Capsule10. Combiner des tableaux

Simon LaRue

12/11/2022

Introduction

En début de projet, il arrive souvent que les données nécessaires se retrouvent éparpillées dans plusieurs jeux de données. Pour réaliser nos analyses statistiques, il faut regrouper ces différents tableaux dans une seule base de données. Dans cette capsule, nous présentons la base du regroupement de tableaux. Nous parlerons d'addition de tables; suivra une petite introduction au concept de clé primaire ainsi que la combinaison de tables avec des jointures. Des commandes de la librairie `dplyr` et quelques commandes de base de `R` seront également présentées.

Base de données pour les exemples

On commence par importer les deux jeu de données `apachePatientResult.csv` et `apacheApsVar.csv`. Dans le cardre de cette capsule nous nous limiterons à des parties des jeu de données pour mieux illustré les concepts de combinaison de table.

```r
library(dplyr)

# Importation des données
apache_resultat <- read.csv("apachePatientResult.csv")
apache_variables <- read.csv("apacheApsVar.csv")

# Préparation des données
apa_res <- apache_resultat %>%
  select(apachepatientresultsid,
         patientunitstayid,
         apachescore) %>%
  rename(resultat_ID = apachepatientresultsid,
         patient_ID = patientunitstayid,
         score = apachescore)

apa_var <- apache_variables %>%
  select(patientunitstayid,
         heartrate,
         temperature) %>%
  rename(patient_ID = patientunitstayid)

# Création de tables pour les exemples

# 4 résultats
table_1 <- apa_res %>% slice(2 * (1:4))

# 4 autres résultats
table_2 <- apa_res %>% slice(100 + 2 * (1:4))

# Mesures de caractéristiques des patients de la tables 1
table_4 <- apa_var %>%
  filter(patient_ID %in% table_1$patient_ID)

# Mesures de caractéristique de 8 patients
table_5 <- apa_var %>% slice(1:8)
```

Addition de tableaux

Commençons par les deux combinaisons élémentaires de vecteurs et de tableaux qui sont la base de l'addition de tableaux : l'addition des lignes et l'addition des colonnes. Ces deux opérations se veulent simples et de base permettant de manipuler nos jeux de données. Avec un peu d'expérience, elles deviennent un outil efficace du traitement des données.

bind_row, rbind

Dans cette addition de table, nous souhaitons combiner les lignes de deux tableaux possédant les mêmes colonnes. Cette opération consiste tout simplement à superposer une table par-dessus une autre. Par exemple, nous voudrions combiner plusieurs registres de patients dans un seul tableau. En additionnant les tables selon leurs lignes, nous ajoutons les patients de notre seconde table à la suite des patients de la première. Pour y parvenir, nous pouvons utiliser la commande `bind_rows()` de la librairie `dplyr`. La commande `rbind()` de base est aussi une solution. Il est important que le nombre de colonnes soit le même dans les deux tableaux et que les noms des colonnes correspondent, sinon une erreur peut survenir. Il faut donc être prudent et s'assurer du bon format des tableaux avant de les combiner.

Exemple

Dans cette exemple, nous allons combiner les 4 individus de la `table_1` et les 4 individus de la `table_2` dans une nouvelle table que nous nommerons `table_3`.

```r
# 4 résultats
table_1
```

```
##   resultat_ID patient_ID score
## 1       31918     141765    47
## 2       21397     143870    60
## 3         182     144815    25
## 4      106475     145427    37
```

```r
# 4 autres résultats
table_2
```

```
##   resultat_ID patient_ID score
## 1       76550     181715    45
## 2      143760     181906    53
## 3      124529     184757    50
## 4       28864     185387    17
```

```r
# bind_rows() de la librairie dplyr
table_3 <- bind_rows(table_1, table_2)
table_3
```

```
##   resultat_ID patient_ID score
## 1       31918     141765    47
## 2       21397     143870    60
## 3         182     144815    25
## 4      106475     145427    37
## 5       76550     181715    45
## 6      143760     181906    53
## 7      124529     184757    50
## 8       28864     185387    17
```

```r
# nous pouvons aussi utiliser la commande rbind() de base comme suit
# rbind(table_1, table_2)
```

