifiée** : tous les algorithmes partagent les mêmes méthodes (`fit`, `predict`, `get_clusters`)
- **Composition** : la classe `VariableClustering` encapsule les algorithmes spécifiques
- **Extensibilité** : ajout facile de nouveaux algorithmes
- **Reproductibilité** : paramètres sauvegardés automatiquement

### 0.3.2 Diagramme de Classes

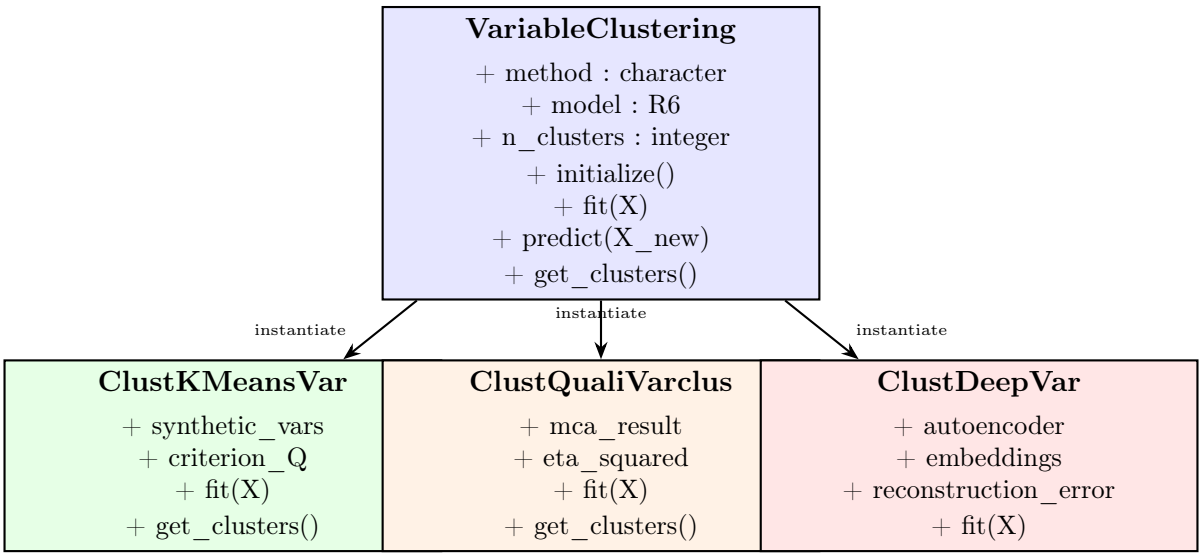



FIGURE 1 – Diagramme de classes UML simplifié de ClustR

### 0.3.3 Workflow Général

**Algorithm 1** Utilisation typique de ClustR

1: **Charger** les données $X \in \mathbb{R}^{n \times p}$
2: **Choisir** la méthode : "kmeans", "quali_varclus" ou "deep"
3: **Initialiser** le modèle avec paramètres
4: **Entraîner** sur les données
5: **Extraire** les clusters
6: **Valider** via métriques (inertie, silhouette, etc.)
7: **Visualiser** les résultats
8: **Exporter** les résultats

## 0.4 Algorithme K-means Réallocatif

### 0.4.1 Principe Général

La méthode de Vigneau & Qannari (2003) adapte l'algorithme K-means classique au clustering de variables en introduisant la notion de **synthetic variable** (variable synthétique) par cluster.

#### Intuition

Au lieu de calculer un centre de cluster comme la moyenne des observations (impossible pour des variables), on utilise la **première composante principale (PC1)** des variables du cluster comme représentant synthétique.

### 0.4.2 Formalisation Mathématique

#### Notations

— $X = [X_1, \ldots, X_p] \in \mathbb{R}^{n \times p}$ : matrice de données ($n$ observations, $p$ variables)
— $k$ : nombre de clusters
— $C_g$ : ensemble des indices de variables dans le cluster $g$ ($g = 1, \ldots, k$)
— $y_g \in \mathbb{R}^n$ : synthetic variable du cluster $g$ (PC1)

