

# Clustering K-prototypes

Équipe 19 – ANACLET Kylian, BIN DAIVIN Muhamad Zaiinizee, CHENNOUF Younes,  
KEBAIRI Ilyes

2025-12-10

Le jeu de données contient à la fois des variables numériques et catégorielles (région, condition d'éclairage, condition atmosphérique, etc.). L'algorithme K-means classique n'est donc pas adapté, car il repose sur la distance euclidienne, qui n'a pas de sens pour les variables catégorielles, même après encodage one-hot.

Il est préférable d'utiliser **K-prototypes**, qui combine la distance euclidienne pour les variables numériques et une mesure de dissimilarité spécifique pour les variables catégorielles. Cet algorithme permet ainsi d'obtenir des clusters plus fiables et cohérents.

## K-prototypes

L'algorithme K-prototypes implémente la méthode de partitionnement par minimisation de la distance entre les observations et les centres de clusters. Plus précisément, il cherche à minimiser la fonction suivante :

$$\min_{c_1, \dots, c_K} \sum_{k=1}^K \sum_{i, G(i)=k} d(x^{(i)}, c_k)$$

où  $d(x^{(i)}, c_k)$  représente la distance entre l'observation  $x^{(i)}$  et le centre du cluster  $c_k$ . Cette distance combine la distance euclidienne pour les variables numériques et une mesure de dissimilarité pour les variables catégorielles.

La minimisation est réalisée automatiquement par la fonction `kproto()` à travers un processus itératif. L'algorithme affecte chaque observation au cluster dont le centre est le plus proche, puis recalcule les centres, et répète ces étapes jusqu'à convergence. Le résultat de cette minimisation est observable via `tot.withinss`, qui représente la somme totale des distances intra-clusters.

## A. Chargement et préparation des données

Le jeu de données final étant **très volumineux**, un échantillon aléatoire a été extrait pour faciliter le traitement et le clustering.

```
if(!require(dplyr)) install.packages("dplyr")
```

```
## Le chargement a nécessité le package : dplyr
```

```
##
```

```
## Attachement du package : 'dplyr'
```

```
## Les objets suivants sont masqués depuis 'package:stats':
##
##     filter, lag

## Les objets suivants sont masqués depuis 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(dplyr)

set.seed(321)
n_sample <- 5000

# Exécuter le script « nettoyage_donnees_clustering.Rmd », situé dans le dossier clustering, afin de gé
df <- read.csv("../data/donnees_clustering.csv")
df_complete <- df %>%
  mutate(across(where(is.character), as.factor)) %>%
  select(agg, lum, atm, col, vma)

# Convertir toutes les variables de type character en factor
# Puis sélectionner uniquement les 5 variables retenues pour le clustering
df <- df %>%
  mutate(across(where(is.character), as.factor)) %>%
  select(agg, lum, atm, col, vma)

# On tire par hasard 1000 accidents
df <- df[sample(nrow(df), n_sample), ]
```

On sélectionne ces 5 variables (agg, lum, atm, col, vma) car elles capturent les conditions principales de l'accident (localisation, luminosité, météo, type de collision et vitesse autorisée) et donnent les meilleurs résultats de clustering.

```
str(df)
```

```
## 'data.frame':    5000 obs. of  5 variables:
## $ agg: Factor w/ 2 levels "en agglo","hors agglo": 1 1 1 2 1 1 2 1 1 2 ...
## $ lum: Factor w/ 5 levels "crepuscule ou aube",...: 2 5 5 5 2 5 1 5 5 1 ...
## $ atm: Factor w/ 9 levels "autre","brouillard ou fumee",...: 8 3 6 6 6 6 6 3 6 ...
## $ col: Factor w/ 7 levels "3+ veh - en chaine",...: 4 4 3 6 7 5 5 6 7 5 ...
## $ vma: int   50 50 50 80 50 50 80 50 50 -1 ...
```

## B. Verification de la légitimité de clustering

Avant de commencer on doit tout d'abord s'assurer que l'utilisation de la méthode de clustering est légitime ici (s'assurer que les données ne sont pas réparties de façon homogène). Pour ce faire on utilise la méthode statistique en calculant la statistique de Hopkins calculé avec la formule :

$$H = \frac{\sum_{i=1}^n x_i}{\sum_{i=1}^n x_i + \sum_{i=1}^n y_i}$$

où :

- $x_i$  la distance d'un i-ème point  $p_i$  choisi aléatoirement dans notre distribution à tester avec son plus proche voisin dans la même distribution
- $y_i$  : la distance d'un i-ème point  $q_i$  choisi aléatoirement dans une distribution aléatoire uniforme avec son plus proche voisin dans la distribution

Nos hypothèses étant :

- $H_0$  : Les données sont uniformément distribués
- $H_1$  : Les données ne sont pas uniformément distribués

```
if(!require(hopkins)) install.packages("hopkins")

## Le chargement a nécessité le package : hopkins

library(hopkins)

numeric_cols <- sapply(df, is.numeric)
n_numeric <- sum(numeric_cols)

if (n_numeric >= 2) {
  h_stat <- hopkins(df[, numeric_cols])
  cat("Statistique de Hopkins :", round(h_stat, 4), "\n")
} else {
  cat("Hopkins non calculable : seulement", n_numeric, "variable numérique (vma).\n")
  cat("    (La fonction nécessite au moins 2 variables numériques)\n")
}

## Hopkins non calculable : seulement 1 variable numérique (vma).
##    (La fonction nécessite au moins 2 variables numériques)
```

