Kmeans\_Clustering\_R

# Analyse de la base de données ‘Live’

La base de données ‘Live’ contient des informations collectées auprès de vendeurs sur Facebook en Thaïlande.

Elle comprend différents types de statuts (photo, vidéo, etc.) ainsi que diverses métriques telles que les ‘likes’, ‘shares’, et ‘comments’.

Les détails précis des variables peuvent varier en fonction du fichier de données spécifique utilisé.

La base de données contient les inforamtions suivantes:

- status\_id : L’ID unique de chaque publication.

- status\_type : Le type de publication (par exemple, photo, statut, lien, vidéo).

- status\_published : La date et l’heure de la publication.

- num\_reactions : Le nombre total de réactions à la publication (par exemple, likes, love, wow, haha, sad, angry).

- num\_comments : Le nombre total de commentaires sur la publication.

- num\_shares : Le nombre total de partages de la publication.

- num\_likes : Le nombre total de “J’aime” pour la publication.

- num\_loves : Le nombre total de “Love” réactions pour la publication.

- num\_wows : Le nombre total de “Wow” réactions pour la publication.

- num\_hahas : Le nombre total de “Haha” réactions pour la publication.

- num\_sads : Le nombre total de “Sad” réactions pour la publication.

- num\_angrys : Le nombre total de “Angry” réactions pour la publication.

L’objectif de cette analyse est **de découvrir des modèles et des tendances dans les données qui peuvent nous aider à comprendre comment les utilisateurs de Facebook en Thaïlande** interagissent avec les publications des vendeurs en ligne.

Ces informations peuvent être utilisées pour **optimiser la stratégie de contenu des vendeurs**, améliorer l’engagement des utilisateurs et, finalement, augmenter les ventes.

5 étapes clés pour réaliser les K-means

* Étape 1 : Analyse et traitement des données
* Étape 2 : Analyse de la relation qui existe entre les variables
* Étape 3 : Analyse de la relation multivariée entre les variables
* Étape 4 : Choix du nombre de groupes optimal
* Étape 5 : Interprétation des résultats

# Etape 1: Analyse et traitement des données

4 points clés à vérifier lors de la description des données :

* Point 1 : Transformation des variables
* Point 2 : Analyse des valeurs manquantes
* Point 3 : Traitement des données aberrantes
* Point 4 : Description des données

## Point 1: Transformation des variables

chemin<-"/Users/natachanjongwayepnga/Documents/GitHub/LeCoinStat/Kmeans\_Logiciel\_R/data"# Remplacez par le chemin dans lequel se trouve votre base de données  
base\_publication<-read.csv(paste0(chemin,"/Live.csv"))  
  
#Visualiser les premières lignes de la base de données   
  
head(base\_publication)

status\_id status\_type status\_published num\_reactions  
1 246675545449582\_1649696485147474 video 4/22/2018 6:00 529  
2 246675545449582\_1649426988507757 photo 4/21/2018 22:45 150  
3 246675545449582\_1648730588577397 video 4/21/2018 6:17 227  
4 246675545449582\_1648576705259452 photo 4/21/2018 2:29 111  
5 246675545449582\_1645700502213739 photo 4/18/2018 3:22 213  
6 246675545449582\_1645650162218773 photo 4/18/2018 2:14 217  
 num\_comments num\_shares num\_likes num\_loves num\_wows num\_hahas num\_sads  
1 512 262 432 92 3 1 1  
2 0 0 150 0 0 0 0  
3 236 57 204 21 1 1 0  
4 0 0 111 0 0 0 0  
5 0 0 204 9 0 0 0  
6 6 0 211 5 1 0 0  
 num\_angrys Column1 Column2 Column3 Column4  
1 0 NA NA NA NA  
2 0 NA NA NA NA  
3 0 NA NA NA NA  
4 0 NA NA NA NA  
5 0 NA NA NA NA  
6 0 NA NA NA NA

#Types des variables dans la base de données  
str(base\_publication)

'data.frame': 7050 obs. of 16 variables:  
 $ status\_id : chr "246675545449582\_1649696485147474" "246675545449582\_1649426988507757" "246675545449582\_1648730588577397" "246675545449582\_1648576705259452" ...  
 $ status\_type : chr "video" "photo" "video" "photo" ...  
 $ status\_published: chr "4/22/2018 6:00" "4/21/2018 22:45" "4/21/2018 6:17" "4/21/2018 2:29" ...  
 $ num\_reactions : int 529 150 227 111 213 217 503 295 203 170 ...  
 $ num\_comments : int 512 0 236 0 0 6 614 453 1 9 ...  
 $ num\_shares : int 262 0 57 0 0 0 72 53 0 1 ...  
 $ num\_likes : int 432 150 204 111 204 211 418 260 198 167 ...  
 $ num\_loves : int 92 0 21 0 9 5 70 32 5 3 ...  
 $ num\_wows : int 3 0 1 0 0 1 10 1 0 0 ...  
 $ num\_hahas : int 1 0 1 0 0 0 2 1 0 0 ...  
 $ num\_sads : int 1 0 0 0 0 0 0 0 0 0 ...  
 $ num\_angrys : int 0 0 0 0 0 0 3 1 0 0 ...  
 $ Column1 : logi NA NA NA NA NA NA ...  
 $ Column2 : logi NA NA NA NA NA NA ...  
 $ Column3 : logi NA NA NA NA NA NA ...  
 $ Column4 : logi NA NA NA NA NA NA ...

### Suppression des colonnes inutiles

#install.packages("dplyr")  
# Assurez-vous que le package dplyr est chargé  
library(dplyr)

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':  
  
 filter, lag

The following objects are masked from 'package:base':  
  
 intersect, setdiff, setequal, union

# Suppression des colonnes inutiles  
base\_publication<-base\_publication %>% select(-c("Column1", "Column2", "Column3","Column4", "status\_published"))  
  
str(base\_publication)

'data.frame': 7050 obs. of 11 variables:  
 $ status\_id : chr "246675545449582\_1649696485147474" "246675545449582\_1649426988507757" "246675545449582\_1648730588577397" "246675545449582\_1648576705259452" ...  
 $ status\_type : chr "video" "photo" "video" "photo" ...  
 $ num\_reactions: int 529 150 227 111 213 217 503 295 203 170 ...  
 $ num\_comments : int 512 0 236 0 0 6 614 453 1 9 ...  
 $ num\_shares : int 262 0 57 0 0 0 72 53 0 1 ...  
 $ num\_likes : int 432 150 204 111 204 211 418 260 198 167 ...  
 $ num\_loves : int 92 0 21 0 9 5 70 32 5 3 ...  
 $ num\_wows : int 3 0 1 0 0 1 10 1 0 0 ...  
 $ num\_hahas : int 1 0 1 0 0 0 2 1 0 0 ...  
 $ num\_sads : int 1 0 0 0 0 0 0 0 0 0 ...  
 $ num\_angrys : int 0 0 0 0 0 0 3 1 0 0 ...

### Résumé rapide des données

#Résumé rapide de la base de données  
summary(base\_publication)

status\_id status\_type num\_reactions num\_comments   
 Length:7050 Length:7050 Min. : 0.0 Min. : 0.0   
 Class :character Class :character 1st Qu.: 17.0 1st Qu.: 0.0   
 Mode :character Mode :character Median : 59.5 Median : 4.0   
 Mean : 230.1 Mean : 224.4   
 3rd Qu.: 219.0 3rd Qu.: 23.0   
 Max. :4710.0 Max. :20990.0   
 num\_shares num\_likes num\_loves num\_wows   
 Min. : 0.00 Min. : 0.0 Min. : 0.00 Min. : 0.000   
 1st Qu.: 0.00 1st Qu.: 17.0 1st Qu.: 0.00 1st Qu.: 0.000   
 Median : 0.00 Median : 58.0 Median : 0.00 Median : 0.000   
 Mean : 40.02 Mean : 215.0 Mean : 12.73 Mean : 1.289   
 3rd Qu.: 4.00 3rd Qu.: 184.8 3rd Qu.: 3.00 3rd Qu.: 0.000   
 Max. :3424.00 Max. :4710.0 Max. :657.00 Max. :278.000   
 num\_hahas num\_sads num\_angrys   
 Min. : 0.0000 Min. : 0.0000 Min. : 0.0000   
 1st Qu.: 0.0000 1st Qu.: 0.0000 1st Qu.: 0.0000   
 Median : 0.0000 Median : 0.0000 Median : 0.0000   
 Mean : 0.6965 Mean : 0.2437 Mean : 0.1132   
 3rd Qu.: 0.0000 3rd Qu.: 0.0000 3rd Qu.: 0.0000   
 Max. :157.0000 Max. :51.0000 Max. :31.0000