#### Étape 1 : Standardisation (optionnelle)

Les variables sont éventuellement centrées-réduites :

$$X_{\text{std}} = \frac{X - \mu}{\sigma} \tag{3}$$

**But :** rendre les variables comparables en échelle.

#### Étape 2 : Synthetic Variable

Pour un cluster $C_g$, la synthetic variable $y_g$ est définie comme la première composante principale de $X_{C_g}$ :

$$y_g = X_{C_g} w_g \tag{4}$$

où $w_g$ est le premier vecteur propre de la matrice de covariance $S_g = X_{C_g}^T X_{C_g}$ :

$$w_g = \text{argmax}_{\|w\|=1} \, w^T S_g w \tag{5}$$

**Interprétation :** $y_g$ résume au mieux les variables du cluster $g$.

#### Étape 3 : Distance Variable-Cluster

La distance entre une variable $X_j$ et un cluster $g$ est définie comme :

$$d(j, g) = 1 - \rho(X_j, y_g)^2 \tag{6}$$

où $\rho(X_j, y_g)$ est le coefficient de corrélation de Pearson :

$$\rho(X_j, y_g) = \frac{\mathrm{Cov}(X_j, y_g)}{\sigma_{X_j} \cdot \sigma_{y_g}} \tag{7}$$

**Règle :** $X_j$ est affectée au cluster avec la distance $d(j, g)$ minimale.

**Étape 4 : Critère Global Q**

L'algorithme maximise le critère $Q = B/T$ :
On décompose l'inertie totale $T$ en :
— Inertie intra-cluster : $W$
— Inertie inter-clusters : $B$

$$T = B + W \tag{8}$$

$$Q = \frac{B}{T} \in [0, 1] \tag{9}$$

**Interprétation :** plus $Q$ est proche de 1, plus les clusters sont bien séparés.

### 0.4.3 Algorithme Complet

**Algorithm 2** K-means Réallocatif (Vigneau & Qannari)

**Require:** Matrice $X \in \mathbb{R}^{n \times p}$, nombre de clusters $k$
**Ensure:** Partition $\mathcal{C} = \{C_1, \ldots, C_k\}$

**Étape 1 : Standardisation (optionnelle)**
$X \leftarrow (X - \mu)/\sigma$
**Étape 2 : Initialisation**
Partitionner les $p$ variables en $k$ clusters (aléatoire ou k-means)
**repeat**
  **Étape 3 : Calcul des synthetic variables**
  **for** $g = 1$ to $k$ **do**
    $y_g \leftarrow \mathrm{PC1}(X_{C_g})$
  **end for**
  **Étape 4 : Réallocation**
  **for** $j = 1$ to $p$ **do**
    Calculer $d(j, g)$ pour tout $g \in \{1, \ldots, k\}$
    Affecter $X_j$ au cluster $g^* = \mathrm{argmin}_g\, d(j, g)$
  **end for**
  **Étape 5 : Calcul du critère**
  Calculer $Q = B/T$
**until** Convergence de $Q$ ou max_iter
**return** Partition finale $\mathcal{C}$

### 0.4.4 Complexité Algorithmique

— **PCA par cluster** : $O(n \cdot |C_g|^2)$ par cluster
— **Calcul des distances** : $O(p \cdot k \cdot n)$
— **Complexité totale par itération** : $O(n \cdot p^2/k)$

— **Nombre d'itérations** : typiquement $< 20$

**Complexité globale** : $O(T \cdot n \cdot p^2/k)$ où $T$ est le nombre d'itérations.

### 0.4.5 Avantages et Limitations

**Avantages**

— **Interprétabilité** : les synthetic variables sont directement interprétables comme des axes principaux
— **Efficacité** : convergence rapide ($< 20$ itérations)
— **Robustesse** : fonctionne bien avec des variables fortement corrélées
— **Critère global** : $Q$ permet de comparer différentes partitions

**Limitations**

— **Initialisation** : résultats dépendants de la partition initiale (multiple runs recommandés)
— **Linéarité** : basé sur PCA, ne capture pas les relations non-linéaires
— **Variables numériques** : ne fonctionne que pour des variables continues
— **Nombre de clusters** : $k$ doit être fixé a priori

## 0.5 Algorithme Qualitatif VARCLUS

### 0.5.1 Principe Général

Le clustering de variables catégorielles repose sur l'**Analyse des Correspondances Multiples (MCA)** et le **rapport de corrélation** $\eta^2$.