## C. Choix du nombre de clusters (k)

On applique l'algorithme k-prototypes pour différentes valeurs de k.

```
set.seed(321)
if(!require(clustMixType)) install.packages("clustMixType")

## Le chargement a nécessité le package : clustMixType

if(!require(cluster)) install.packages("cluster")

## Le chargement a nécessité le package : cluster

library(clustMixType)
library(cluster)

k_max <- 10
cost <- numeric(k_max)
```

```

sil_avg <- numeric(k_max)

# Calculer la distance de Gower sur les données non normalisées
d <- daisy(df, metric = "gower")

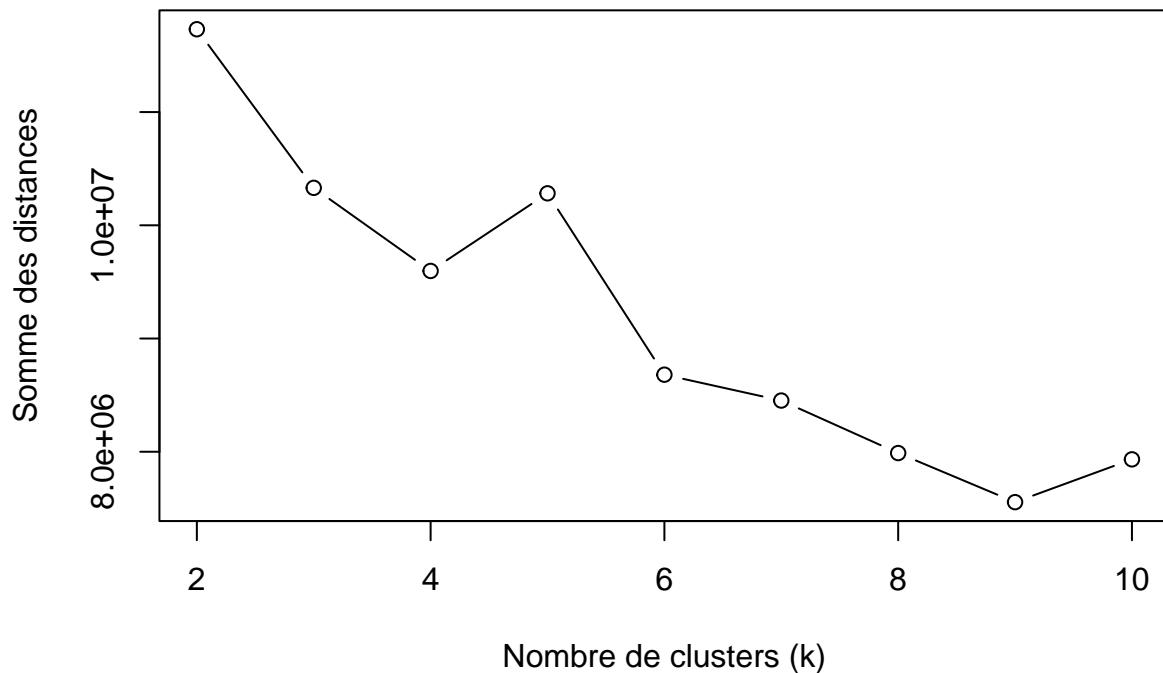
# Exécuter k-prototypes pour différentes valeurs de k
for (k in 2:k_max) {
  kp <- kproto(df, k, verbose = FALSE)
  cost[k] <- kp$tot.withinss

  sil <- silhouette(kp$cluster, d)
  sil_avg[k] <- mean(sil[, 3])
}

# Plot méthode du coude
plot(2:k_max, cost[2:k_max],
     main = "Méthode du coude pour choisir k",
     type = "b", xlab = "Nombre de clusters (k)",
     ylab = "Somme des distances")

```

## Méthode du coude pour choisir k

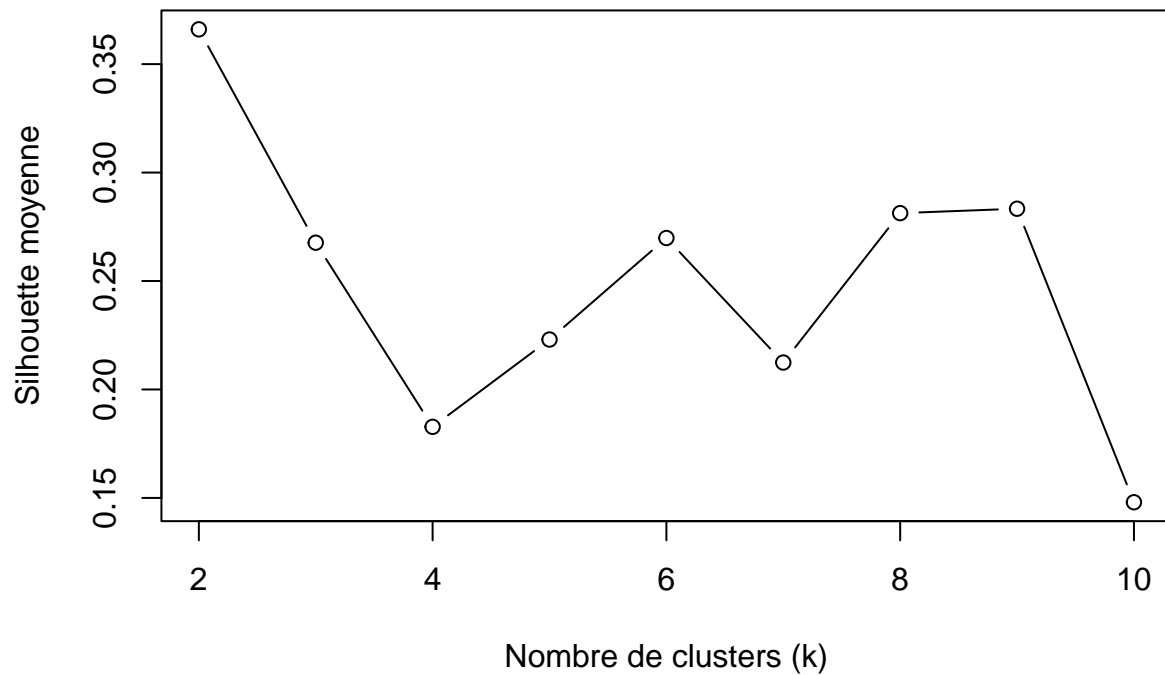


```

# Plot méthode de la silhouette
plot(2:k_max, sil_avg[2:k_max],
     main = "Méthode de la silhouette pour choisir k",
     type = "b", xlab = "Nombre de clusters (k)",
     ylab = "Silhouette moyenne")

