Master 1 – Informatique
Analyse et visualisation des données

Sujet TP4

Contexte

Le TP4 porte sur l'étude décrite au TP3.

L'objectif du TP4 est de réaliser des graphiques à partir du jeu de données **124**, désigné **france_eaurob** relatif à l'analyse de l'eau du robinet en France disponible sur `https://pdh.cnrs.fr/fr/`. Vous allez apprendre à manipuler et visualiser des données géographiques en R.

Exercice 1 : Graphique nuage de mots wordcloud()

L'objectif de cet exercice est de réaliser un graphique nuage de mots afin de mettre en évidence le nom des substances détectées. Pour cela, il vous est conseillé de créer un tableau de deux colonnes : les substances et leur nombre d'occurrence qui correspondront respectivement aux paramètres **words** et **freq** de la fonction **wordcloud()**.

Travail à faire : graphique Nuage de mots

1. Créer un tableau **table** des fréquences des valeurs uniques présentes dans la colonne pfas_values de data2. Ne prendre que les valeurs supérieures à 1.
2. Convertir ce tableau en un data frame **as.data.frame()**
3. Renommer les colonnes avec **colnames** par "substance" et "n"
4. La substance la plus fréquente doit être au centre du graphique.
5. Choisir une palette de couleur adaptée.
6. Ajouter une légende **title(main = "Substances détectées")**

Exercice 2 : Créer une carte interactive de la France

L'objectif de cet exercice est de générer une carte interactive de la France avec des points représentant les lieux de prélèvement contenant des PFAS. Les points seront affichés sous forme de punaises avec un message popup contenant le nom de la ville et la somme des PFAS (pfas-sum).

Travail à faire :

1. Créer une carte interactive centrée sur la France avec :
   - les points des villes sous forme de punaise,
   - une popup avec le nom de la ville et la *pfas_sum*.
2. Enregistrer la carte au format HTML avec la commande **saveWidget()**
3. Ouvrir la carte dans un navigateur avec la commande **browseURL()**
   Que remarquez vous ? Pourquoi Dijon n'apparaît pas ?

Exercice 3 : Mise à jour des données

Dans l'exercice précédent, vous constatez que des villes apparaissent dans le golfe de Guinée, au large des côtes ouest d'Afrique, dans l'hémisphère sud. Elles comportent donc des latitudes négatives. D'autre part, Dijon n'apparaît pas sur la carte car ses coordonnées géographiques ne figurent pas dans le fichier .csv. L'objectif de cet exercice est de corriger ces données.

Travail à faire :

1. Afficher le nom des villes dont les latitudes sont négatives.
2. A partir du site `https://www.coordonnees-gps.fr`, trouver les coordonnées géographiques de Dijon.
3. Créer un nouveau dataset **data3** avec :
   - les coordonnées géographiques de Dijon,
   - uniquement les lignes avec des coordonnées géographiques valides (supprimer les coordonnées NA ou latitudes négatives).
4. Afficher la carte du dataset **data3** dans un navigateur .

Utiliser data3 pour les exercices suivants.

Exercice 4 : Personnaliser la carte

L'objectif de cet exercice est de vous familiariser avec les éléments de mise en forme et de personnalisation de la carte interactive.

Travail à faire :

- Modifier la carte avec 4 couleurs différentes pour les punaises selon la valeur de pfas_sum :
  - Rouge → Valeurs les plus élevées.
  - Violet clair → Valeurs moyennes-hautes.
  - Bleu → Valeurs moyennes-basses.

  - Gris → Valeurs faibles.
- Ajouter une légende pour expliquer ces couleurs.
- Ajouter le titre : "PFAS détectés de 2015 à 2024".

Exercice 5 : Carte choroplèthe des concentrations de PFAS par régions de France

L'objectif de cet exercice est de créer une carte choroplèthe interactive représentant la moyenne des concentrations de PFAS par région française. La carte devra afficher :

- Les régions colorées selon la moyenne des concentrations de PFAS.
- Des popup indiquant :
  - le nom de la région,
  - la moyenne des PFAS,
  - le nombre de prélèvements réalisés.
- Une légende pour les différentes couleurs.
- Un titre.