### 0.5.2 Formalisation Mathématique

#### Étape 1 : Encodage Disjonctif Complet (implicite)

Chaque variable catégorielle est transformée en indicatrices (one-hot) :

$$X \Rightarrow Z \in \{0,1\}^{n \times m} \tag{10}$$

**But :** représenter chaque modalité comme variable binaire.
**Note technique :** L'encodage one-hot est réalisé **automatiquement** par FactoMineR : :MCA. Pas besoin de coder explicitement les indicatrices.

#### Étape 2 : MCA par Cluster

Pour un cluster de variables, on réalise une MCA sur les colonnes associées. L'axe principal de la MCA synthétise le cluster.

#### Étape 3 : Rapport de Corrélation $\eta^2$

Pour une variable qualitative $V$ et un axe factoriel $Y$ :

$$\eta^2(V,Y) = \frac{\mathrm{Var}(\mathbb{E}[Y \mid V])}{\mathrm{Var}(Y)} \in [0,1] \tag{11}$$

**Interprétation :** $\eta^2$ proche de $1 \Rightarrow V$ bien expliquée par l'axe du cluster.

#### Étape 4 : Distance entre Variables (implicite)

La distance entre deux variables qualitatives peut être dérivée de $\eta^2$ ou de Cramér's V :

$$d(V_1,V_2) = 1 - \eta^2(V_1,V_2) \quad \text{ou} \quad d(V_1,V_2) = 1 - V_C(V_1,V_2) \tag{12}$$

où Cramér's V est défini par :

$$V_C(V_1,V_2) = \sqrt{\frac{\chi^2}{n \cdot (k-1)}} \tag{13}$$

avec $k = \min(\text{nb modalités } V_1, \text{nb modalités } V_2)$.
**Note d'implémentation :** L'algorithme VARCLUS utilise $\eta^2$ directement via l'ACM. La distance n'est pas calculée explicitement (clustering indirect via axes factoriels). Cramér's V est mentionné comme alternative théorique mais non utilisé dans cette implémentation.

### 0.5.3 Algorithme Complet

**Algorithm 3** VARCLUS Qualitatif

**Require:** Variables qualitatives $X$, nombre de clusters $k$
**Ensure:** Partition $\mathcal{C}$

**Étape 1 : Initialisation**
Partition initiale en $k$ groupes (aléatoire ou basée sur Cramér's V)
**repeat**
    **Étape 2 : MCA par cluster**
    **for** $g = 1$ to $k$ **do**
        Calculer MCA sur $X_{C_g}$
        Extraire l'axe principal $Y_g$
    **end for**
    **Étape 3 : Réallocation**
    **for** chaque variable $V_j$ **do**
        Calculer $\eta^2(V_j, Y_g)$ pour tout $g$
        Affecter $V_j$ au cluster $g^* = \text{argmax}_g \, \eta^2(V_j, Y_g)$
    **end for**
    **Étape 4 : Vérification convergence**
    **if** Partition stable **then**
        **break**
    **end if**
**until** max_iter atteint
**return** Partition $\mathcal{C}$

### 0.5.4 Avantages

— Spécifique aux variables catégorielles
— Utilise la structure $\chi^2$ via MCA / Cramér's V
— Partitions interprétables via les axes factoriels
— Gère naturellement les variables qualitatives ordinales et nominales

## 0.6 Algorithme Deep Clustering

### 0.6.1 Principe Général

**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 $X^T$.

### 0.6.2 Architecture de l'Autoencodeur

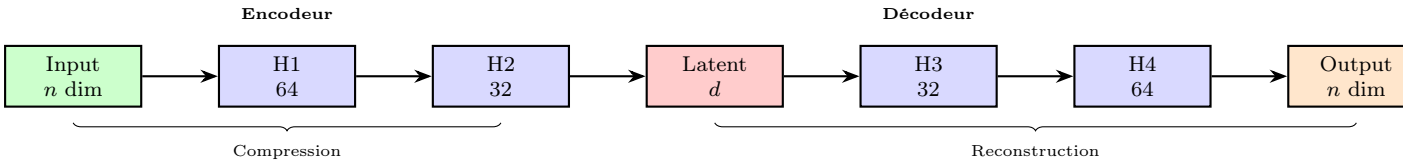



FIGURE 2 – Architecture de l'autoencodeur pour ClustDeepVar (exemple avec `hidden_layers = [64, 32]`)