bind_cols, cbind

Une autre addition de table, où, cette fois-ci, nous souhaitons combiner les colonnes de deux tableaux possédant les mêmes lignes. Cette opération consiste à juxtaposer une table à côté d'une autre. Par exemple, nous voudrions ajouter à un registre de patients une nouvelle colonne d'informations pour ces mêmes patients provenant d'un autre jeu de données. Pour y parvenir, nous pouvons utiliser la commande `bind_cols()` de la librairie `dplyr`. La commande `cbind()` de base est aussi une solution qui fonctionne de la même manière. Peu importe la commande employée, nous retournerons une erreur si le nombre de lignes n'est pas le même dans les deux tableaux. Il est aussi à noter que les colonnes possédant des noms identiques seront toutes deux conservées.

Exemple

Dans cette exemple, nous allons combiner les 4 individus de la `table_1` avec leurs caractérisqtiques présente dans la `table_4`.

```r
# 4 résultats
table_1
```

```
##   resultat_ID patient_ID score
## 1       31918     141765    47
## 2       21397     143870    60
## 3         182     144815    25
## 4      106475     145427    37
```

```r
# Mesure des caratéristique des patients correspondant
table_4
```

```
##   patient_ID heartrate temperature
## 1     141765        88        36.2
## 2     143870        40        36.4
## 3     144815       131        36.7
## 4     145427        49        36.2
```

```r
# bind_cols de la librairie dplyr
bind_cols(table_1, table_4) # Noms identiques des colonnes patient_ID modifiés
```

```
## New names:
## * patient_ID -> patient_ID...2
## * patient_ID -> patient_ID...4
```

```
##   resultat_ID patient_ID...2 score patient_ID...4 heartrate temperature
## 1       31918         141765    47         141765        88        36.2
## 2       21397         143870    60         143870        40        36.4
## 3         182         144815    25         144815       131        36.7
## 4      106475         145427    37         145427        49        36.2
```

```r
# Supression d'une colonne patient_ID avant la fusion
bind_cols(table_1, table_4 %>% select(-patient_ID))
```

```
##   resultat_ID patient_ID score heartrate temperature
## 1       31918     141765    47        88        36.2
## 2       21397     143870    60        40        36.4
## 3         182     144815    25       131        36.7
## 4      106475     145427    37        49        36.2
```

```r
# on peut aussi utiliser la commande cbind de base comme suit
# cbind(table_1, table_4)
```

Clés primaires

Le concept de clé primaire consiste en une variable, ou une combinaison de variables, qui permet d'identifier de manière unique une seule ligne d'un jeu de données. Nous la retrouvons dans plusieurs de nos tableaux, de sorte que s'établisse le lien entre les différentes tables. Par exemple, connaissant l'identifiant d'un individu présent dans un registre de patient, il serait possible de retrouver cet individu dans la base de données. Dans ce cas, l'identifiant serait un numéro unique pour le patient, donc personne n'aurait le même. Nous pourrions alors utiliser le numéro d'identification dans un autre tableau pour associer de l'information à notre patient. Pour continuer dans notre exemple, si nous disposions d'une base de données de résultats de prise de sang, l'identifiant permettrait d'associer l'échantillon correspondant au patient. En somme, les clés primaires nous permettraient d'élaborer sur les liens entre les tableaux.

Jointure

Une jointure est une opération d'association de tables utilisant le concept de clés primaires. Contrairement aux opérations élémentaires d'addition de tableau, l'association des tables avec une jointure se fait par un lien logique. Par le fait même, nous nous assurons que l'information des tables soit jointe aux bons individus et non seulement juxtaposée à côté ou dupliquée en dessous de table. La combinaison par le bien d'une clé primaire permet une plus grande flexibilité. Différents types de jointures permettront de manipuler plus efficacement l'information à conserver dans nos jeux de données. Les jointures couramment utilisées en `R` sont la jointure complète, la jointure interne, la jointure externe gauche et la jointure externe droite.