## Point 2: Analyse des données manquantes

# Fonction pour calculer la proportion de valeurs manquantes par variable  
proportion\_valeurs\_manquantes <- function(data) {  
 # Calcul du nombre de valeurs manquantes par colonne  
 nb\_valeurs\_manquantes <- sapply(data, function(x) sum(is.na(x)))  
  
 # Calcul de la proportion de valeurs manquantes  
 proportion\_manquantes <- nb\_valeurs\_manquantes / nrow(data)  
  
 # Création d'un dataframe pour le résultat  
 resultat <- data.frame(Nombre = nb\_valeurs\_manquantes, Proportion = proportion\_manquantes)  
  
 return(resultat)  
}  
  
# Utilisation de la fonction avec votre base de données  
resultat <- proportion\_valeurs\_manquantes(base\_publication)  
resultat

Nombre Proportion  
status\_id 0 0  
status\_type 0 0  
num\_reactions 0 0  
num\_comments 0 0  
num\_shares 0 0  
num\_likes 0 0  
num\_loves 0 0  
num\_wows 0 0  
num\_hahas 0 0  
num\_sads 0 0  
num\_angrys 0 0

### Découverte d’un package VIM

# Charger le package VIM  
if (!require(VIM)) install.packages("VIM")

Loading required package: VIM

Loading required package: colorspace

Loading required package: grid

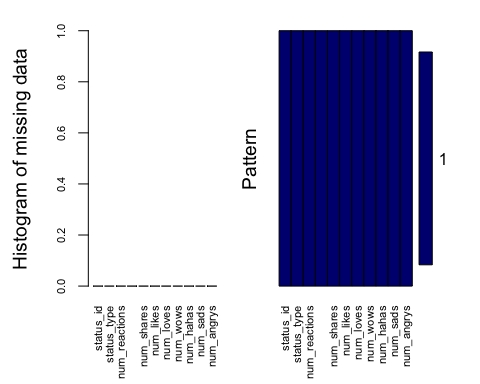
VIM is ready to use.

Suggestions and bug-reports can be submitted at: https://github.com/statistikat/VIM/issues

Attaching package: 'VIM'

The following object is masked from 'package:datasets':  
  
 sleep

library(VIM)  
  
# Utilisation de la fonction aggr() pour visualiser les valeurs manquantes  
# base\_publication : le dataframe contenant vos données  
# col : définit les couleurs utilisées dans le graphique (navyblue pour les valeurs existantes, yellow pour les manquantes)  
# numbers : si TRUE, affiche le pourcentage de valeurs manquantes pour chaque variable  
# sortVars : si TRUE, trie les variables par le taux de valeurs manquantes  
# labels : les étiquettes à utiliser pour chaque variable, ici on utilise les noms des colonnes de base\_publication  
# cex.axis : taille du texte des étiquettes d'axe (réduite ici à 0.7 pour une meilleure lisibilité)  
# gap : espace entre les barres dans l'histogramme  
# ylab : étiquettes pour l'axe des y, ici définies comme un histogramme des données manquantes et leur motif  
  
aggr(base\_publication, col=c('navyblue','yellow'), numbers=TRUE, sortVars=TRUE,   
 labels=names(base\_publication), cex.axis=.7, gap=3, ylab=c("Histogram of missing data","Pattern"))



Variables sorted by number of missings:   
 Variable Count  
 status\_id 0  
 status\_type 0  
 num\_reactions 0  
 num\_comments 0  
 num\_shares 0  
 num\_likes 0  
 num\_loves 0  
 num\_wows 0  
 num\_hahas 0  
 num\_sads 0  
 num\_angrys 0

## Points 3 et 4 : Analyse des outliers et description des données

Deux points:

-Analyses des variables qualitatives

-Analyse des variables quantitatives

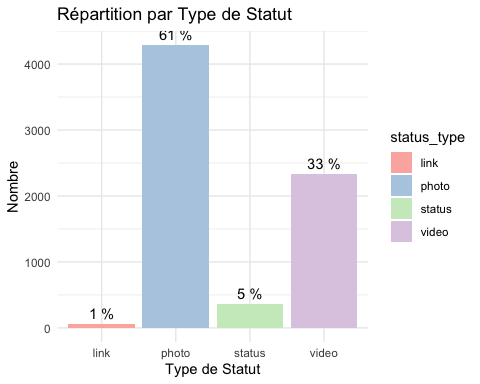
### Variable qualitative: type de statut

Deux analyses:

-Tableau des fréquences

-Barplot

# Chargement du package ggplot2 pour la visualisation  
library(ggplot2)  
  
# Création d'une table des fréquences pour la variable status\_type  
  
type\_statut\_table <- table(base\_publication$status\_type)  
  
# Conversion de la table en dataframe pour la visualisation  
type\_statut\_df <- as.data.frame(type\_statut\_table)  
  
# Renommage des colonnes  
names(type\_statut\_df) <- c("status\_type", "Count")  
  
# Ajout d'une colonne pour les proportions  
type\_statut\_df$Proportion <- type\_statut\_df$Count / sum(type\_statut\_df$Count)  
  
# Création du diagramme en barres  
ggplot(type\_statut\_df, aes(x = status\_type, y = Count, fill = status\_type)) +  
 geom\_bar(stat = "identity") +  
 geom\_text(aes(label = paste(round(Proportion \* 100), "%")), vjust = -0.5) +  
 labs(title = "Répartition par Type de Statut", x = "Type de Statut", y = "Nombre") +  
 theme\_minimal() +  
 scale\_fill\_brewer(palette = "Pastel1")



#Afficher le nombre  
type\_statut\_table

link photo status video   
 63 4288 365 2334

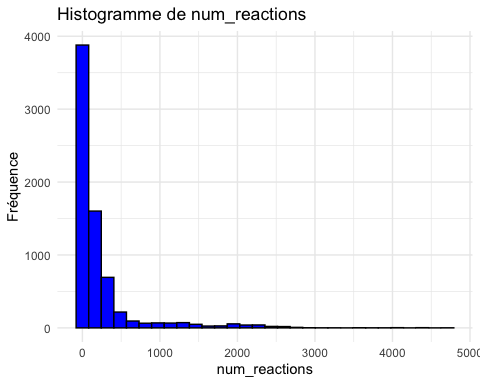
### Analyse des variables quantitatives

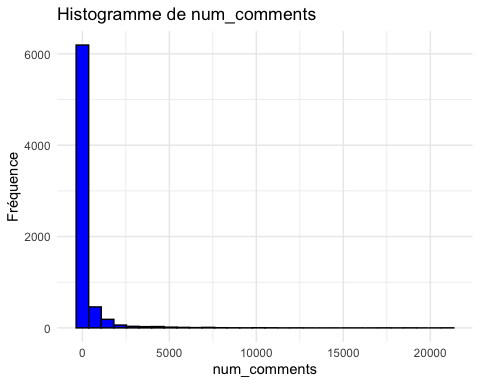
Deux graphiques à réaliser:

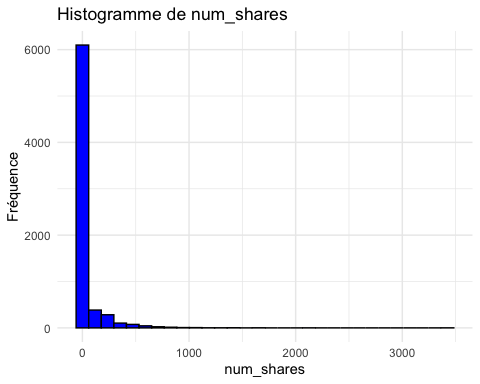
-Histogrammes

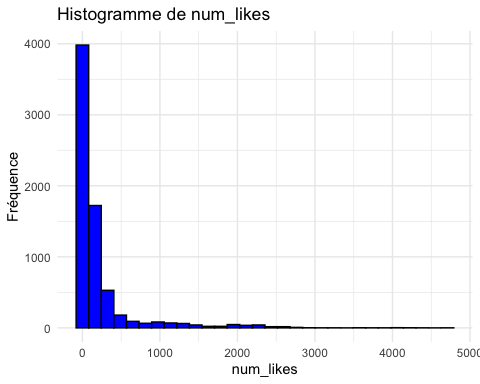
-Boxplot

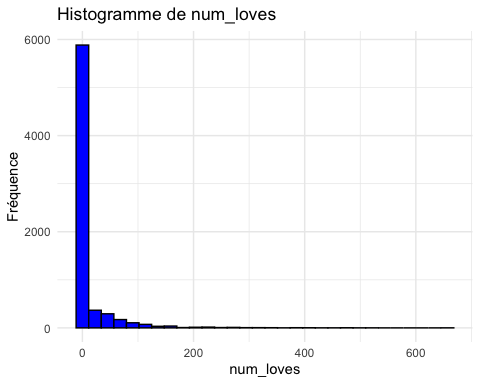
# Installer les packages nécessaires si ce n'est pas déjà fait  
if (!require(ggplot2)) install.packages("ggplot2")  
  