### 0.6.3 Formalisation Mathématique

#### Étape 1 : Standardisation

Chaque variable est centrée-réduite avant l'apprentissage :

$$X_{\text{std}} = \frac{X - \mu}{\sigma} \tag{14}$$

**But :** rendre toutes les variables comparables.

#### Étape 2 : Transposition

On entraîne l'autoencodeur sur $X^T$ : chaque variable devient une "observation" composée de $n$ valeurs (les observations d'origine).

$$X^T \in \mathbb{R}^{p \times n} \tag{15}$$

**But :** apprendre une signature latente propre à chaque variable.

#### Étape 3 : Encodeur — Embeddings

L'encodeur réduit chaque variable à un vecteur latent de dimension $d = \text{latent\_dim}$ :

$$Z = f_{\text{enc}}(X^T), \quad Z \in \mathbb{R}^{p \times d} \tag{16}$$

**Interprétation :** chaque variable est représentée par un embedding latent compact.

**Étape 4 : Décodeur — Reconstruction**

L'autoencodeur reconstruit chaque variable à partir de son embedding :

$$\hat{X}^T = f_{\text{dec}}(Z) \tag{17}$$

**But :** forcer le réseau à capturer l'information essentielle de chaque variable.

**Étape 5 : Fonction de Perte**

L'autoencodeur minimise l'erreur de reconstruction avec régularisation L2 :

$$\min_{\theta} \frac{1}{p} \sum_{j=1}^{p} \mathcal{L}(x_j, \hat{x}_j) + \lambda \|\theta\|_2^2 \tag{18}$$

où :
— $\mathcal{L}$ = MSE (Mean Squared Error)
— $\lambda$ = coefficient de régularisation L2
— $\theta$ = ensemble des poids du réseau
**Objectif :** obtenir un espace latent stable, régularisé et robuste.

**Étape 6 : Régularisation et Robustesse**

Techniques de régularisation pour éviter le surapprentissage :
— **Dropout** : désactivation aléatoire de neurones
  — Taux standard : 0.05-0.10
  — Datasets complexes : $> 0.20$
  — Appliqué après chaque couche cachée
— **Régularisation L2** : pénalité $\lambda \|\theta\|_2^2$ sur les poids
  — Standard ($p \leq 100$) : $\lambda = 0.001$
  — Gros datasets ($p > 200$) : $\lambda = 0.01 - 0.02$
  — Réduit la complexité du modèle
— **Early Stopping** : arrêt automatique de l'entraînement
  — Patience = 10 epochs
  — Monitore `val_loss` sur ensemble de validation
  — Restaure les meilleurs poids
**But :** trade-off optimal biais/variance et prévention de l'overfitting.

**Étape 7 : Clustering des Embeddings**

Les embeddings des variables sont clusterisés via k-means :

$$\mathcal{C} = \text{k-means}(Z, k) \tag{19}$$

**Interprétation :** des variables proches dans l'espace latent sont regroupées.

**Étape 8 : Soft-Clustering (probabilités)**

Les distances aux centres sont converties en probabilités via un softmax :

$$P_{jk} = \frac{\exp(-d_{jk}/T)}{\sum_{c=1}^{k} \exp(-d_{jc}/T)} \tag{20}$$

où :
— $d_{jk} = \|z_j - \mu_k\|_2$ : distance euclidienne entre embedding $z_j$ et centre du cluster $k$
— $T$ = température (contrôle la "dureté" du clustering)
**But :** obtenir une appartenance floue des variables aux clusters.

**Étape 9 : Projection de Variables Illustratives**

Une variable illustrative $v$ (numérique ou factorielle) est projetée dans l'espace latent via :

$$z_{\text{illu}} = \frac{\sum_{j=1}^{p} \text{cor}(x_j, v)\, z_j}{\left\| \sum_{j=1}^{p} \text{cor}(x_j, v)\, z_j \right\|} \tag{21}$$

**But :** situer graphiquement une variable illustrative parmi les clusters sans influencer le clustering.

### 0.6.4 Algorithme Complet

**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($n$) → Hidden_layers → Latent($d$)
Décodeur : Latent($d$) → Hidden_layers (inversé) → 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** $\mathcal{C}$, $Z$, $P$

