

Projet mmrClustVar

Clustering de variables et package R6 en R

Marin Mazilda Rina

Master 2 SISE
Université Lumière Lyon 2

2025

Plan

- 1 Contexte du projet et cahier des charges
- 2 Méthodes et architecture
- 3 Étude de cas : metal_universe
- 4 Limites et perspectives

Contexte pédagogique et scientifique

- Projet du module **Programmation R** (M2 SISE, Université Lyon 2).
- Sujet général : **clustering de variables**, en s'inspirant notamment de ClustOfVar et des cours de classification / R.
- Objectif global du sujet :
 - ① créer un **package R** proposant plusieurs méthodes de clustering de variables ;
 - ② fournir des **indicateurs pour l'interprétation** des résultats (nature des partitions, groupes obtenus, degrés d'appartenance) ;
 - ③ livrer un package installable depuis GitHub, avec **documentation en anglais** et **application Shiny** de démonstration.

Cahier des charges : algorithmes (sujet)

Partie "algorithmes" du cahier des charges (extraits du sujet)

- Implémenter au moins trois algorithmes de clustering de variables.
- Parmi eux :
 - au moins un algorithme **basé sur les techniques de réallocation** ;
 - au moins un algorithme **adapté aux variables qualitatives**.
- Mettre en place une **stratégie d'identification du nombre de clusters K** :
 - au minimum des approches qui « aident » à choisir K (par exemple via des courbes d'inertie, des indicateurs d'adhésion).
- Intégrer des **outils d'interprétation** :
 - tableaux, graphiques ;
 - description de la nature des partitions, des groupes, et du degré d'appartenance des variables aux clusters.
- Arbitrer entre « ce qu'on réimplémente » et ce qu'on « délègue » à des **packages existants référencés** (ex. `hclust`) et **documenter ces choix**.

Classe de calcul (norme R6)

- Concevoir une classe R6 avec :
 - un **constructeur** paramétrable ;
 - la méthode `$fit(X)` pour ajuster le modèle sur un data frame de variables actives ;
 - la méthode `$predict(X)` pour rattacher des variables illustratives à des clusters ;
 - `$print()` pour un résumé court ; `$summary()` pour un résumé détaillé ;
 - des **propriétés exposées** (clusters, protos, inertie, etc.).

Application Shiny (obligatoire)

- L'application doit permettre :
 - de **sélectionner un jeu de données**, interne ou fichier utilisateur (CSV tabulé, éventuellement XLSX) ;
 - de choisir les **variables actives** et les **variables illustratives** ;
 - de lancer le clustering, d'afficher les **résultats et diagnostics** ;
 - « éventuellement d'autres fonctionnalités » à définir.

Algorithmes implémentés

- 4 méthodes (au lieu des 3 demandées) :
 - **k-means** de variables (numérique) ;
 - **k-modes** de variables (catégoriel) ;
 - **k-prototypes** de variables (mixte) ;
 - **k-medoids** de variables (dissimilarités).
- Les trois premiers sont explicitement des algorithmes à **réallocation itérative** (affectation → recalculation des prototypes → convergence).
- **k-modes** est dédié aux variables qualitatives : profil modal + simple matching.
- **k-medoids** procède aussi par réallocation, mais en choisissant des **médoïdes** (variables centrales) à partir d'une matrice de dissimilarités.

Stratégie pour K et interprétation

- Fonction `compute_inertia_path()` : courbe d'inertie (méthode du coude).
- Indicateurs d'**adhésion** et d'inertie expliquée accessibles via `$summary()` et `$plot()`.
- Outils d'interprétation : `$interpret_clusters()`, tableaux de clusters, barplots de membership, heatmaps, factor maps, dendrogrammes.

Architecture logicielle

- Classe de base `mmrClustVarBase` (R6) + 4 moteurs spécialisés.
- Façade haut niveau `mmrClustVar` :
 - `method = "auto"` pour choisir automatiquement entre k-means, k-modes, k-prototypes ;
 - interface homogène `$fit()`, `$predict()`, `$summary()`, `$plot()`.