```

## Méthode de la silhouette pour choisir k



### Observations

La méthode du coude montre une stabilisation à partir de  $k = 4$ , tandis que l'indice de Silhouette est maximal à  $k = 2$ .

Pour choisir le meilleur  $k$ , nous testons  $k = 2, 3$  et  $4$  en comparant leurs indices de qualité.

## D. Application de k-prototypes et analyse des résultats

```
set.seed(321)
res_k2 <- kproto(df, k = 2, verbose = FALSE)
res_k3 <- kproto(df, k = 3, verbose = FALSE)
res_k4 <- kproto(df, k = 4, verbose = FALSE)
```

## E. Test de la qualité du clustering

Afin d'évaluer la qualité du clustering on évalue les deux indices :

- l'indice de Silhouette pour chaque cluster qui pour chaque élément évalue à quel point un élément est "similaire" par rapport aux autres éléments du même cluster .
- l'indice de Dunn qui est défini comme suit :

$$D = \frac{\text{min.separation}}{\text{max.diameter}}$$

De plus on veille à adapter cette methode d'évaluation à la methode de clustering utilisée (k-prototype) et ce en utilisant justement la distance de gower qui prend en compte les variables catégorielles

```
if(!require(fpc)) install.packages("fpc")

## Le chargement a nécessité le package : fpc

library(cluster)
library(fpc)

dist_gower <- daisy(df, metric = "gower")

evaluer_k <- function(res, k_val) {
  cat("=== k =", k_val, "===\n")
  sil <- silhouette(res$cluster, dist_gower)
  cat("Silhouette moyenne :", round(mean(sil[,3]), 4), "\n")
  cat("Détail :", paste(round(summary(sil)$clus.avg.widths, 4), collapse = ", "), "\n")
  dunn <- cluster.stats(dist_gower, res$cluster)$dunn
  cat("Dunn :", round(dunn, 6), "\n\n")
}

evaluer_k(res_k2, 2)

## === k = 2 ===
## Silhouette moyenne : 0.366
## Détail : 0.3327, 0.3853
## Dunn : 0.001128

evaluer_k(res_k3, 3)

## === k = 3 ===
## Silhouette moyenne : 0.2677
## Détail : 0.2861, 0.3107, 0.2277
## Dunn : 0.001133

evaluer_k(res_k4, 4)

## === k = 4 ===
## Silhouette moyenne : 0.1828
## Détail : 0.3536, 0.1734, 0.1281, 0.034
## Dunn : 0.00226
```

Bien que  $k = 2$  offre la meilleure qualité statistique avec des clusters très cohérents, une partition en seulement deux groupes est trop simpliste pour refléter la diversité des accidents routiers. Nous retenons donc  $k = 3$ , qui propose trois profils plus nuancés et interprétables, malgré une qualité statistique légèrement moindre.

```
table(res_k3$cluster)
```

### Distribution des observations par cluster

```
##  
##      1      2      3  
## 1807 1135 2058
```

L'analyse des clusters montre que les quatres groupes identifiés présentent des tailles différentes. Le cluster 2 regroupe la majorité des observations, tandis que le cluster 1 correspond a un petit sous-groupe spécifique. Les clusters 3 occupe une taille intermédiaire.

Cette répartition révèle l'existence de groupes homogènes mais de taille variées, permettant d'identifier et d'interpréter les profils distincts propres à chaque cluster.

*Remarque : Ces effectifs varient selon l'échantillon sélectionné.*

## F. Comparaison échantillon vs données complètes

### Moyennes des variables numériques par cluster

```
library(clustMixType)  
  
# Moyennes vma  
cat("Moyenne vma sur l'échantillon :\n")
```

```
## Moyenne vma sur l'échantillon :
```

```
print(round(tapply(df$vma, res_k3$cluster, mean), 1))
```

```
##      1      2      3  
## 85.8 41.8 49.0
```

```
res_full <- kproto(df_complete, k = 3, verbose = FALSE)  
cat("\nMoyenne vma sur les données complètes :\n")
```

```
##  
## Moyenne vma sur les données complètes :
```

```
print(round(tapply(df_complete$vma, res_full$cluster, mean), 1))
```

```
##      1      2      3  
## 48.9 42.3 87.8
```

### Répartition des variables catégorielles par cluster

```

cat_cols <- sapply(df, is.factor)

for (col in names(df)[cat_cols]) {
  cat("\n=== Variable :", col, "===\n\n")

  cat("Échantillon :\n")
  print(table(df[[col]], res_k3$cluster))

  cat("\nDonnées complètes :\n")
  print(table(df_complete[[col]], res_full$cluster))

  cat("\n")
}