# Charger les packages  
library(ggplot2)  
# Identifier les colonnes quantitatives  
liste\_variable\_quanti <-colnames(base\_publication)[-c(1,2)]  
  
# Créer un histogramme pour chaque variable quantitative  
for (var in liste\_variable\_quanti) {  
 # Vérifier si la variable est continue  
 if (is.numeric(base\_publication[[var]])) {  
 print(ggplot(base\_publication, aes(x = .data[[var]])) +  
 geom\_histogram(bins = 30, fill = "blue", color = "black") +  
 theme\_minimal() +  
 labs(title = paste("Histogramme de", var), x = var, y = "Fréquence"))  
 } else {  
 message(paste("La variable", var, "n'est pas quantitative et sera ignorée."))  
 }  
}

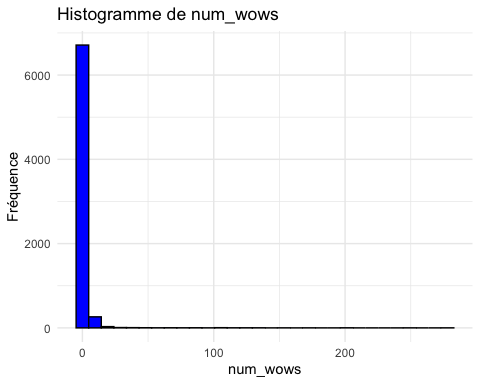




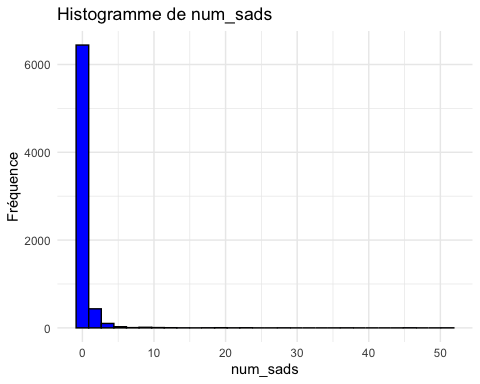






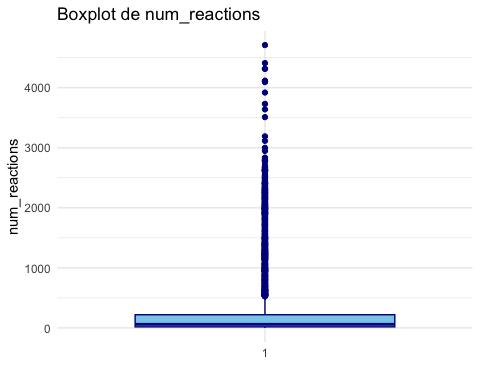


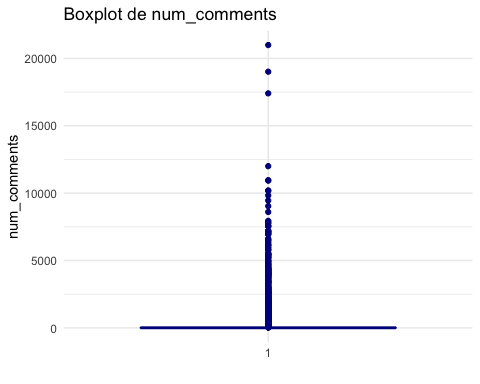


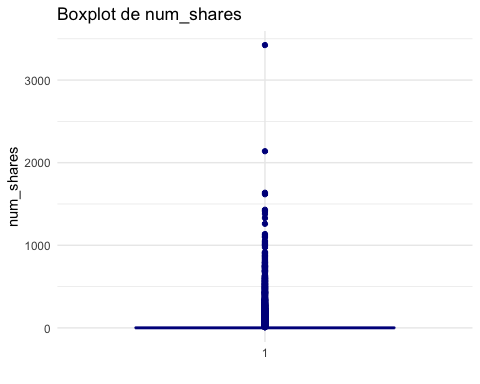


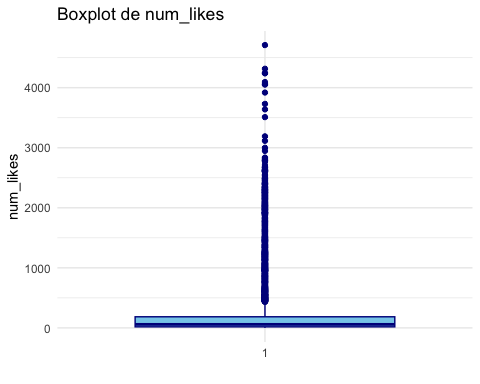


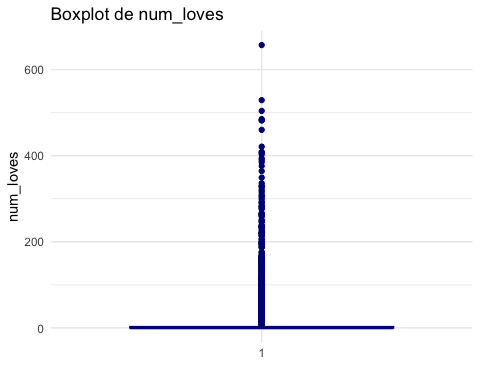
# Créer un boxplot pour chaque variable quantitative  
for (var in liste\_variable\_quanti) {  
 # Vérifier si la variable est continue  
 if (is.numeric(base\_publication[[var]])) {  
 print(ggplot(base\_publication, aes(x = factor(1), y = .data[[var]])) +  
 geom\_boxplot(fill = "skyblue", color = "darkblue") +  
 theme\_minimal() +  
 labs(title = paste("Boxplot de", var), x = "", y = var))  
 } else {  
 message(paste("La variable", var, "n'est pas quantitative et sera ignorée pour les boxplots."))  
 }  
}

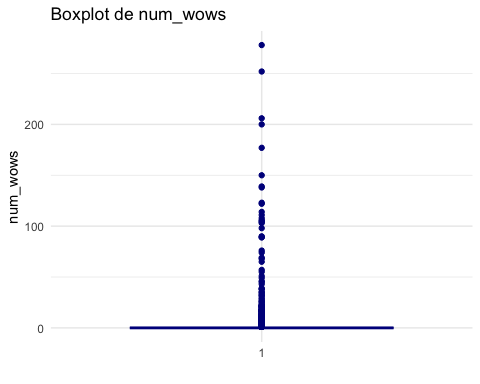




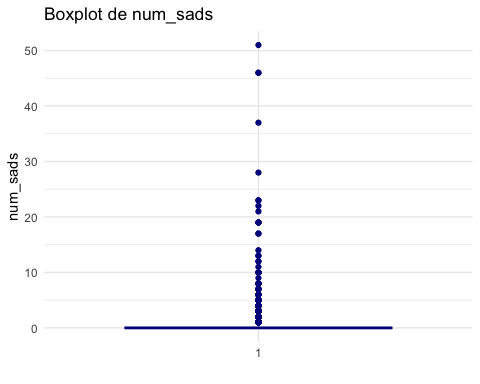


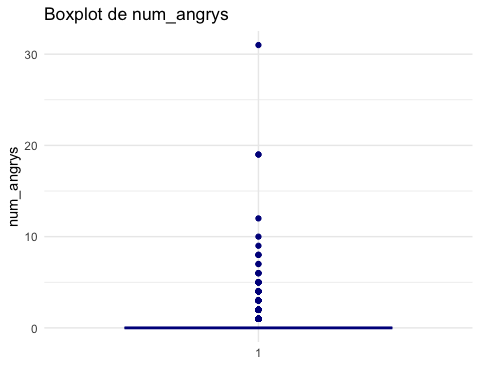












library(psych)