Jointure complète

La jointure complète produit une table qui conserve l'intégralité des lignes des jeux de données. Les lignes présentes dans toutes les tables sont combinées à l'aide de la clé primaire. Les lignes présentent dans seulement l'un ou l'autre des tableaux sont aussi ajoutées au tableau final. Seules les colonnes, pour lesquelles il manque de l'information, seront complétées avec des valeurs manquantes. Par exemple, nous avons deux registres de patients dont les données ont été prises à des dates différentes. Certains patients sont présents dans les deux registres alors que d'autres se retrouvent seulement dans un. Nous voulons combiner les deux registres en un seul afin de faire notre base de données. La jointure complète nous permet de faire un tableau contenant tous les patients enregistrés, qu'ils se soient présentés une seule fois ou deux. Dans `R`, la commande `full_joint()` de la librairie `dplyr` permet de combiner des jeux de données par une jointure complète.

Exemple

```r
# Tables des résultats de 8 patients
table_3
```

```
##   resultat_ID patient_ID score
## 1       31918     141765    47
## 2       21397     143870    60
## 3         182     144815    25
## 4      106475     145427    37
## 5       76550     181715    45
## 6      143760     181906    53
## 7      124529     184757    50
## 8       28864     185387    17
```

```r
# Tables des caractéristiques de 8 patients
table_5
```

```
##   patient_ID heartrate temperature
## 1     141765        88        36.2
## 2     143870        40        36.4
## 3     144815       131        36.7
## 4     145427        49        36.2
## 5     147307       115        36.8
## 6     147784       109        36.0
## 7     148611       105        36.2
## 8     149433        98        -1.0
```

```r
# jointure complète de la librairie dplyr
full_join(table_3, table_5, by = c("patient_ID"))
```

```
##    resultat_ID patient_ID score heartrate temperature
## 1        31918     141765    47        88        36.2
## 2        21397     143870    60        40        36.4
## 3          182     144815    25       131        36.7
## 4       106475     145427    37        49        36.2
## 5        76550     181715    45        NA          NA
## 6       143760     181906    53        NA          NA
## 7       124529     184757    50        NA          NA
## 8        28864     185387    17        NA          NA
## 9           NA     147307    NA       115        36.8
## 10          NA     147784    NA       109        36.0
## 11          NA     148611    NA       105        36.2
## 12          NA     149433    NA        98        -1.0
```

Jointure interne

La jointure interne permet de regrouper dans une table l'information des lignes communes aux jeux de données. L'opération conserve uniquement les lignes du premier tableau identifiées dans le deuxième tableau par une clé primaire. Pour ce qui est des lignes identifiées seulement dans une table, peu importe laquelle, elles sont exclues du jeu de données final. Par exemple, reprenons nos deux registres de patients complétés à des dates différentes, dont nous voulons étudier seulement les patients enregistrés dans les deux registres. La jointure interne va alors nous faire un tableau contenant ces individus, tout en excluant ceux qui se sont présentés à un seul moment. Pour réaliser une jointure interne dans `R`, il suffit d'utiliser la commande `inner_joint()` de la librairie `dplyr`.

Exemple

```r
# Tables des résultats de 8 patients
table_3
```

```
##   resultat_ID patient_ID score
## 1       31918     141765    47
## 2       21397     143870    60
## 3         182     144815    25
## 4      106475     145427    37
## 5       76550     181715    45
## 6      143760     181906    53
## 7      124529     184757    50
## 8       28864     185387    17
```

```r
# Tables des caractéristiques de 8 patients
table_5
```

```
##   patient_ID heartrate temperature
## 1     141765        88        36.2
## 2     143870        40        36.4
## 3     144815       131        36.7
## 4     145427        49        36.2
## 5     147307       115        36.8
## 6     147784       109        36.0
## 7     148611       105        36.2
## 8     149433        98        -1.0
```

```r
# jointure interne de la librairie dplyr
inner_join(table_3, table_5, by = c("patient_ID"))
```

```
##   resultat_ID patient_ID score heartrate temperature
## 1       31918     141765    47        88        36.2
## 2       21397     143870    60        40        36.4
## 3         182     144815    25       131        36.7
## 4      106475     145427    37        49        36.2
```

Jointure externe (gauche/droite)