### 0.6.5 Avantages Spécifiques

— **Relations non-linéaires** : capture des structures complexes inaccessibles aux méthodes linéaires (PCA, corrélation)
— **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

## 0.7 Métriques de Validation et Sélection de k

### 0.7.1 Métriques Internes

**Inertie Intra-Cluster (W)**

$$W = \sum_{g=1}^{k} \sum_{j \in C_g} d(X_j, y_g) \tag{22}$$

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

**Inertie Inter-Clusters (B)**

$$B = \sum_{g=1}^{k} |C_g| \cdot d(\bar{y}_g, \bar{y}) \tag{23}$$

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

**Critère Q (K-means uniquement)**

$$Q = \frac{B}{T}, \quad T = W + B \tag{24}$$

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

**Silhouette**

Pour une variable $j$ dans le cluster $C_g$ :

$$a_j = \text{distance moyenne intra-cluster} \tag{25}$$

$$b_j = \text{distance moyenne au cluster le plus proche} \tag{26}$$

$$s_j = \frac{b_j - a_j}{\max(a_j, b_j)} \tag{27}$$

Silhouette moyenne :

$$\bar{s} = \frac{1}{p} \sum_{j=1}^{p} s_j \in [-1, 1] \tag{28}$$

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

**Reconstruction Error (Deep uniquement)**

$$\text{MSE} = \frac{1}{p \cdot n} \sum_{j=1}^{p} \sum_{i=1}^{n} (x_{ij} - \hat{x}_{ij})^2 \tag{29}$$

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

### 0.7.2 Sélection Automatique du Nombre de Clusters

ClustR implémente un système de sélection automatique de $k$ basé sur un **consensus de 4 métriques**.

#### Méthode 1 : Elbow (Coude)

**Principe :** Tracer l'inertie intra-cluster $W$ en fonction de $k$ et détecter le "coude" automatiquement.

**Détection automatique du coude :**

1. Calculer $W(k)$ pour $k \in [2, k_{\max}]$
2. Normaliser : $W_{\text{norm}}(k) = \frac{W(k) - \min W}{\max W - \min W}$
3. Calculer la dérivée seconde : $d^2W_{\text{norm}}/dk^2$
4. Coude = $\text{argmax}_k \, |d^2W_{\text{norm}}/dk^2|$

#### Méthode 2 : Silhouette Moyenne

**Principe :** Maximiser le coefficient de silhouette moyen.

$$k^* = \text{argmax}_{k \in [2, k_{\max}]} \, \bar{s}(k) \tag{30}$$

#### Méthode 3 : Calinski-Harabasz (CH)

**Principe :** Maximiser le ratio variance inter/intra.

$$\text{CH}(k) = \frac{\text{SSB}/(k-1)}{\text{SSW}/(n-k)} \tag{31}$$

où :
— SSB = sum of squares between clusters
— SSW = sum of squares within clusters

$$k^* = \text{argmax}_{k \in [2, k_{\max}]} \, \text{CH}(k) \tag{32}$$

#### Méthode 4 : Davies-Bouldin Index (DBI)

**Principe :** Minimiser l'indice de Davies-Bouldin.

$$\text{DBI}(k) = \frac{1}{k} \sum_{i=1}^{k} \max_{j \neq i} \left( \frac{\sigma_i + \sigma_j}{d(c_i, c_j)} \right) \tag{33}$$

où :
— $\sigma_i$ = dispersion intra-cluster $i$
— $d(c_i, c_j)$ = distance entre centres $c_i$ et $c_j$

$$k^* = \text{argmin}_{k \in [2, k_{\max}]} \, \text{DBI}(k) \tag{34}$$

**Consensus des 4 Métriques**

**Algorithm 5** Sélection Automatique de k par Consensus

**Require:** Dataset $X$, $k_{\max}$ (défaut = 10), algorithme (kmeans, quali_varclus, deep)
**Ensure:** $k^*_{\text{consensus}}$, confiance