Attaching package: 'psych'

The following objects are masked from 'package:ggplot2':  
  
 %+%, alpha

describe(base\_publication[,liste\_variable\_quanti])

vars n mean sd median trimmed mad min max range  
num\_reactions 1 7050 230.12 462.63 59.5 112.55 77.84 0 4710 4710  
num\_comments 2 7050 224.36 889.64 4.0 35.21 5.93 0 20990 20990  
num\_shares 3 7050 40.02 131.60 0.0 7.06 0.00 0 3424 3424  
num\_likes 4 7050 215.04 449.47 58.0 100.85 74.13 0 4710 4710  
num\_loves 5 7050 12.73 39.97 0.0 2.95 0.00 0 657 657  
num\_wows 6 7050 1.29 8.72 0.0 0.24 0.00 0 278 278  
num\_hahas 7 7050 0.70 3.96 0.0 0.08 0.00 0 157 157  
num\_sads 8 7050 0.24 1.60 0.0 0.00 0.00 0 51 51  
num\_angrys 9 7050 0.11 0.73 0.0 0.00 0.00 0 31 31  
 skew kurtosis se  
num\_reactions 3.74 16.72 5.51  
num\_comments 9.03 126.74 10.60  
num\_shares 7.10 96.77 1.57  
num\_likes 3.92 18.41 5.35  
num\_loves 6.00 50.52 0.48  
num\_wows 18.24 415.18 0.10  
num\_hahas 20.30 586.60 0.05  
num\_sads 17.57 426.65 0.02  
num\_angrys 19.50 624.13 0.01

# Étape 2 : Analyse des relations entre les variables

* Mettre en place le pair plot (nuage de points).
* Mettre en place la heatmap de corrélation.
* Analyser les relations entre les variables qualitatives et quantitatives.
* Analyser les relations entre les variables qualitatives.

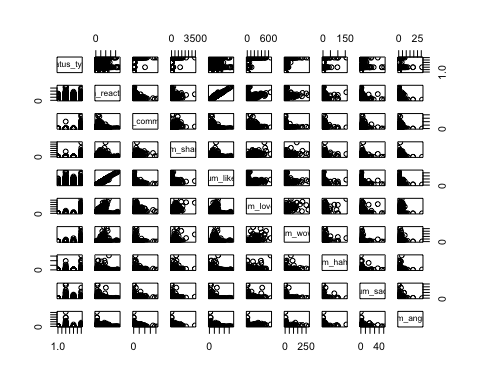
## Réaliser le pair plot entre les variables

# Chargement des packages nécessaires  
if (!require(GGally)) install.packages("GGally")

Loading required package: GGally

Registered S3 method overwritten by 'GGally':  
 method from   
 +.gg ggplot2

library(GGally)  
# Sélection des variables quantitatives  
base\_anal <- base\_publication[,-1]#Enlever l'identifiant de la base  
  
# Création du pairplot  
plot(base\_anal)

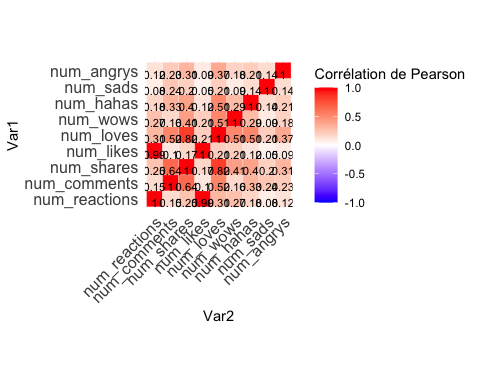


## Heatmap de corrélation entre les variables quantitatives

if (!require(ggplot2)) install.packages("ggplot2")  
if (!require(reshape2)) install.packages("reshape2") # Pour transformer la matrice de corrélation

Loading required package: reshape2

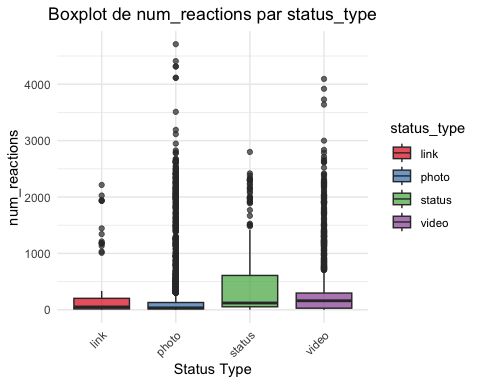
library(ggplot2)  
library(reshape2)  
base\_quati <- base\_publication[,-c(1,2)] # Enlever les colonnes non quantitatives  
matrice\_correlation <- cor(base\_quati, use = "complete.obs")  
  
# Fondre la matrice de corrélation pour une utilisation avec ggplot2  
melted\_cormat <- melt(matrice\_correlation)  
  
  
ggplot(data = melted\_cormat, aes(Var2, Var1, fill = value)) +  
 geom\_tile(color = "white") +  
 geom\_text(aes(label = round(value, 2)), vjust = 1, size = 3) + # Ajout de geom\_text avec une taille de police augmentée  
 scale\_fill\_gradient2(low = "blue", high = "red", mid = "white",   
 midpoint = 0, limit = c(-1,1), space = "Lab",   
 name="Corrélation de Pearson") +  
 theme\_minimal() +   
 theme(axis.text.x = element\_text(angle = 45, vjust = 1,   
 size = 12, hjust = 1),  
 axis.text.y = element\_text(size = 12)) +  
 coord\_fixed()

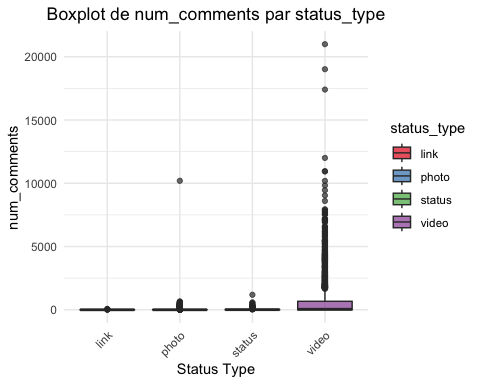


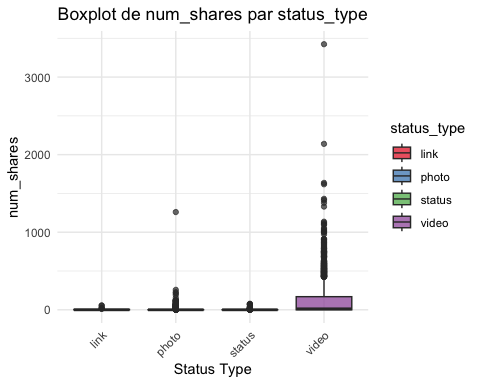
## Relation variable qualitative et une variable quantitative

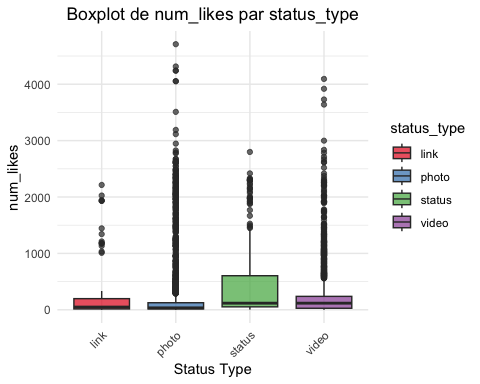
* Mettre en place les boxplots bivariés.
* Réaliser les tests statistiques adaptés (Wilcoxon, Kruskal-Wallis, ANOVA, Student, Welch).