```

```

##
## === Variable : agg ===
##
## Échantillon :
##
##           1      2      3
## en agglo   16 1090 1995
## hors agglo 1791   45   63
##
## Données complètes :
##
##           1      2      3
## en agglo  12540 6938   135
## hors agglo   789   568 10146
##
##
## === Variable : lum ===
##
## Échantillon :
##
##           1      2      3
## crepuscule ou aube      220  113  209
## nuit avec éclairage allume    69  350  533
## nuit avec éclairage non allume  18   13   14
## nuit sans éclairage      486   53   53
## plein jour      1014  606 1249
##
## Données complètes :
##
##           1      2      3
## crepuscule ou aube      1350  768 1189
## nuit avec éclairage allume 3408 2159 419
## nuit avec éclairage non allume 137  111  104
## nuit sans éclairage      417  437 2961
## plein jour      8017 4031 5608
##
##
## === Variable : atm ===
##

```



```

## Échantillon :
##
##      1      2      3
##  autre      28      3     10
##  brouillard ou fumee      33      4      9
##  couvert      75     52     83
##  éblouissant      28     35     33
##  neige ou grele      16      3      6
##  normale     1375    864   1649
##  pluie forte      49     27     26
##  pluie legere     196    142    239
##  vent fort ou tempete      7      5      3
##
## Données complètes :
##
##      1      2      3
##  autre      61     40    120
##  brouillard ou fumee      60     53    215
##  couvert     512    334    420
##  éblouissant     191    192    172
##  neige ou grele      30     28     75
##  normale    10773   5674   7821
##  pluie forte      204    168    250
##  pluie legere     1482   1001   1176
##  vent fort ou tempete      16     16     32
##
##
## === Variable : col ===
##
## Échantillon :
##
##      1      2      3
##  3+ veh - en chaine    108      5     60
##  3+ veh - multiple      97     11     50
##  autre      465    958      0
##  deux veh - arriere    323     59    295
##  deux veh - cote      324      0   1252
##  deux veh - frontal    218     48    216
##  sans      272     54    185
##
## Données complètes :
##
##      1      2      3
##  3+ veh - en chaine    414     42    722
##  3+ veh - multiple     342     53    522
##  autre      0   6380   2579
##  deux veh - arriere   2003    336   1861
##  deux veh - cote     7606      0   1864
##  deux veh - frontal   1620    365   1303
##  sans      1344    330   1430