**Étape 1 : Calcul des 4 métriques**
$k_{\text{elbow}} \leftarrow$ detect_elbow($X$, $k_{\max}$)
$k_{\text{sil}} \leftarrow \text{argmax}_k$ silhouette($X$, $k$)
$k_{\text{CH}} \leftarrow \text{argmax}_k$ Calinski-Harabasz($X$, $k$)
$k_{\text{DBI}} \leftarrow \text{argmin}_k$ Davies-Bouldin($X$, $k$)
**Étape 2 : Vote majoritaire**
votes $\leftarrow [k_{\text{elbow}}, k_{\text{sil}}, k_{\text{CH}}, k_{\text{DBI}}]$
$k^*_{\text{consensus}} \leftarrow$ mode(votes)
**Étape 3 : Calcul de la confiance**
confiance $\leftarrow \frac{\text{nb votes pour } k^*_{\text{consensus}}}{4} \times 100\%$
**return** $k^*_{\text{consensus}}$, confiance

**Exemple :**
— Elbow : $k = 3$
— Silhouette : $k = 3$
— Calinski-Harabasz : $k = 4$
— Davies-Bouldin : $k = 3$
$\Rightarrow$ Consensus : $k^* = 3$ (75% de confiance)

# 0.8 Conclusion

## 0.8.1 Résumé des Contributions

Ce projet a développé **ClustR**, un package R complet pour le clustering de variables, avec trois contributions majeures :

1. **Interface unifiée R6** : premier package offrant une API cohérente pour comparer K-means, qualitatif et deep learning
2. **Deep clustering de variables** : première implémentation R d'un autoencodeur pour le clustering de variables, avec :
   — Régularisation avancée (dropout, L2, early stopping)
   — Soft-clustering via softmax
   — Projection de variables illustratives
   — Prédiction de nouvelles variables
3. **Sélection automatique de k** : système de consensus basé sur 4 métriques (Elbow, Silhouette, CH, DBI)

## 0.8.2 Comparaison des Trois Algorithmes

| **Critère** | **K-means** | **Quali** | **Deep** |
|---|---|---|---|
| Type de données | Numériques | Catégorielles | Numériques |
| Relations | Linéaires | $\chi^2$ | 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 |

TABLE 3 – Comparaison synthétique des trois algorithmes

## 0.8.3 Limitations

— **K-means** : sensibilité à l'initialisation, relations linéaires seulement
— **Quali** : ne gère que les variables catégorielles, complexité $O(p^2)$
— **Deep** : temps d'entraînement long, hyperparamètres nombreux, interprétabilité limitée

# Bibliographie


[1] Vigneau, E., & Qannari, E. M. (2003). *Clustering of variables around latent components*. Communications in Statistics-Simulation and Computation, 32(4), 1131-1150.

[2] Chavent, M., Kuentz-Simonet, V., Labenne, A., & Saracco, J. (2012). *ClustOfVar : An R package for the clustering of variables*. Journal of Statistical Software, 50(13), 1-16.

[3] Xie, J., Girshick, R., & Farhadi, A. (2016). *Unsupervised deep embedding for clustering analysis*. In International conference on machine learning (pp. 478-487).

[4] Guo, X., Liu, X., Zhu, E., & Yin, J. (2017). *Deep clustering with convolutional autoencoders*. In International conference on neural information processing (pp. 373-382).

[5] Jiang, Z., Zheng, Y., Tan, H., Tang, B., & Zhou, H. (2017). *Variational deep embedding : An unsupervised and generative approach to clustering*. In Proceedings of the 26th International Joint Conference on Artificial Intelligence (pp. 1965-1972).

[6] Zou, H., Hastie, T., & Tibshirani, R. (2006). *Sparse principal component analysis*. Journal of computational and graphical statistics, 15(2), 265-286.

[7] Husson, F., Lê, S., & Pagès, J. (2010). *Exploratory multivariate analysis by example using R*. CRC press.

[8] Chollet, F., et al. (2015). *Keras*. https://keras.io.

[9] Rousseeuw, P. J. (1987). *Silhouettes : a graphical aid to the interpretation and validation of cluster analysis*. Journal of computational and applied mathematics, 20, 53-65.

[10] Caliński, T., & Harabasz, J. (1974). *A dendrite method for cluster analysis*. Communications in Statistics-theory and Methods, 3(1), 1-27.

[11] Davies, D. L., & Bouldin, D. W. (1979). *A cluster separation measure*. IEEE transactions on pattern analysis and machine intelligence, (2), 224-227.