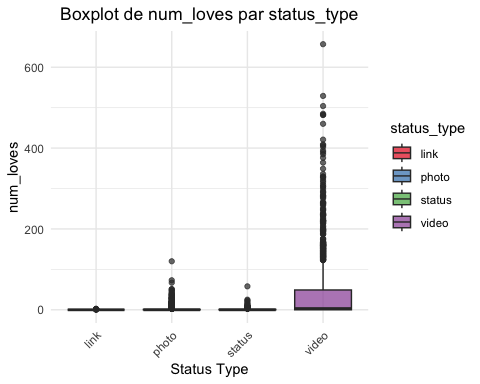
if (!require(ggplot2)) install.packages("ggplot2")  
  
# Charger les packages  
library(ggplot2)  
  
# Pour chaque variable quantitative, créer un boxplot par rapport à status\_type  
for (col in liste\_variable\_quanti) {  
 # Créer le boxplot  
 p <- ggplot(base\_publication, aes(x = status\_type, y = .data[[col]], fill = status\_type)) +  
 geom\_boxplot(alpha = 0.7) + # Supprimer la bordure noire  
 scale\_fill\_brewer(palette = "Set1") + # Utiliser une palette de couleurs vive  
 labs(title = paste("Boxplot de", col, "par status\_type"), x = "Status Type", y = col) +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1), # Incliner le texte de l'axe X  
 plot.title = element\_text(hjust = 0.5)) # Centrer le titre  
  
 # Afficher le boxplot  
 print(p)  
}

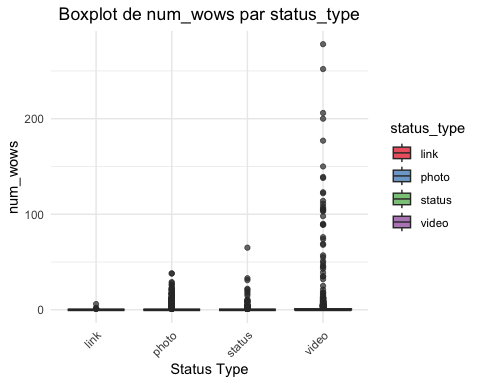


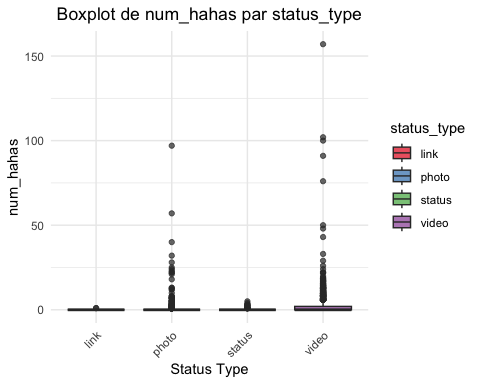


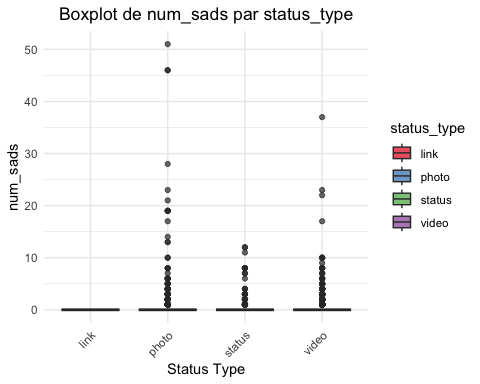


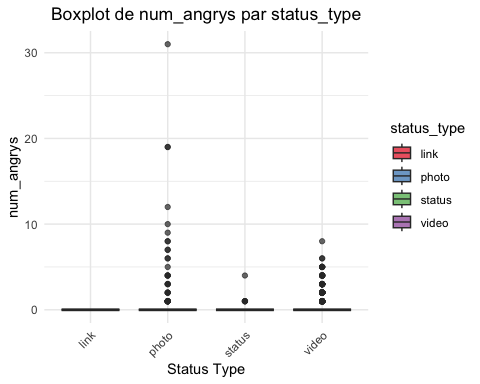












# Effectuer le test de Wilcoxon pour chaque variable quantitative par rapport à une variable qualitative  
  
resultats\_tests <- list()  
  
for (col in liste\_variable\_quanti) {  
 # Exécution du test de Kruskal-Wallis  
 test <- kruskal.test(as.formula(paste(col, '~ status\_type')), data = base\_publication)  
   
 # Stocker la statistique de test et la p-valeur  
 resultats\_tests[[col]] <- c(statistic = test$statistic, p.value = test$p.value)  
}  
  
# Conversion de la liste des résultats en dataframe  
resultats\_dataframe <- as.data.frame(t(sapply(resultats\_tests, c)))  
  
# Nommer les colonnes et les lignes du dataframe  
colnames(resultats\_dataframe) <- c("statistic", "p.value")  
rownames(resultats\_dataframe) <- liste\_variable\_quanti  
  
resultats\_dataframe

statistic p.value  
num\_reactions 430.0462 6.860611e-93  
num\_comments 748.0684 7.916056e-162  
num\_shares 1539.8307 0.000000e+00  
num\_likes 357.6951 3.216644e-77  
num\_loves 883.4864 3.379917e-191  
num\_wows 196.5584 2.337563e-42  
num\_hahas 1336.2606 1.994865e-289  
num\_sads 511.4759 1.554607e-110  
num\_angrys 643.2248 4.288563e-139

# Etape 3: Réaliser l’ACP

2 points clés:

-Choix du nombre d’axe factoriel

-Interprétation des axes

## 1 - Choisir le nombre d’axe factoriel

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

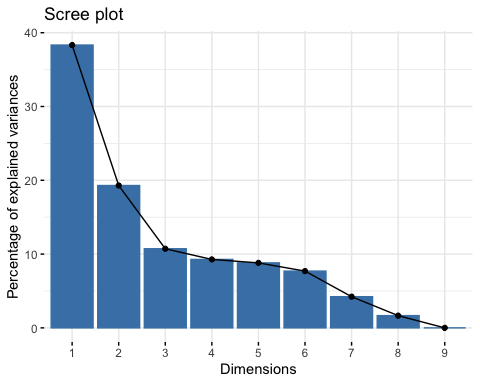
Loading required package: FactoMineR

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

Loading required package: factoextra

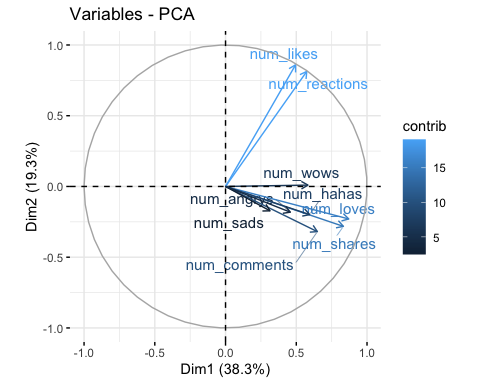
Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa

library(FactoMineR)  
library(factoextra)  
  
# Sélection des variables quantitatives  
base\_quanti <- base\_publication[,-c(1,2)] # Exclure les colonnes non quantitatives  
  
# Variable qualitative comme supplémentaire  
status\_type <- base\_publication$status\_type  
  
# Normalisation (centrage et réduction) des variables quantitatives  
base\_quanti\_norm <- scale(base\_quanti)  
acp <- PCA(base\_quanti\_norm, graph = FALSE)  
fviz\_eig(acp)

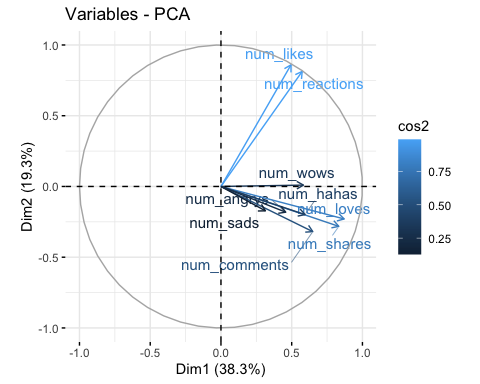


## 2- Interpréter les axes

# Cercle de corrélation  
fviz\_pca\_var(acp, col.var = "contrib", repel = TRUE) # Colorer par contribution



fviz\_pca\_var(acp, col.var = "cos2", repel = TRUE) # Colorer par contribution



# Mise en place des kmeans

## Etape 4: Détermination du nombre de cluster

2 méthodes pour déterminer le nombre de groupe

* Méthode du coude
* score de silhouette

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

Loading required package: cluster

library(factoextra)  
library(cluster)

### Méthode 1: Méthode du coude

L’idée de base derrière les méthodes de partitionnement de clusters, telles que le clustering k-means, est de définir des clusters afin de **minimiser** la **variation intra-cluster totale.** Cette variation est connue sous le nom de variation totale intra-cluster ou somme totale des carrés intra-cluster (WSS). La formule pour minimiser la WSS est la suivante :

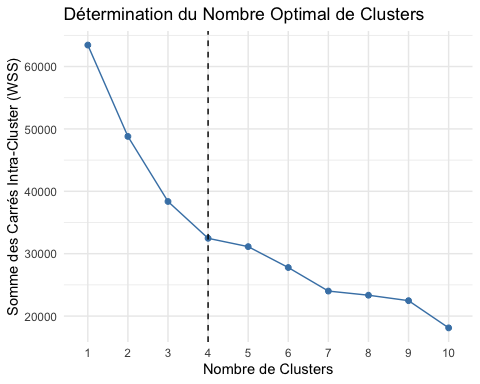
Dans cette formule, représente le k-ième cluster et $W(C\_k) $est la variation intra-cluster. La WSS quantifie la densité du regroupement et l’objectif est de la réduire au minimum possible.