Shiny et jeux de données

- Application `mmrClustVar_app` :
 - jeux internes variés (numérique, qualitatif, mixte) ;
 - **dataset thématique original** metal_universe pour un cas d'étude complet ;
 - import CSV/XLSX, choix des variables actives / supplémentaires ;
 - export des clusters, résumés et diagnostics (CSV / ZIP).
- Arbitrage packages existants :
 - algorithmes de réallocation **réimplémentés** en interne ;
 - usage ciblé de fonctions standards (ex. `hclust`, `stats`) pour certaines visualisations « de surface ».

Clustering de variables : principes

Perspective duale

- Les individus restent les lignes du tableau.
- Les variables sont représentées comme des vecteurs en dimension n (ou comme des profils de modalités pour les variables qualitatives).

Objectif

- Regrouper les variables en K clusters homogènes :
 - même structure de corrélation (variables numériques) ;
 - profils de modalités similaires (variables qualitatives) ;
 - cohérence mixte pour les données num + cat.
- Faciliter la réduction de dimension, l'interprétation et la compréhension des blocs thématiques.

Mesures clés du clustering de variables

Mesures de similarité	Mesures d'évaluation du clustering
Corrélation au carré (r^2) pour les variables numériques	Inertie intra-cluster (compacité des groupes)
Dissimilité <i>simple matching</i> pour les variables qualitatives	Part d'inertie expliquée (qualité globale de la partition)
Combinaison pondérée (numérique + catégorielle) pour les données mixtes	Adhésion des variables aux clusters (degree of membership)
Dissimilité $d(x_j, x_\ell)$ pour k-medoids	Diagnostics complémentaires : tailles, variables représentatives, inerties par cluster

Algorithmes implémentés (1/4) : k-means de variables (bloc numérique)

Principe

- Méthode conçue pour les **variables numériques**.
- Les variables sont vues comme des vecteurs en dimension n .
- Objectif : regrouper les variables ayant une **structure de corrélation similaire**.

Prototype du cluster

- Prototype $Z_k = \text{composante principale locale}$ (ACP) calculée sur les variables du cluster.
- Critère d'affectation : maximiser $r^2(x_j, Z_k)$, la corrélation au carré entre la variable et son prototype.

Affectation et convergence

- Processus de **réallocation itérative** :
affectation → mise à jour Z_k → convergence.
- Critère global : inertie intra-cluster minimale.

Algorithmes implémentés (2/4) : k-modes de variables (bloc qualitatif)

Principe

- Méthode adaptée aux **variables catégorielles**.
- Regroupe les variables partageant des **profils de modalités similaires**.

Prototype du cluster

- Prototype $m_k(i)$ = **modalité la plus fréquente** pour chaque variable du cluster.
- Dissimilité utilisée : **simple matching** (nombre de désaccords).

Affectation et convergence

- Affectation : une variable est envoyée vers le cluster dont le mode lui ressemble le plus.
- Réallocation itérative : recalcul du mode après chaque cycle.
- Critère global : minimiser les désaccords intra-cluster.

Algorithmes implémentés (3/4) : k-prototypes de variables (données mixtes)

Principe

- Méthode hybride traitant **simultanément des variables numériques et qualitatives.**
- Vise à retrouver des blocs thématiques mixtes cohérents.

Prototype du cluster : prototype hybride

- partie numérique : **composante principale locale** Z_k ;
- partie catégorielle : **profil modal** m_k .

Distance et réallocation

- Distance globale :

$$d_{\text{tot}}(x_j, c_k) = d_{\text{num}}(x_j, Z_k) + \lambda d_{\text{cat}}(x_j, m_k).$$

- λ règle le poids du bloc catégoriel.
- Réallocation itérative : mise à jour conjointe (Z_k, m_k) .

Algorithmes implémentés (4/4) : k-medoids de variables (dissimilarités générales)

Principe

- Méthode basée sur une **matrice de dissimilarités** $D_{j\ell}$.
- Applicable à tous types de variables (num, cat, mixtes).