Travail à faire :

1. Préparation de l'environnement
   - Charger les packages nécessaires (leaflet, sf, dplyr, htmlwidgets).
2. Importer les données
   - Charger le fichier GeoJSON des régions de France.
   - Importer le fichier .csv de data3
   - Gérer les valeurs manquantes : si pfas_sum = NA alors on remplace par 0
3. Convertir les données des prélèvements en objets géospatiaux (sf).
4. Analyse géospatiale
   - Associer chaque prélèvement à sa région (st_join).
   - Pour chaque région, calculer la moyenne des concentrations de PFAS et le nombre total de prélèvements.
   - Fusionner les statistiques avec les contours des régions (jointure tabulaire)
5. Création de la carte choroplèthe avec leaflet :
   - Colorier les régions selon la moyenne de PFAS avec une palette dégradée.
   - Ajouter des labels au survol des régions avec les statistiques calculées.
   - Afficher une légende pour expliquer les différentes couleurs.
   - Ajouter un titre.
6. Enregistrer et affichager
   - Enregistrer la carte dans un fichier HTML.
   - Ouvrir automatiquement la carte dans un navigateur.

INDICATIONS

Quelques packages de R :

- library(ggplot2) → créer des graphiques
- library(RColorBrewer) → gérer des palettes de couleurs pour les graphiques
- library(tidyr) → nettoyer les données
- library(purrr) → manipuler liste et vecteur (map)
- library(leaflet) → créer des cartes interactives.
- library(sf) → manipuler des données géospatiales.
- library(dplyr) → traiter les données.
- library(htmlwidgets) → sauvegarder avec HTML.

Format HTML

Enregistrer la carte France au format HTML
saveWidget(carte, "France.html", selfcontained = TRUE)
Ouvrir la carte dans le navigateur
browseURL("France.html")

NUAGE DE MOTS

La fonction wordcloud() est utilisée pour générer des nuages de mots dans R, où la taille de chaque mot est proportionnelle à sa fréquence d'apparition dans un jeu de données.

La fonction wordcloud() a besoin d'un vecteur de mots et d'un vecteur de fréquences pour fonctionner. Les deux vecteurs doivent avoir la même longueur et être alignés (chaque fréquence doit correspondre à son mot respectif).

Syntaxe de base

```
wordcloud(words, freq)
```

avec :
words : Vecteur des mots à afficher, exemple : words <- c("R", "Data", "Science")
freq : Vecteur des fréquences associées aux mots, exemple : freq <- c(15, 12, 8)

Syntaxe complémentaire

- min.freq → Fréquence minimale pour afficher un mot. (Par défaut = 3)
- max.words → Nombre maximum de mots à afficher. (Par défaut = 200)
- random.order → Si TRUE, les mots sont affichés aléatoirement. Sinon, les plus fréquents sont au centre. (Par défaut = TRUE)

- rot.per → Proportion de mots affichés verticalement. (Par défaut = 0.1 — 10
- scale → Taille maximale et minimale des mots. (Ex: c(4, 0.5) — mots les plus fréquents sont 4x plus grands que les moins fréquents)
- colors → Palette de couleurs pour les mots. (Ex: brewer.pal(8, "Dark2"))

CARTE INTERACTIVE

leaflet est un package R qui permet de créer des cartes interactives.

Syntaxe de base

```
leaflet(data) %>%
  addTiles() %>%
  addMarkers(lng = ~lon, lat = ~lat, popup = ~info) %>%
  setView(lng = <longitude>, lat = <latitude>, zoom = <niveau_de_zoom>)
```

- leaflet(data) : Initialise la carte avec un jeu de données (optionnel).
- addTiles() : Ajoute le fond de carte (OpenStreetMap par défaut).
- addMarkers() : Place des marqueurs sur la carte.
- setView() : Centre la carte sur une localisation spécifique.

Syntaxe complémentaire

- leaflet(data) → Initialise la carte avec ou sans données.
- addTiles() → Ajoute un fond de carte (OpenStreetMap par défaut).
- addProviderTiles() → Ajoute des fonds de carte spécifiques (Stamen, CartoDB, etc.).
- addMarkers() → Place des marqueurs simples.
- addCircleMarkers() → Marqueur circulaire (de la forme d'une punaise), taille et couleur personnalisables.
- addPolygons() → Dessine des polygones (par ex. contours géographiques).
- addPolylines() → Dessine des lignes (par ex. routes, trajets).
- addPopups() → Ajoute des popups fixes sur la carte.
- addLegend() → Ajoute une légende à la carte.
- fitBounds() → Ajuste la vue pour inclure tous les marqueurs.
- setView() → Centre la carte et définit le niveau de zoom.
- setView(lng = 2.2137, lat = 46.603354) : Centre la carte sur la France (longitude 2.2137, latitude 46.603354).