Pour déterminer le nombre optimal de clusters, on peut utiliser l’algorithme suivant :

1. Appliquer l’algorithme de clustering pour différentes valeurs de k, par exemple, en faisant varier k de 1 à 10 clusters.
2. Calculer la WSS pour chaque valeur de k.
3. Tracer la courbe de WSS en fonction du nombre de clusters k.
4. Identifier l’emplacement d’un coude dans le graphique, qui est généralement considéré comme un indicateur du nombre approprié de clusters.

<https://uc-r.github.io/kmeans_clustering#optimal>

# Détermination du nombre optimal de clusters avec la méthode du coude  
fviz\_nbclust(base\_quanti\_norm, kmeans, method = "wss") +   
 geom\_vline(xintercept = 4, linetype = 2) + # Ajuster le xintercept selon le résultat  
 labs(title = "Détermination du Nombre Optimal de Clusters",  
 x = "Nombre de Clusters",  
 y = "Somme des Carrés Intra-Cluster (WSS)") +  
 theme\_minimal()



### Méthode 2: Méthode de silhouette

L’approche de la silhouette détermine **à quel point chaque objet est bien intégré dans son cluster**. Une largeur moyenne de silhouette élevée indique un bon clustering. La méthode de la silhouette moyenne calcule la silhouette moyenne des observations pour différentes valeurs de k. Le nombre optimal de clusters k est celui qui maximise la silhouette moyenne sur une plage de valeurs possibles pour k.

<https://uc-r.github.io/kmeans_clustering>

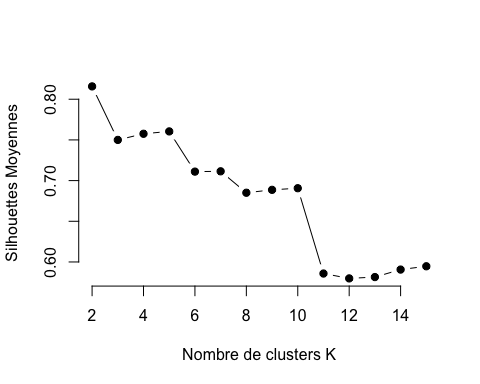
if (!require(purrr)) install.packages("purrr")

Loading required package: purrr

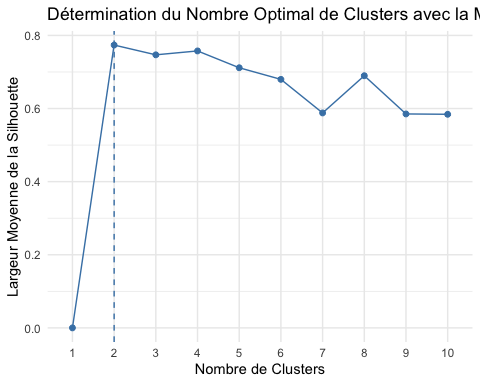
library(purrr)  
  
# function to compute average silhouette for k clusters  
avg\_sil <- function(k) {  
 km.res <- kmeans(base\_quanti\_norm, centers = k, nstart = 25)  
 ss <- silhouette(km.res$cluster, dist(base\_quanti\_norm))  
 mean(ss[, 3])  
}  
  
# Compute and plot wss for k = 2 to k = 15  
k.values <- 2:15  
  
# extract avg silhouette for 2-15 clusters  
avg\_sil\_values <- map\_dbl(k.values, avg\_sil)

Warning: did not converge in 10 iterations  
  
Warning: did not converge in 10 iterations  
  
Warning: did not converge in 10 iterations  
  
Warning: did not converge in 10 iterations  
  
Warning: did not converge in 10 iterations

plot(k.values, avg\_sil\_values,  
 type = "b", pch = 19, frame = FALSE,   
 xlab = "Nombre de clusters K",  
 ylab = "Silhouettes Moyennes")

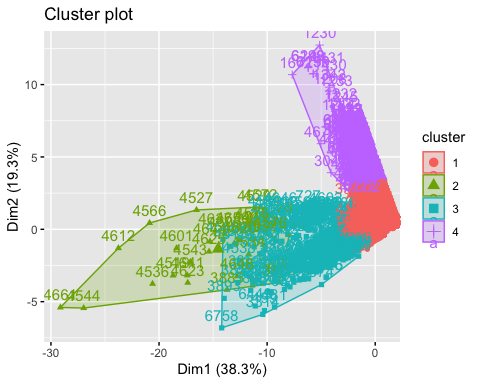


fviz\_nbclust(base\_quanti\_norm, kmeans, method = "silhouette") +  
 labs(title = "Détermination du Nombre Optimal de Clusters avec la Méthode de la Silhouette",  
 x = "Nombre de Clusters",  
 y = "Largeur Moyenne de la Silhouette") +  
 theme\_minimal()



set.seed(123) # Pour la reproductibilité  
kmeans\_result <- kmeans(base\_quanti\_norm, centers = 4, nstart = 25)

fviz\_cluster(kmeans\_result, data = base\_quanti\_norm)

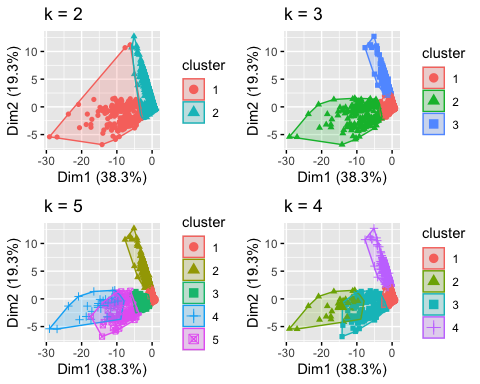


set.seed(123) # Pour la reproductibilité  
k2 <- kmeans(base\_quanti\_norm, centers = 2, nstart = 25)  
k3 <- kmeans(base\_quanti\_norm, centers = 3, nstart = 25)  
k5 <- kmeans(base\_quanti\_norm, centers = 5, nstart = 25)  
  
# plots to compare  
p1 <- fviz\_cluster(k2, geom = "point", data = base\_quanti\_norm) + ggtitle("k = 2")  
p2 <- fviz\_cluster(k3, geom = "point", data = base\_quanti\_norm) + ggtitle("k = 3")  
p3 <- fviz\_cluster(k5, geom = "point", data = base\_quanti\_norm) + ggtitle("k = 5")  
p4 <- fviz\_cluster(kmeans\_result, geom = "point", data = base\_quanti\_norm) + ggtitle("k = 4")  
  
library(gridExtra)

Attaching package: 'gridExtra'

The following object is masked from 'package:dplyr':  
  
 combine

grid.arrange(p1, p2, p3,p4, nrow = 2)