Prototype du cluster

- Prototype = **médoïde** : une variable réelle représentant le mieux son cluster.
- Choisi pour minimiser la somme des distances :

$$\sum_{x_j \in C_k} D(x_j, \text{médoïde}_k).$$

Affectation et propriétés

- Réallocation de type PAM : échanges médoïde / non-médoïde.
- Méthode **robuste** aux valeurs aberrantes.
- Coût computationnel : calcul de D en $O(p^2)$.

Architecture R6 : base, engines, façade

- **Classe de base mmrClustVarBase :**
 - gère les données actives / nouvelles (FX_active, FX_new) ;
 - stocke FClusters, FCenters, FIertia, FConvergence ;
 - fournit \$fit(), \$predict(), \$summary(), \$plot().
- **Engines spécialisés :**
 - mmrClustVarKMeans, mmrClustVarKModes, mmrClustVarKPrototypes, mmrClustVarKMedoids ;
 - implémentent les hooks : run_clustering(), predict_one_variable(), summary_membership_impl(), plot_membership_impl(), etc.
- **Façade mmrClustVar :**
 - choisit la méthode effective (mode "auto") ;
 - instancie l'engine approprié via create_engine() ;
 - expose compute_inertia_path(), interpret_clusters(), plot().

Exemple d'utilisation (code)

```
library(mmrClustVar)

data("metal_universe", package = "mmrClustVar")
X <- metal_universe

obj <- mmrClustVar$new(
  method = "auto", K = 5,
  scale = TRUE, lambda = 1
)
obj$fit(X)

obj$summary()
obj$plot(type = "inertia")
```

Dataset metal_universe : structure

- Jeu mixte fictif décrivant ≈ 100 groupes de metal (ou assimilés).
- **Variables réalistes :**
 - group_name : nom du groupe ;
 - subgenre : sous-genre (black, death, doom, power, etc.) ;
 - country : pays d'origine ;
 - cat_front_gender : genre du/de la chanteur·se principal·e.
- **Variables numériques fictives mais plausibles :**
 - num_bpm_pref : tempo préféré (bpm), tiré dans des plages réalistes selon le sous-genre ;
 - num_blastbeat_tolerance, num_distortion_preference, num_aggressivity_score, num_heaviness_score, num_speed_score, etc.
- **Variables de bruit :**
 - colonnes générées purement au hasard, sans structure particulière.
- **Variables dérivées :**
 - dérivées de variables numériques (transformations linéaires ou bruitées) ;
 - dérivées de variables catégorielles (regroupements, recodages).

Génération des variables fictives : principe

- **Étape 1 : base réaliste**
 - Curation manuelle de groupes, sous-genres et pays
 - Pour chaque ligne : tirage de subgenre, country, cat_front_gender dans des listes finies.
- **Étape 2 : bloc numérique principal**
 - Pour chaque sous-genre, définition de plages « plausibles » :
 - ex. doom : BPM plus faibles ; death / black : BPM plus élevés ;
 - agressivité, distorsion, lourdeur corrélées mais pas identiques.
 - Tirages aléatoires (uniformes ou normales tronquées) dans ces plages pour num_bpm_pref, num_speed_score, num_heaviness_score, etc.
- **Étape 3 : bruit numérique et catégoriel**
 - Création de colonnes de bruit : valeurs indépendantes et sans structure, pour tester la sensibilité des algorithmes.
- **Étape 4 : dérivées**
 - Variables numériques dérivées :
 - versions bruitées ou transformées (ex. $Y = aX + \varepsilon$) de variables de base ;
 - Variables catégorielles dérivées :
 - regroupements de modalités (ex. fusion de sous-genres proches) ;
 - recodages de subsumation (région du monde, familles de style, etc.).

Objectifs de l'étude metal_universe

- **Usage pédagogique :**
 - démontrer l'utilisation de mmrClustVar sur un jeu mixte ;
 - fournir un « laboratoire contrôlé » où la structure de redondance est connue.
- **Questions explorées :**
 - les algorithmes retrouvent-ils les **blocs de variables corrélées** (variables dérivées) ?
 - arrivent-ils à distinguer :
 - un bloc « rythmique » (BPM, vitesse, variabilité de tempo) ;
 - un bloc « violence du son » (distorsion, agressivité, lourdeur) ;
 - un bloc « socio-géographique » (pays, sous-genre, genre du/de la chanteur·se) ?
 - comment se comportent les **variables de bruit** ?