```

## G. Visualisation des clusters k-prototypes

```
library(ggplot2)
library(dplyr)
library(gridExtra)

##
## Attachement du package : 'gridExtra'

## L'objet suivant est masqué depuis 'package:dplyr':
##
##      combine

library(scales) # Pour percent()

df_vis <- df %>% mutate(cluster = as.factor(res_k3$cluster))

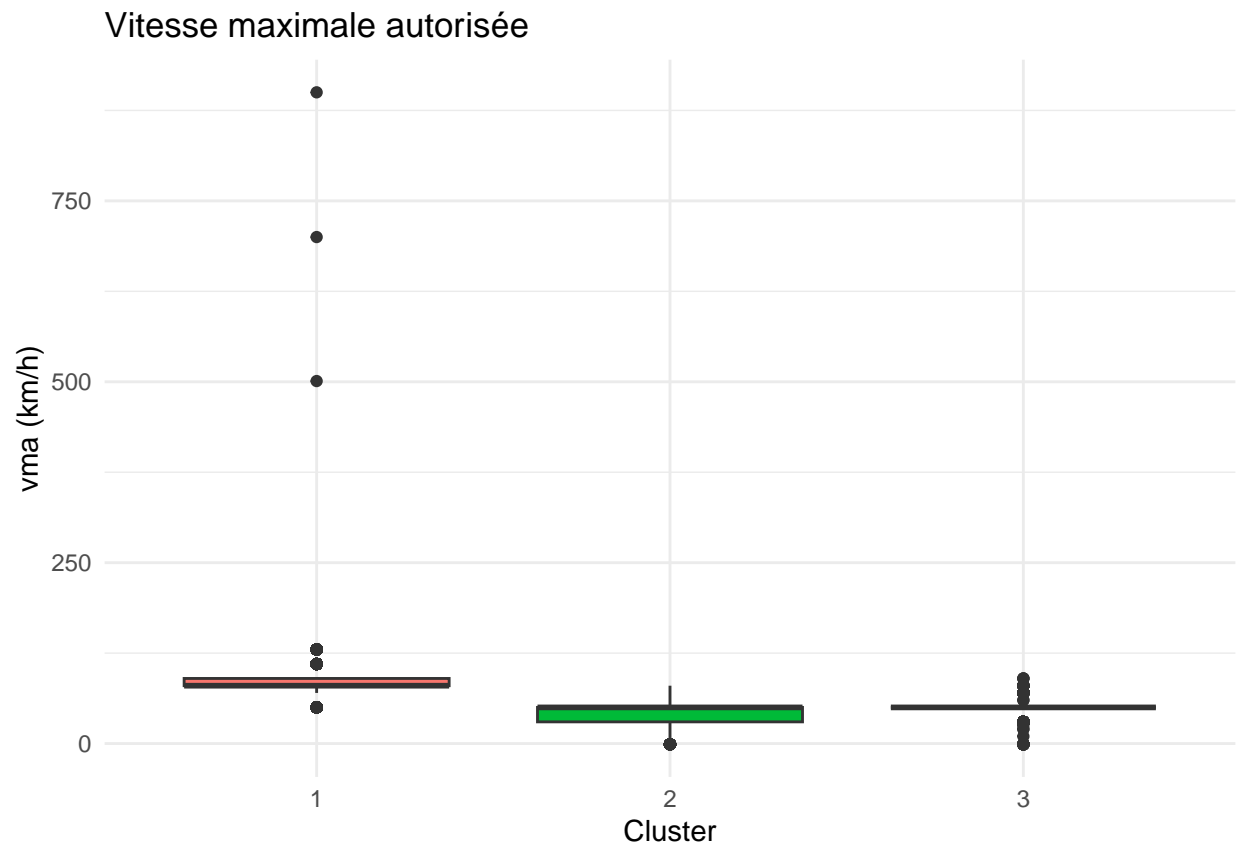
p1 <- ggplot(df_vis, aes(x = cluster, y = vma, fill = cluster)) +
  geom_boxplot() + theme_minimal() + theme(legend.position = "none") +
  labs(title = "Vitesse maximale autorisée", x = "Cluster", y = "vma (km/h)")

p2 <- ggplot(df_vis, aes(x = cluster, fill = agg)) + geom_bar(position = "fill") +
  scale_y_continuous(labels = percent) + theme_minimal() +
  labs(title = "Localisation", x = "Cluster", y = "Proportion")