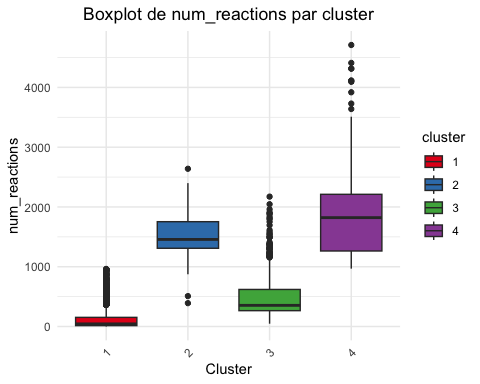


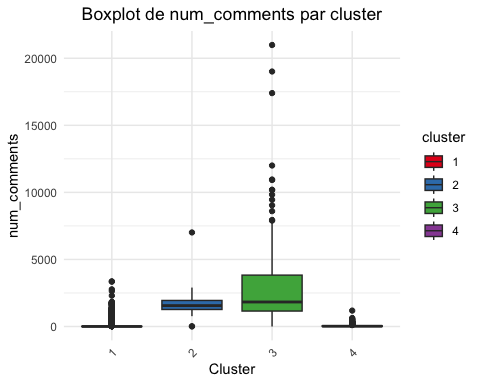
# Etape 5: Interprétation des résultats

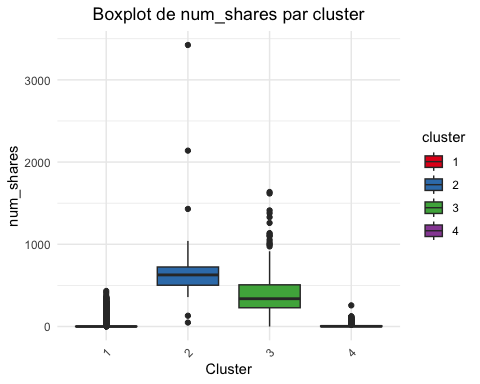
Source code: <https://www.datanovia.com/en/blog/beautiful-radar-chart-in-r-using-fmsb-and-ggplot-packages/>

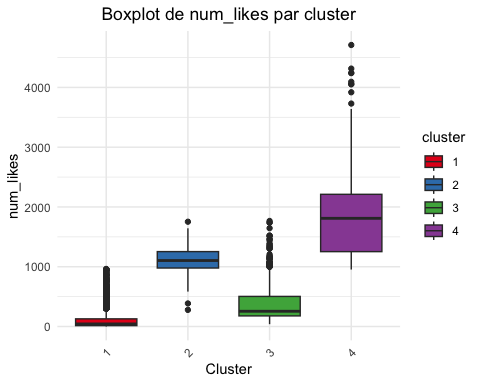
# Ajouter la colonne de cluster à base\_publication  
base\_publication$cluster <- factor(kmeans\_result$cluster)

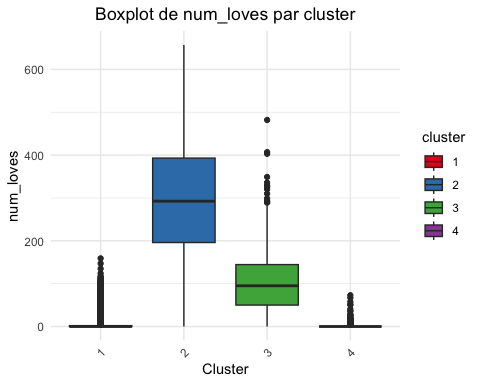
# Pour chaque variable quantitative, créer un boxplot par rapport à cluster  
for (col in liste\_variable\_quanti) {  
 p <- ggplot(base\_publication, aes(x = cluster, y = .data[[col]], fill = cluster)) +  
 geom\_boxplot() +  
 scale\_fill\_brewer(palette = "Set1") + # Utiliser une palette de couleurs  
 labs(title = paste("Boxplot de", col, "par cluster"), x = "Cluster", y = col) +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1), # Incliner le texte de l'axe X  
 plot.title = element\_text(hjust = 0.5)) # Centrer le titre  
  
 # Afficher le boxplot  
 print(p)  
}

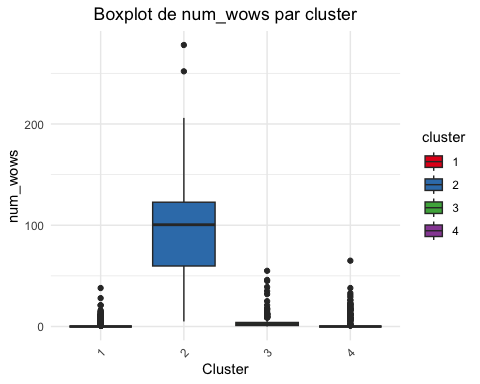




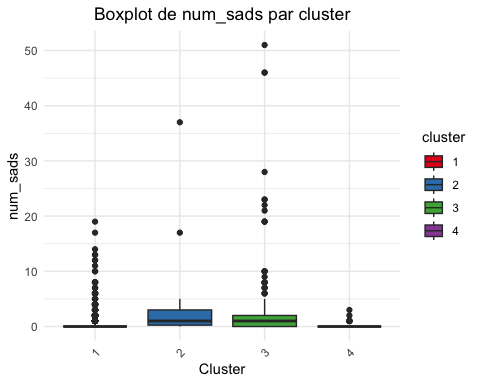


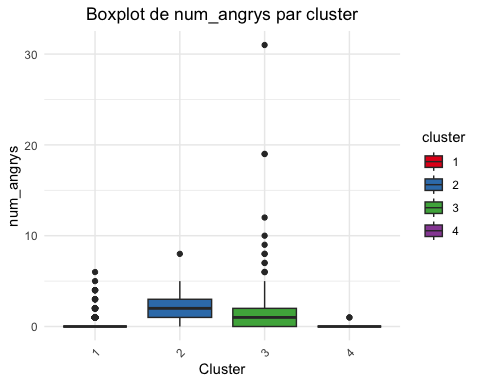




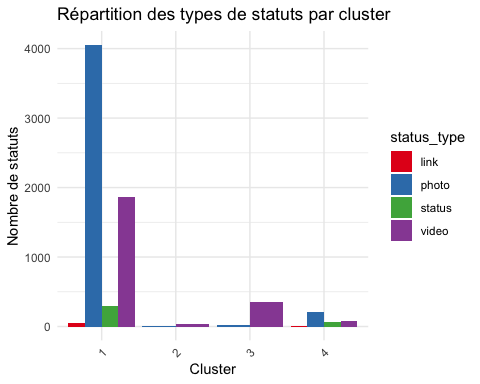








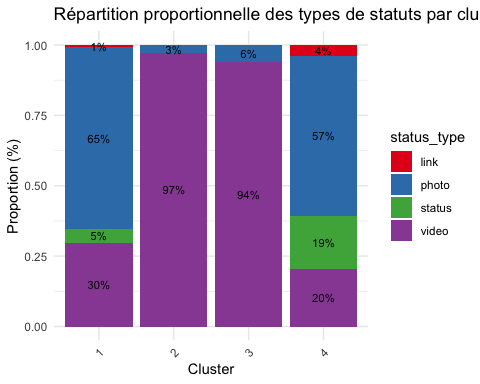
ggplot(base\_publication, aes(x = factor(cluster), fill = status\_type)) +  
 geom\_bar(position = "dodge") +  
 scale\_fill\_brewer(palette = "Set1") + # Utiliser une palette de couleurs  
 labs(title = "Répartition des types de statuts par cluster",  
 x = "Cluster",  
 y = "Nombre de statuts") +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1)) # Incliner le texte de l'axe X



base\_groupe <- base\_publication %>%  
 group\_by(cluster, status\_type) %>%  
 summarise(count = n(), .groups = 'drop') %>%  
 group\_by(cluster) %>%  
 mutate(freq = count / sum(count)) %>%  
 ungroup()  
base\_groupe

# A tibble: 12 × 4  
 cluster status\_type count freq  
 <fct> <chr> <int> <dbl>  
 1 1 link 49 0.00782  
 2 1 photo 4052 0.647   
 3 1 status 295 0.0471   
 4 1 video 1869 0.298   
 5 2 photo 1 0.0294   
 6 2 video 33 0.971   
 7 3 photo 23 0.0607   
 8 3 video 356 0.939   
 9 4 link 14 0.0376   
10 4 photo 212 0.570   
11 4 status 70 0.188   
12 4 video 76 0.204

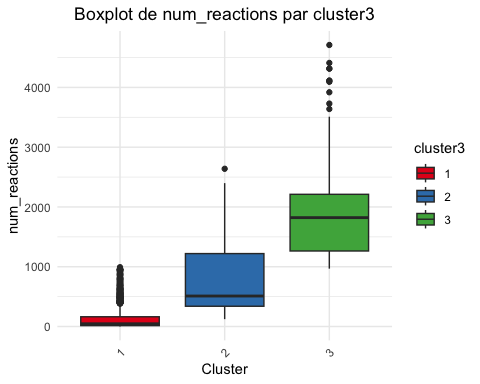
ggplot(base\_groupe, aes(x = factor(cluster), y = freq, fill = status\_type)) +  
 geom\_bar(stat = "identity", position = "fill") +  
 geom\_text(aes(label = scales::percent(freq, accuracy = 1)),   
 position = position\_fill(vjust = 0.5),   
 size = 3,   
 color = "black") +  
 scale\_fill\_brewer(palette = "Set1") + # Utiliser une palette de couleurs  
 labs(title = "Répartition proportionnelle des types de statuts par cluster",  
 x = "Cluster",  
 y = "Proportion (%)") +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1)) # Incliner le texte de l'axe X

