mmrClustVar – Démo
Clustering de variables et package R6 en R

Marin Mazilda Rina

Master 2 SISE
Université Lumière Lyon 2

2025

Objectif de la démo

Présenter concrètement l'utilisation de la classe R6 `mmrClustVar`.
Montrer les quatre algorithmes :

`kmeans`, `kmodes`, `kprototypes`, `kmedoids`.

Illustrer le mode `method = "auto"`.
Utiliser la courbe d'inertie pour guider le choix de K.
Terminer par une démonstration de l'application Shiny.
Tout est scripté : si je fige, quelqu'un peut lire les slides et suivre les commandes.

Étape 1 – Installation depuis GitHub

Le package est installé une seule fois, directement depuis GitHub avec `devtools`. Cette étape n'est pas nécessaire si le package est déjà installé sur la machine.

Installation (une seule fois) :

```
# Installer devtools si besoin
# install.packages("devtools")

# Installer mmrClustVar depuis GitHub
devtools::install_github("rsquaredata/mmrClustVar")
```

Étape 2 – Chargement du package

Une fois installé, le package se charge comme n'importe quel package R.
La classe centrale s'appelle `mmrClustVarBase` et suit la logique R6 : `new()` → `$fit()` → `$summary()`, `$plot()`.

Charger le package :

```
library(mmrClustVar)
```

Étape 3 – Structure générale de la classe

Interface commune pour tous les algorithmes :

- **Création** : `Interface$new(method, K, ...)` ;
- **Apprentissage** : `$fit(X)` sur un data frame de variables actives ;
- **Résumé** : `$summary()` (texte) ;
- **Graphiques** : `$plot(type = ...)`.

Méthodes disponibles :

- "kmeans" : bloc numérique ;
- "kmodes" : bloc qualitatif ;
- "kprototypes" : bloc mixte ;
- "kmedoids" : dissimilarités générales ;
- "auto" : choix automatique en fonction des types de variables.

Démo 1 – k-means sur un jeu numérique

Exemple de départ : un jeu purement numérique `iris_num`.
Je force la méthode à "kmeans" avec $K = 3$ et `scale = TRUE`.
La méthode effectue des réallocations de variables : affectation → mise à jour de l'ACP locale → convergence.

Charger le jeu de données et lancer k-means :

```
data("iris_num", package = "mmrClustVar")
X_num <- iris_num

obj_km <- Interface$new(
  method = "kmeans",
  K      = 3,
  scale  = TRUE
)

obj_km$fit(X_num)
```

Démo 1 (suite) – Résumé et inertie

`$summary()` affiche :

- la méthode utilisée, K ;
- l'inertie intra-classe et la part d'inertie expliquée ;
- des indicateurs d'adhésion des variables aux clusters ;
- la composition des clusters de variables.

`$plot(type = "inertia")` trace la courbe d'inertie pour ce modèle.

Interpréter les résultats :

```
obj_km$summary()

# Courbe d'inertie du modle (pour K = 3)
obj_km$plot(type = "inertia")
```

Démo 2 – k-modes sur un jeu qualitatif

Je passe à un jeu purement catégoriel `arthritis_cat`.
L'algorithme "kmodes" est adapté aux variables qualitatives : prototypes = profils modaux, dissimilarité simple matching.

Charger le jeu et lancer k-modes :

```
data("arthritis_cat", package = "mmrClustVar")
X_cat <- arthritis_cat

obj_kmodes <- Interface$new(
  method = "kmodes",
  K      = 4
)

obj_kmodes$fit(X_cat)
obj_kmodes$summary()
```

Démo 2 (suite) – Adhésion catégorielle

Le graphique d'adhésion permet de voir quelles variables sont bien représentées par leur cluster.
Les barplots ou heatmaps mettent en évidence les blocs de variables qualitatives homogènes.

Visualiser l'adhésion des variables :

```
obj_kmodes$plot(type = "membership")
```

Démo 3 – k-prototypes sur metal_universe

Je passe à l'étude de cas metal_universe, un jeu mixte (numérique + catégoriel).
L'algorithme "kprototypes" combine :

- une composante principale locale pour la partie numérique ;
- un profil modal pour la partie catégorielle.

Je choisis ici $K = 5$ et lambda = 1.

