Capsule 7 - Les tableaux (*dataframe*)

Bonjour à tous, je m'appelle Guillaume Dubé et je suis étudiant à la maîtrise en épidémiologie à l'École de santé publique de l'Université de Montréal. Bienvenue dans cette septième capsule qui abordera le concept des tableaux (*dataframe*).

Section théorique

Qu'est-ce qu'un tableau ? Selon Wikipédia, un tableau (base de données ou *dataframe*) permet de stocker et de retrouver des données structurées, semi-structurées ou des données brutes, souvent en rapport avec un thème ou une activité. La différence entre une base de données et une matrice est que la base de données peut contenir plusieurs types de vecteurs. On se rappelle qu'une matrice pouvait seulement accueillir des vecteurs de même type (se référer à la capsule 6). Contrairement aux matrices, les bases de données n'ont pas de type spécifique.

En résumé, dans le langage de programmation R, un tableau de données à deux dimensions se nomme base de données (*dataframe*). Cette base de données n'est pas soumise à la condition de contenir des vecteurs de même type. Nous allons nous familiariser à ce concept dans la section pratique.

Section pratique

Tout d'abord, nous allons vouloir créer une base de données à l'aide de la fonction data.frame. La fonction suivante nous permet de créer une nouvelle base de données qui comporte quatre variables.

```
data <- data.frame(
  patient_id = 1:5,
  patient_age = c(82,33,19,51,47),
  patient_sex = c("Male", "Male", "Female", "Female", "Female"),
  patient_disease = c(131,90,5,5,25)
  )
```

Une fois cette dernière créer, nous pouvons l'afficher avec diverses fonctions, soit View(data), print(data) ou head(data).

Il est maintenant de savoir qu'il existe une multitude de bases de données préenregistrée dans R. Pour les obtenir, il suffit d'utiliser la fonction data. Par exemple : data("mtcars"), data("iris") ,data("USArrests").

En ce qui a trait à l'ouverture d'une base de données sur notre ordinateur, il existe plusieurs méthodes. Premièrement, il existe la méthode « point and click » du menu déroulant. Pour ce faire, il suffit d'aller dans *File* → *Import Dataset* → *From text (base)* et de sélectionner notre base de données. Par la suite, un nouveau menu s'affichera et nous allons pouvoir prévisualiser cette dernière. Quelques modifications sont alors possibles. Une deuxième méthode nécessite l'adresse de la base de données. Par exemple, un fichier .csv peut s'ouvrir avec la fonction

```
read.csv(read.csv("C:/Users/Guill/Desktop/Université/Projet capsules R médecine/apacheApsVar.csv")
```

Pour lire des fichiers Excel, nous pouvons utiliser la librairie readxlsx qui contient la fonction read.xlsx(). La documentation en ligne sur l'ouverture de base de données est très grande, je vous conseille fortement d'y jeter un coup d'œil au besoin.

Il est maintenant possible de sélectionner des variables (colonnes) ou des rangées (patients), par exemple :

```
apacheApsVar$intubated
```

ou

```
apacheApsVar[145,]
```

Nous pouvons aussi émettre une ou plusieurs conditions :

```
apacheApsVar[apacheApsVar$wbc > 10,]
apacheApsVar[apacheApsVar$wbc > 10 & apacheApsVar$motor == 6,]
```

Il est maintenant possible de créer un sous-ensemble de la base de données avec la fonction subset

```
subset1 <- subset(apacheApsVar, wbc > 10 | motor == 6)
```

Nous allons donc pouvoir manipuler les variables telles que nous l'avons vu dans les deux capsules précédentes. Par exemple, nous allons pouvoir remplacer toutes les valeurs -1 de la base de données par une valeur manquante (NA)

```
subset1[subset1 == -1] <- NA
```

Finalement, pour enregistrer notre nouvelle base de données, nous pouvons utiliser la fonction save.

```
save(subset1, file = "subset1.Rdata")
```

Pour connaître l'emplacement de cette nouvelle base de données, nous devons utiliser la fonction getwd().

Nous pouvons changer le working directory (wd) avec la fonction setwd ou avec le menu déroulant, Session → Set Working Directory → Choose Directory.

Finalement, il existe plusieurs packages qui nous permettent de sauvegarder les bases de données en d'autres types de fichiers. Par exemple, pour les fichiers Excel, nous pouvons utiliser la fonction write_xlsv du package writexl.

```
library("writexl")
write_xlsx(subset1,"C:/Users/Guill/Desktop/Université/Projet capsules R médecine/subset2.xlsx")
```

Voilà ce qui complète la septième capsule.