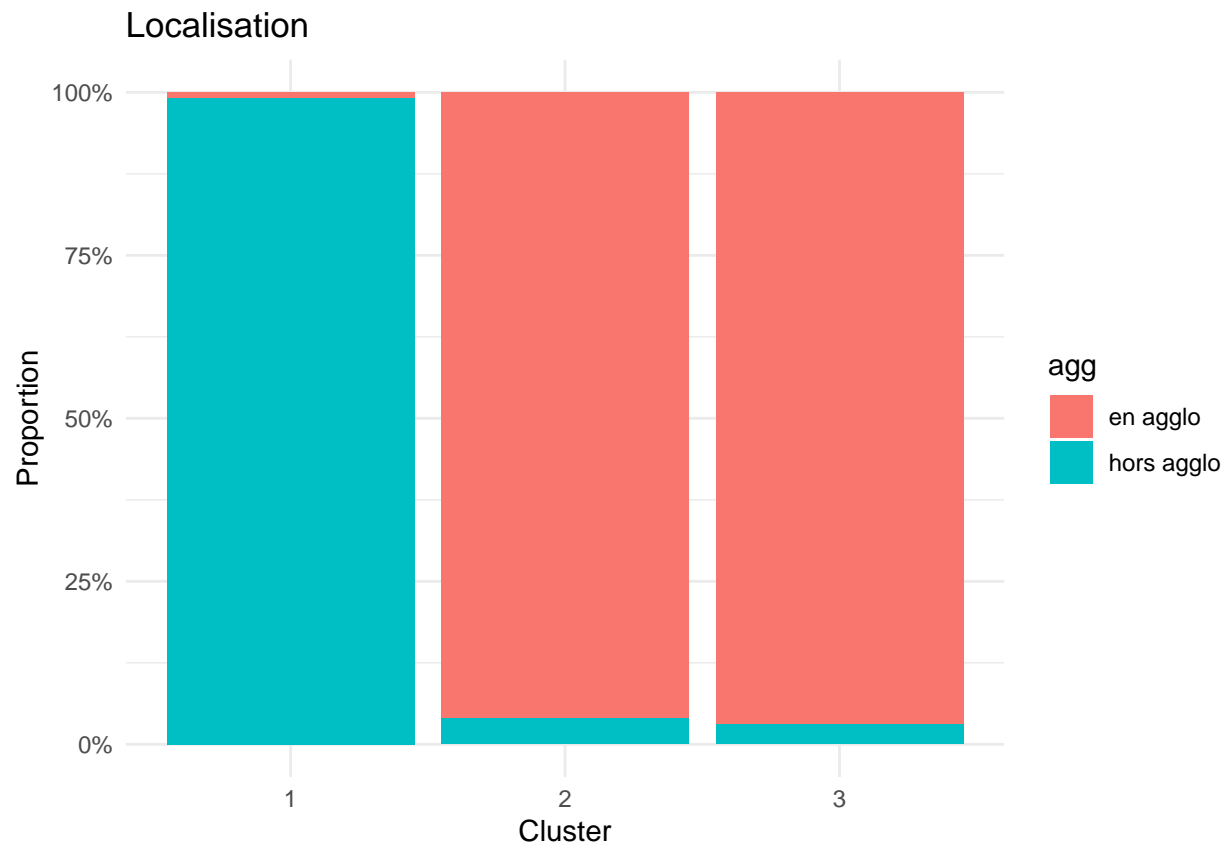
p3 <- ggplot(df_vis, aes(x = cluster, fill = lum)) + geom_bar(position = "fill") +
  scale_y_continuous(labels = percent) + theme_minimal() +
  labs(title = "Luminosité", x = "Cluster", y = "Proportion")

p4 <- ggplot(df_vis, aes(x = cluster, fill = atm)) + geom_bar(position = "fill") +
  scale_y_continuous(labels = percent) + theme_minimal() +
  labs(title = "Conditions atmosphériques", x = "Cluster", y = "Proportion")

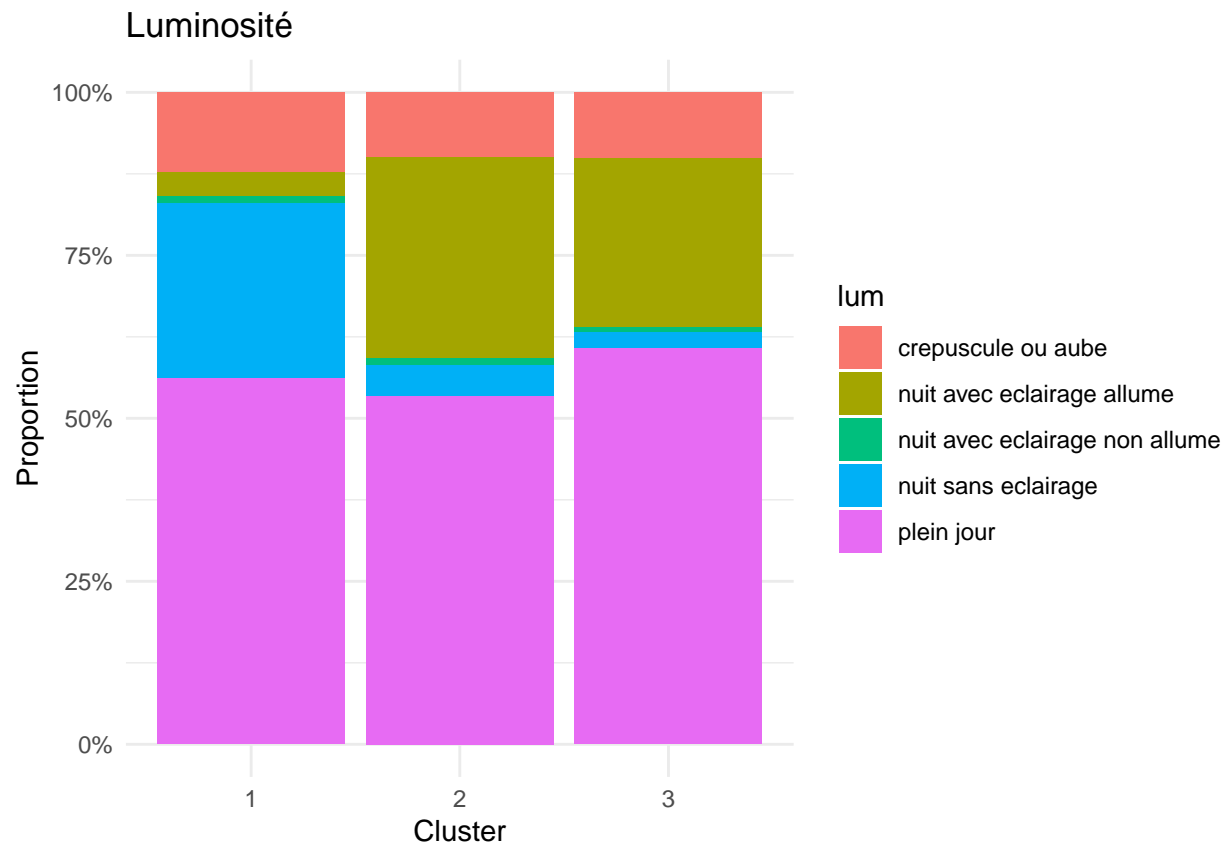
p5 <- ggplot(df_vis, aes(x = cluster, fill = col)) + geom_bar(position = "fill") +
  scale_y_continuous(labels = percent) + theme_minimal() +
  labs(title = "Type de collision", x = "Cluster", y = "Proportion")
p1
```