La jointure externe gauche ou jointure externe droite permet de conserver l'entièreté des lignes d'une seule des deux tables. Autrement dit, le tableau final contient toutes les lignes d'une première table auxquelles est jointe l'information de la deuxième table avec la clé primaire. Les lignes de la deuxième table sont exclues dans la table finale. Une jointure gauche est équivalente à une jointure droite qu'on aurait inversé les tables. Reprenons l'exemple des deux registres pris à des dates différentes. Supposons que nous nous intéressons uniquement aux patients du premier registre, en sachant tout de même lesquels se sont enregistrés au deuxième. La jointure externe gauche nous permet de compléter notre tableau en allant chercher l'information de la deuxième table sans en récupérer les lignes. Le principe de la jointure externe droite, dans ce cas, serait lorsqu'on s'intéresse plutôt à tous les patients du deuxième registre et qu'on souhaite regrouper l'information du premier sans en copier les lignes. Les jointures externes gauche et droite sont implémentées en R dans la librairie `dplyr` avec les commandes `left_joint()` et `right_joint()` respectivement.

Exemple

```r
# Tables des résultats de 8 patients
table_3
```

```
##   resultat_ID patient_ID score
## 1       31918     141765    47
## 2       21397     143870    60
## 3         182     144815    25
## 4      106475     145427    37
## 5       76550     181715    45
## 6      143760     181906    53
## 7      124529     184757    50
## 8       28864     185387    17
```

```r
# Tables des caractéristiques de 8 patients
table_5
```

```
##   patient_ID heartrate temperature
## 1     141765        88        36.2
## 2     143870        40        36.4
## 3     144815       131        36.7
## 4     145427        49        36.2
## 5     147307       115        36.8
## 6     147784       109        36.0
## 7     148611       105        36.2
## 8     149433        98        -1.0
```

```r
# jointure externe gauche de la librairie dplyr
left_join(table_3, table_5, by = c("patient_ID"))
```

```
##   resultat_ID patient_ID score heartrate temperature
## 1       31918     141765    47        88        36.2
## 2       21397     143870    60        40        36.4
## 3         182     144815    25       131        36.7
## 4      106475     145427    37        49        36.2
## 5       76550     181715    45        NA          NA
## 6      143760     181906    53        NA          NA
## 7      124529     184757    50        NA          NA
## 8       28864     185387    17        NA          NA
```

```r
# jointure externe droite de la librairie dplyr
right_join(table_3, table_5, by = c("patient_ID"))
```

```
##   resultat_ID patient_ID score heartrate temperature
## 1       31918     141765    47        88        36.2
## 2       21397     143870    60        40        36.4
## 3         182     144815    25       131        36.7
## 4      106475     145427    37        49        36.2
## 5          NA     147307    NA       115        36.8
## 6          NA     147784    NA       109        36.0
## 7          NA     148611    NA       105        36.2
## 8          NA     149433    NA        98        -1.0
```

```r
# équivalence entre jointure externe droite et jointure externe gauche
left_join(table_5, table_3, by = c("patient_ID"))
```

```
##   patient_ID heartrate temperature resultat_ID score
## 1     141765        88        36.2       31918    47
## 2     143870        40        36.4       21397    60
## 3     144815       131        36.7         182    25
## 4     145427        49        36.2      106475    37
## 5     147307       115        36.8          NA    NA
## 6     147784       109        36.0          NA    NA
## 7     148611       105        36.2          NA    NA
## 8     149433        98        -1.0          NA    NA
```

Exercises

Utiliser les jeux de données présentés dans l'introduction pour répondre aux questions.

1. Identifié la clé primaire du jeu de données apache_resultat?
2. Combiner le jeu apache_resultat avec le jeu de données apache_variables et identifier combient de lignes sont présentes dans la table résultante. Réaliser la combinaison en utilisant:
    - une jointure complète.
    - une jointure interne.
    - une jointure externe droite avec apache_resultat comme première table.
    - une jointure externe gauche avec apache_resultat comme première table.
3. À la lueur des résultats obtenu en 2, combient de lignes du tableau apache_resultat nous ne retrouvons pas de correspondance dans le tableau apache_variables? qu'en est-il du nombre de lignes du tableau apache_variables nous ne retrouvons pas de correspondance dans le tableau apache_resultat?