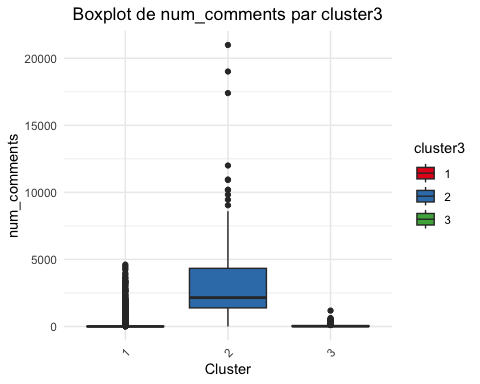


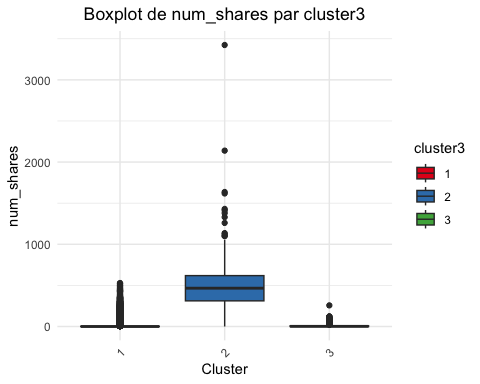
## Interprétation avec 3 clusters

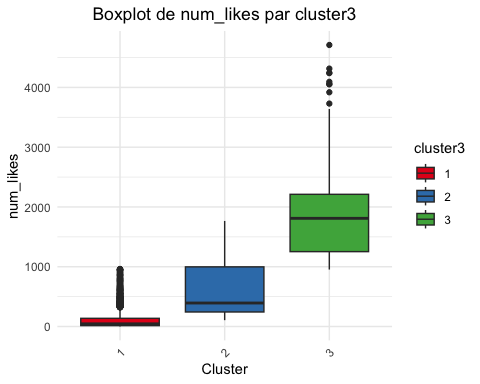
base\_publication$cluster3 <- factor(k3$cluster)

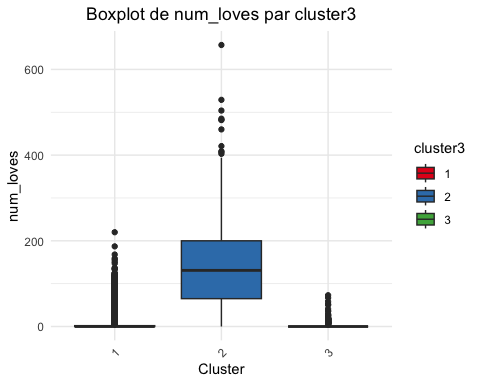
# Pour chaque variable quantitative, créer un boxplot par rapport à cluster  
for (col in liste\_variable\_quanti) {  
 p <- ggplot(base\_publication, aes(x = cluster3, y = .data[[col]], fill = cluster3)) +  
 geom\_boxplot() +  
 scale\_fill\_brewer(palette = "Set1") + # Utiliser une palette de couleurs  
 labs(title = paste("Boxplot de", col, "par cluster3"), x = "Cluster", y = col) +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1), # Incliner le texte de l'axe X  
 plot.title = element\_text(hjust = 0.5)) # Centrer le titre  
  
 # Afficher le boxplot  
 print(p)  
}

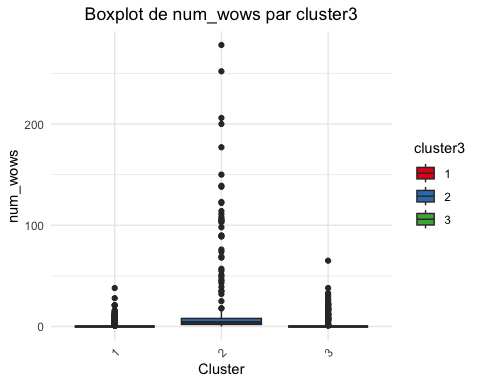


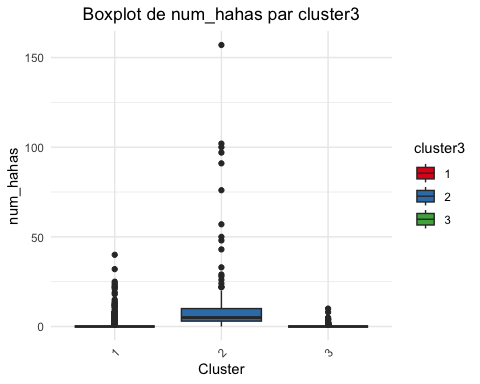


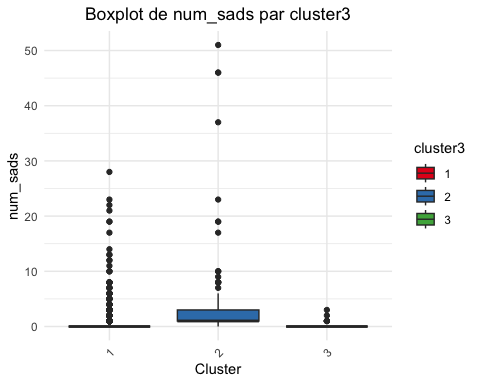


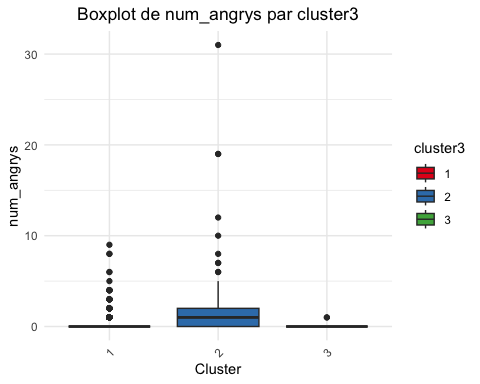












library(dplyr)  
#install.packages("tidyverse")  
library(tidyverse)

── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
✔ forcats 1.0.0 ✔ stringr 1.5.1  
✔ lubridate 1.9.3 ✔ tibble 3.2.1  
✔ readr 2.1.4 ✔ tidyr 1.3.0  
── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
✖ psych::%+%() masks ggplot2::%+%()  
✖ psych::alpha() masks ggplot2::alpha()  
✖ gridExtra::combine() masks dplyr::combine()  
✖ dplyr::filter() masks stats::filter()  
✖ dplyr::lag() masks stats::lag()  
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

# Calculer les proportions  
base\_groupe <- base\_publication %>%  
 group\_by(cluster3, status\_type) %>%  
 summarise(count = n(), .groups = 'drop') %>%  
 group\_by(cluster3) %>%  
 mutate(freq = count / sum(count)) %>%  
 ungroup()  
base\_groupe

# A tibble: 10 × 4  
 cluster3 status\_type count freq  
 <fct> <chr> <int> <dbl>  
 1 1 link 49 0.00765  
 2 1 photo 4061 0.634   
 3 1 status 295 0.0461   
 4 1 video 2000 0.312   
 5 2 photo 15 0.0549   
 6 2 video 258 0.945   
 7 3 link 14 0.0376   
 8 3 photo 212 0.570   
 9 3 status 70 0.188   
10 3 video 76 0.204

ggplot(base\_groupe, aes(x = cluster3, y = freq, fill = status\_type)) +  
 geom\_bar(stat = "identity", position = "fill") +  
 geom\_text(aes(label = scales::percent(freq, accuracy = 1)),   
 position = position\_fill(vjust = 0.5),   
 size = 3,   
 color = "black") +  
 scale\_fill\_brewer(palette = "Set1") + # Utiliser une palette de couleurs  
 labs(title = "Répartition proportionnelle des types de statuts par cluster",  
 x = "Cluster",  
 y = "Proportion (%)") +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1)) # Incliner le texte de l'axe X