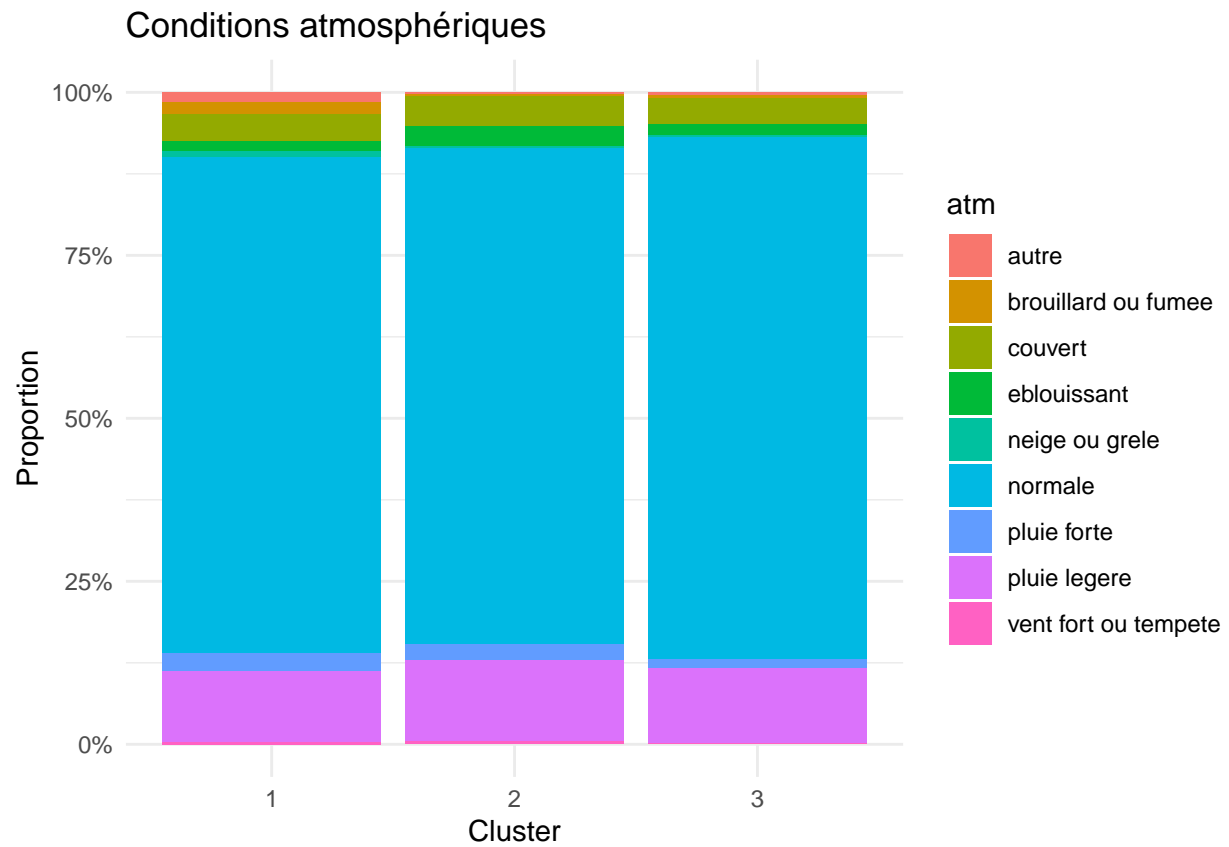
p2



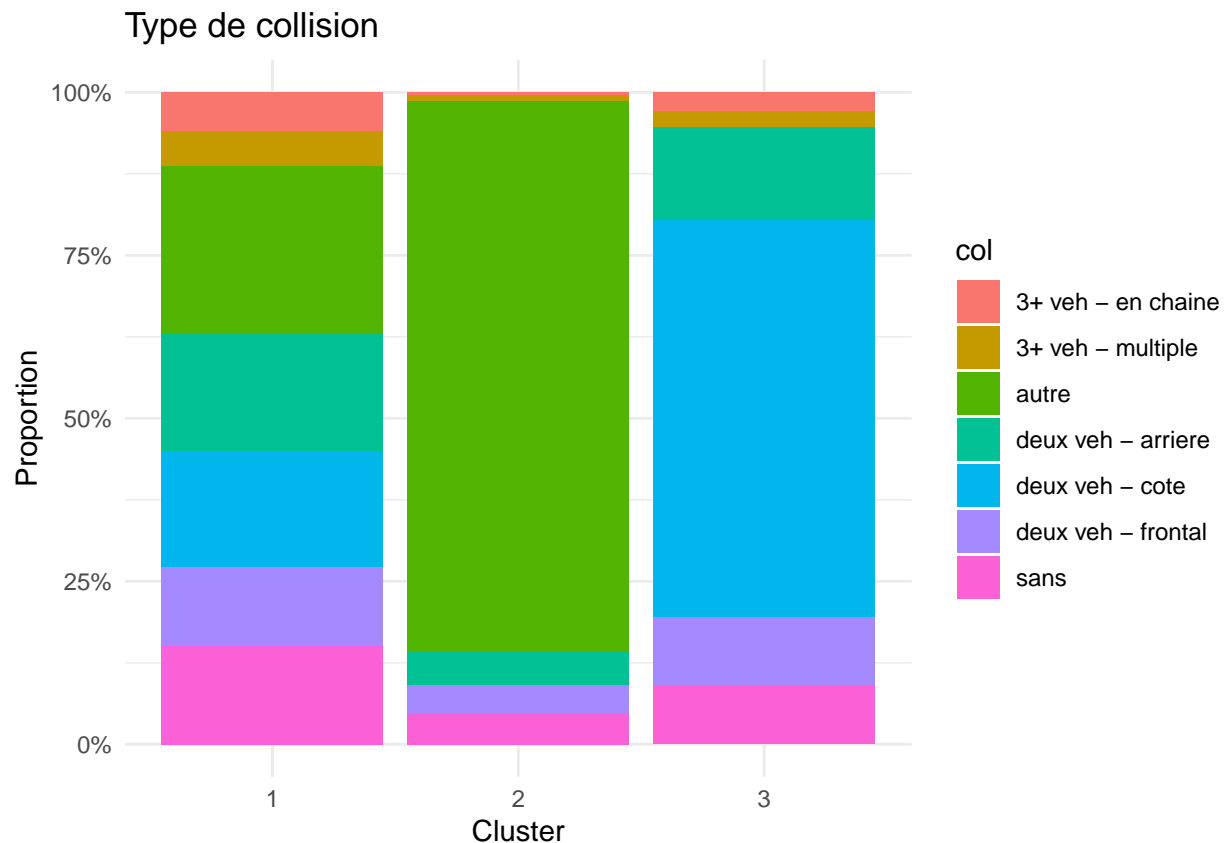
p3



p4



p5



```
set.seed(321)
library(cluster)
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library(viridis)
```

```
## Le chargement a nécessité le package : viridisLite
```

```
##
```

```
## Attachement du package : 'viridis'
```