Configuration typique : k-prototypes, $K = 5$

- Variables actives :
 - bloc numérique : scores esthétiques (BPM, speed, heaviness, aggressivity, etc.) ;
 - bloc catégoriel : subgenre, country, cat_front_gender.
- Variables supplémentaires :
 - variables de bruit explicite ;
 - certaines dérivées secondaires.
- Méthode : **k-prototypes** avec $\lambda \approx 1$.
- Exemple de résultats pour $K = 5$:
 - inertie intra-classe totale $\approx 1,63$;
 - adhésion globale moyenne $\approx 0,875$;
 - adhésion moyenne numérique $\approx 0,82$; adhésion catégorielle ≈ 1 .
- Interprétation :
 - un cluster fortement corrélé autour de la distorsion / agressivité ;
 - un cluster rythmique (BPM, speed score, tempo variability) ;
 - un cluster mélangeant caractéristiques géographiques et style ;
 - variables dérivées généralement regroupées ; bruit peu adhérent.

Comparaison k-prototypes vs k-medoids

- **k-medoids** sur la matrice de dissimilarités D :
 - distance basée sur $1 - r^2$ (numérique), simple matching (catégoriel), 1 pour les paires mixtes.
- Observations sur metal_universe :
 - médoïdes fortement interprétables (ex. num_bpm_pref comme médoïde du cluster rythmique) ;
 - partition globalement cohérente avec k-prototypes ;
 - parfois plus robuste aux variables aberrantes ou bruitées.
- Intérêt pédagogique :
 - comparer deux familles de méthodes sur un même terrain de jeu ;
 - illustrer le coût de calcul $O(p^2)$ de k-medoids lorsque le nombre de variables augmente.

- **Bloc numérique :**
 - dépendance aux composantes principales locales (ACP) pour résumer les clusters de variables numériques ;
 - hypothèse implicite de structures plutôt linéaires.
- **Pondération :**
 - fixé par l'utilisateur pour la partie catégorielle des méthodes mixtes ;
 - pas encore de mécanisme automatique de calibration.
- **k-medoids :**
 - construction de la matrice de dissimilarités en $O(p^2)$;
 - peut devenir coûteux sur des tableaux très larges en nombre de variables.
- **Aspect hiérarchique :**
 - pas encore de méthode hiérarchique de clustering de variables intégrée dans mmrClustVar (seulement des dendrogrammes en aval).

• Algorithmes et critères

- sélection automatique ou guidée de (équilibrage contributions num/cat) ;
- initialisations plus sophistiquées (type *k-means++*) pour stabiliser les solutions ;
- intégration de nouvelles mesures de similarité (corrélations non linéaires, etc.) ;
- ajout de **méthodes hiérarchiques** de clustering de variables.

• Dataset metal_universe

- enrichir la partie « réelle » via du **web scraping** contrôlé ;
- documenter plus finement le **processus de génération** des variables fictives (distribution par sous-genre, corrélations cibles, etc.) ;
- proposer plusieurs versions du dataset (plus / moins bruitées) pour la comparaison d'algorithmes.

• Shiny et expérience utilisateur

- sauvegarde / rechargement de sessions ;
- thèmes graphiques, filtres plus avancés, aide contextuelle intégrée.

- `mmrClustVar` remplit le cahier des charges du projet :
 - au moins trois méthodes, dont réallocation et traitement qualitatif ;
 - classe R6 complète, aide en anglais, application Shiny ;
 - stratégie de choix de K et outils d'interprétation.
- Le package va au-delà des attentes :
 - quatre algorithmes (k-means, k-modes, k-prototypes, k-medoids) ;
 - architecture extensible (base + engines + façade) ;
 - jeux internes variés + dataset thématique `metal_universe` ;
 - fonctions d'export et tutoriel reproductible.
- `metal_universe` sert de « laboratoire » pour comparer les méthodes sur des structures de redondance contrôlées.

Merci pour votre attention !
Questions ?