Marqueurs avec label et popup

Exemple illustrant l'utilisation de addMarkers() dans leaflet pour ajouter un marqueur sur Paris dont la longitude est 2.3522 et la latitude est 48.8566. Lorsqu'on survole le marqueur, le nom de la ville "Paris" s'affiche (c'est le label) et lorsqu'on clique dessus le message "Plus belle ville du monde !" s'affiche (c'est le popup).

```
library(leaflet)
leaflet() %>%
  addTiles() %>%  # Fond de carte OpenStreetMap
  addMarkers(
    lng = 2.3522,
    lat = 48.8566,
    label = "Paris"
    popup = "Plus belle ville du monde !"
  )
```

Marqueurs colorés

La fonction getColor permet d'ajouter des marqueurs colorés en fonction d'une catégorie ou d'une valeur numérique. Exemple illustrant une carte avec 4 couleurs différentes pour les marqueurs selon la valeur de pfas_sum :

```
getColor <- function(pfas_sum) {
  if (pfas_sum >= 50) {
    return("red")    # Rouge : valeurs élevées
  } else if (pfas_sum >= 20) {
    return("violet") # Violet clair : moyennes - hautes
  } else if (pfas_sum >= 5) {
    return("blue")   # Bleu : moyennes - basses
  } else {
    return("gray")   # Gris : faibles
  }
}
```

Remarque : En R, la dernière ligne exécutée dans un bloc de code est automatiquement renvoyée comme résultat même si return() n'est pas mentionné. On peut donc écrire :

```
getColor <- function(pfas_sum) {
  if (pfas_sum >= 50) {
    ("red")
 } else if (pfas_sum >= 20) {
....
....
```

Personnaliser des éléments HTML avec addControl()

La fonction addControl() dans leaflet permet d'ajouter des éléments HTML sur une carte interactive, comme des titres, des descriptions, des logos, des boutons, ou des instructions.

Syntaxe
addControl(html, position = "topright", layerId = NULL, className = "")

- html (obligatoire) Contenu HTML à afficher (texte, images, boutons, etc.).
- position (obligatoire) Position du contrôle sur la carte avec "topleft", "topright", "bottomleft", "bottomright".

- layerId (optionnel) Identifiant unique pour le contrôle (utile pour le modifier ou le retirer).
- className (optionnel) Classe CSS personnalisée pour styliser le contrôle.

CARTE CHOROPLETHE

Régions de France

Pour obtenir l'identification des régions de France, il est conseillé d'utiliser un fichier GeoJSON contenant les contours géographiques des régions. Une source fiable est le dépôt GitHub de Grégoire David, accessible via l'URL suivante : `https://raw.githubusercontent.com/gregoiredavid/france-geojson/master/regions.geojson` Ce fichier GeoJSON, compatible avec les packages R (sf, leaflet), contient les polygones des régions ainsi que leurs attributs (nom, code). Il utilise le système de coordonnées EPSG:4326 (WGS 84), standard pour les cartes interactives. En R, l'import se fait facilement avec la commande : regions <- st_read("https://raw.githubusercontent.com/ gregoiredavid/france-geojson/master/regions.geojson")

Convertir les données des prélèvements en objets géospatiaux.

Pour pouvoir réaliser des opérations géospatiales (comme associer des points aux régions), il est nécessaire de transformer les données tabulaires du dataset en un objet géospatial à l'aide de la fonction **sf** (Simple Features).

Fusion des données résumées avec la carte des régions.

Pour chaque région, il faut ajouter les informations calculées (moyenne et nombre de prélèvements). **left_join** fusionnent les informations calculées avec la couche géospatiale des régions.

Centrer la carte sur la France

setView(lng = 2.2137, lat = 46.603354, zoom = 5) : Centre la carte sur la France (longitude 2.2137, latitude 46.603354). zoom = 5 pour faire afficher toutes les régions