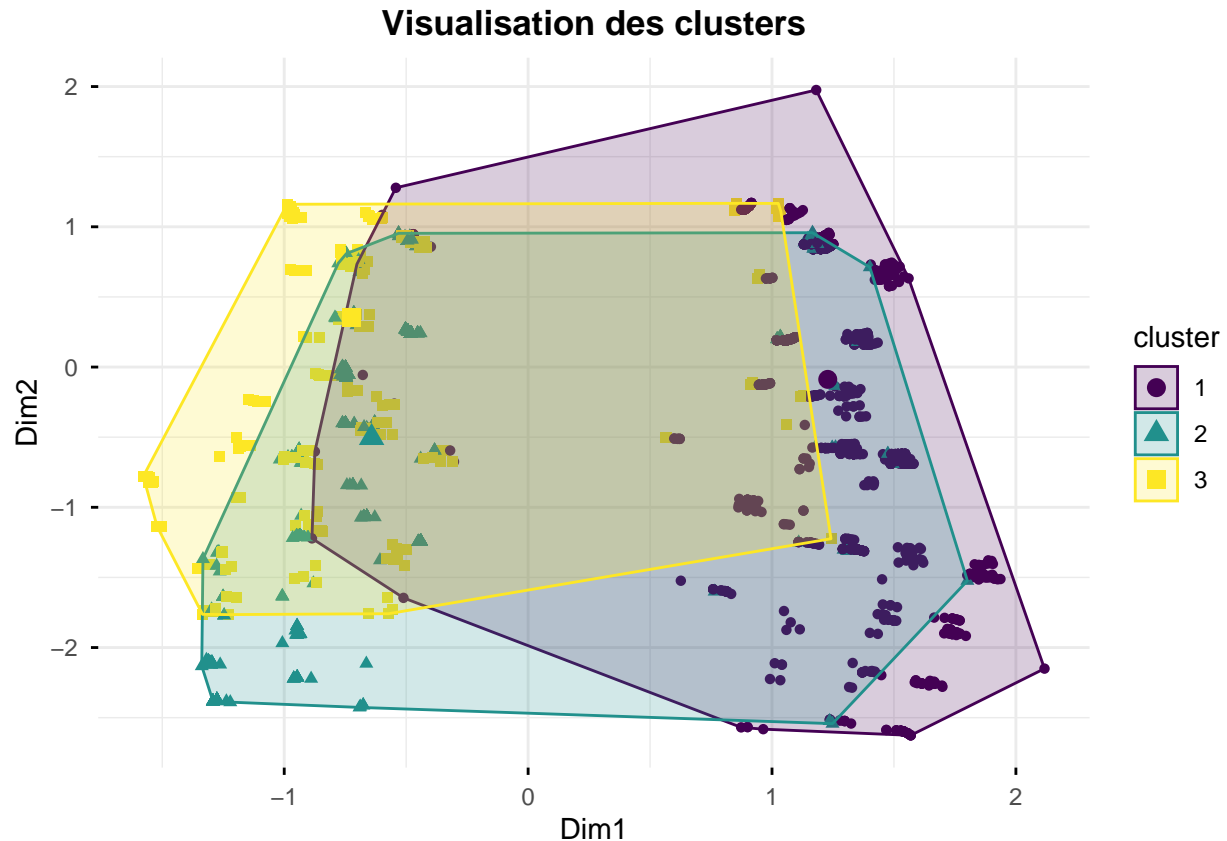
```
## L'objet suivant est masqué depuis 'package:scales':
```

```
##
```

```
## viridis_pal
```

```
mds_df <- cmdscale(dist_gower, k = 2) %>% as.data.frame()
colnames(mds_df) <- c("Dim1", "Dim2")
```

```
fviz_cluster(list(data = mds_df, cluster = res_k3$cluster),
  geom = "point", palette = viridis(3), ellipse.type = "convex",
  show.clust.cent = TRUE, ggtheme = theme_minimal(),
  main = "Visualisation des clusters") +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"))
```



```
ggsave("../Figures/clusters_fviz_final.png", width = 10, height = 8, dpi = 300)
```

Les groupes peuvent se chevaucher car la visualisation 2D (via PCA) projette un espace multidimensionnel sur un simple plan. Cette réduction dimensionnelle entraîne une perte d'information, et des clusters bien séparés dans l'espace original peuvent alors apparaître mélangés sur le graphique.

## H. Interprétation de chaque cluster

### Cluster 1 : Accidents Urbains sous Intempéries

Ce groupe se distingue quasi-exclusivement par des conditions météorologiques et de surface dégradées.

- **Conditions** : Forte prédominance de la pluie (légère ou forte) associée à une surface mouillée.
- **Localisation** : Majoritairement en agglomération.
- **Véhicules impliqués** : Une surreprésentation notable des transports en commun, suggérant des problématiques de freinage ou d'adhérence impliquant des véhicules lourds dans un trafic dense.
- **Cinétique** : Vitesse moyenne faible.

### Cluster 2 : Accidents de Flux Urbain

Il s'agit du segment le plus volumineux, représentant l'accidentologie "quotidienne" en ville par conditions normales.



- **Configuration** : Exclusivement en agglomération, concentré sur les intersections (carrefours, giratoires) avec une prédominance de chocs latéraux (refus de priorité, changements de file).
- **Usagers Vulnérables** : Forte implication des deux-roues (vélos et motos).
- **Conditions** : Environnement favorable (plein jour, météo normale, surface sèche).
- **Cinétique** : Vitesse moyenne la plus basse des trois groupes.

### Cluster 3 : Accidents Routiers à Haute Cinétique

Ce cluster regroupe les accidents hors agglomération, caractérisés par une dangerosité potentielle accrue due à la vitesse.

- **Localisation** : Routes départementales et autoroutes (hors agglomération).
- **Facteurs aggravants** : Surreprésentation des accidents nocturnes (“Nuit sans éclairage”) et des tracés en courbe.
- **Typologie** : Nombreuses pertes de contrôle (sorties de route sur accotement) ou chocs frontaux.
- **Véhicules** : Implication majeure des véhicules légers et poids lourds ; absence quasi-totale d’usagers vulnérables (piétons/vélos).
- **Cinétique** : Vitesse moyenne très élevée.

## I. Conclusion

L’algorithme K-prototypes a identifié quatre groupes d’accidents distincts en combinant variables numériques et catégorielles. Cette segmentation permet d’analyser les différents profils d’accidents et d’identifier les facteurs de risque spécifiques à chaque groupe.

Ces résultats dépendent de l’échantillon sélectionné et pourraient varier avec un échantillonnage différent.