Charger le jeu et lancer k-prototypes :

```
data("metal_universe", package = "mmrClustVar")
X_mix <- metal_universe

obj_kprot <- Interface$new(
  method = "kprototypes",
  K      = 5,
  scale  = TRUE,
  lambda = 1
)

obj_kprot$fit(X_mix)
```

Démo 3 (suite) – Blocs thématiques et adhésion

`$summary()` met en évidence des blocs thématiques :

- un bloc rythmique (BPM, vitesse, variabilité de tempo) ;
- un bloc agressivité / distorsion ;
- un bloc plus socio-géographique (pays, sous-genre, genre du/de la chanteur·se).

Les variables dérivées se regroupent en général avec leurs variables sources, tandis que les variables de bruit ont une adhésion plus faible.

Examiner le résumé et l'adhésion :

```
obj_kprot$summary()
obj_kprot$plot(type = "membership")
```

Démo 4 – k-medoids sur dissimilarités

"kmedoids" travaille à partir d'une matrice de dissimilarités entre variables.
Le prototype de chaque cluster est un **médoïde** : une vraie variable qui minimise la somme des distances intra-cluster.
C'est plus général, mais le calcul de la matrice de dissimilarités coûte $O(p^2)$.

Lancer k-medoids sur le même jeu mixte :

```
obj_kmed <- Interface$new(
  method = "kmedoids",
  K      = 4
)

obj_kmed$fit(X_mix)
obj_kmed$summary()
obj_kmed$plot(type = "inertia")
```

Démo 5 – Mode method = "auto"

Au lieu de choisir manuellement l'algorithme, on peut utiliser method = "auto", pratique pour un utilisateur non spécialiste en clustering de variables.

La classe détecte le type de variables :

- bloc numérique ⇒ k-means ;
- bloc qualitatif ⇒ k-modes ;
- bloc mixte ⇒ k-prototypes.

Exemples de sélection automatique :

```
# Cas numrique -> auto choisit k-means
obj_auto_num <- Interface$new(method = "auto", K = 3)
obj_auto_num$fit(X_num)
obj_auto_num$summary()

# Cas qualitatif -> auto choisit k-modes
obj_auto_cat <- Interface$new(method = "auto", K = 4)
obj_auto_cat$fit(X_cat)
obj_auto_cat$summary()

# Cas mixte -> auto choisit k-prototypes
obj_auto_mix <- Interface$new(method = "auto", K = 5, lambda = 1)
obj_auto_mix$fit(X_mix)
obj_auto_mix$summary()
```

Démo 6 – Courbe d'inertie pour le choix de K

Avant de fixer K, on peut explorer plusieurs valeurs et tracer la courbe d'inertie.
La fonction `compute_inertia_path()` calcule l'inertie intra-classe pour $K = 1, 2, \ldots, K_{\max}$.
On peut ensuite appliquer une heuristique de type « méthode du coude » pour choisir un K raisonnable.

Exemple sur le bloc numérique :

```
path_km <- compute_inertia_path(
  X      = X_num,
  method = "kmeans",
  K_max  = 8,
  scale  = TRUE
)

plot(path_km)
```

Démo 7 – Application Shiny `mmrClustVar_app`

L'application Shiny fournit une interface interactive au-dessus de la classe R6.
Elle permet :

- de choisir un jeu de données interne ou d'importer un fichier CSV/XLSX ;
- de sélectionner les variables actives et supplémentaires ;
- de choisir la méthode, K et les options (scale, lambda, etc.) ;
- d'afficher les résumés, les graphes d'inertie et d'adhésion, les cartes factorielles, etc.

C'est utile à la fois pour l'enseignement et pour l'exploration rapide.

Lancer l'application :

```
mmrClustVar_app()
```

Conclusion de la démo

La classe `mmrClustVar` offre une interface unifiée pour :

- quatre algorithmes de clustering de variables ;
- le choix automatique de la méthode ;
- des indicateurs d'inertie et d'adhésion, accessibles en texte et en graphiques.

L'application Shiny permet d'utiliser ces outils sans écrire de code, tout en restant cohérente avec la logique du package.

L'ensemble forme un module complet pour explorer, interpréter et enseigner le clustering de variables.

Merci pour votre attention !

Questions ?