DPLYR

Xueting YIN

2020/12/9

Introduction

Dplyr est une extension facilitant le traitement et la manipulation de données contenues dans une ou plusieurs tables. Elle propose une syntaxe claire et cohérente, sous formes de verbes, pour la plupart des opérations de ce type. Ses fonctions sont en général plus rapides que leur équivalent sous R de base, elles permettent donc de traiter efficacement des données de grande dimension.

Dans ce minituto, nous allons utiliser les données iris qui existent dans Rstudio. Les données iris est un data frame avec 150 observations (lignes) et 5 variables (colonnes) nommées Sepal.Length, Sepal.Width, Petal.Length, Petal.Width et Species.

```
# install.packages("dplyr")
library(dplyr) #chargement de package dplyr
```

Manipulation des données:

Select: sélectionner des colonnes d'un tableau de données

```
select(iris, 1:2) #selectionner la 1ère et la 2ème colonne
select(iris,c(1,2))
select(iris,c("Sepal.Length","Sepal.Width"))
#les résultats sont idem les codes ci-dessus

select(iris, -Species) # ne pas montrer la colonne Species
grep("Sepal", colnames(iris)) # vérifier si "Sepal" existe dans les colonnes de iris et ret
ourner à sa position
select(iris, grep("Sepal", colnames(iris))) # selectionner les colonnes qui contiennent le
mot "Sepal"
```

Les autres fonctions de select

```
select(iris, contains(".")) #selectionner les colonnes qui contiennent "."
select(iris, starts_with("Sepal")) #selectionner les colonnes qui commencent par "Sepal"
select(iris, ends_with("Length")) #selectionner les colonnes qui finissent par "Length"
select(iris, everything()) #selectionner toutes les colonnes
select(iris, one_of("Species", "blabla")) #selectionner un de deux colonnes, même si la col
onne blabla n'existe pas dans iris
select(iris, matches(".t.")) # "." est une chaine de caractère, qui veut dire __+t+__, sele
ctionner les colonne contient "t"
select(iris, num_range("x", 1:5)) # selectionner les colonnes nommées x1,x2...x5, ce code n
'approprie pas à iris
```

filter

Sélectionne des lignes d'une table selon une condition

```
filter(iris, Sepal.Length >5) #selectionner les lignes où Sepal.Length est supérieur à 5.
```

slice

Sélectionne des lignes du tableau selon leur position

```
slice(iris, 1:5) # selectionner de la ligne 1 à la ligne 5
slice(iris, 2) # selectionner la ligne 2
```

distinct

Filtre les lignes du tableau pour ne conserver que les lignes distinctes, en supprimant toutes les lignes en double

```
distinct(iris) # supprimer les lignes redondantes
```

sample

Tirage

```
sample_frac(iris, 0.1) # tirer 10% de lignes
sample_n(iris, 10) # tirer 10 lignes
```

arrange

Réordonne les lignes d'un tableau selon une ou plusieurs colonnes

```
arrange(iris, Sepal.Length) # selon sepal length, trier en ordre croissant
arrange(iris, desc(Sepal.Length)) #selon sepal length, trier en ordre décroissant
```

join

Fusionner par le même nom de colonne, donc data.frame doit être nommé

Nous allons créer un data frame:

```
a <- 1:10
b <- 20:7
df1 = cbind(a,a^2)
df2 = cbind(b,b^3)
```

Nous utilisons la fonction join

```
colnames(df1) = c("x1","x2") # renommer les colonnes de df1
df1 = tibble::as_tibble(df1)
colnames(df2) = c("x1","x3") # renommer les colonnes de df2
df2 = tibble::as_tibble(df2)

left_join(df1,df2) # Correspond à toutes les valeurs de df1, les valeurs qui n'existent pas
 dans df2 afficheront NA
#nous pouvons également utiliser by = pour correspondre quel ligne
right_join(df1,df2) # Correspond à toutes les valeurs de df2, les valeurs qui n'existent pa
```

```
s dans df1 afficheront NA
inner_join(df1,df2) # Correspond à toutes les valeurs communes de df1 et df2
full_join(df1,df2) # Correspond à toutes les valeurs de df1 et df2
```

mutate

Mutate permet de créer de nouvelles colonnes dans le tableau de données, en général à partir de variables existantes

```
mutate(iris, Petal.Size = Petal.Width + Petal.Length)
```

group by

Permet de définir des groupes de lignes à partir des valeurs d'une ou plusieurs colonnes

Utilisation avec summarise qui permet d'agréger les lignes du tableau en effectuant une opération "résumée" sur une ou plusieurs colonnes

```
summarise(group_by(iris, Species), mean(Sepal.Length)) # Regrouper par Species et trouver l
a moyenne
```

Nous pouvons résumer plusieurs colonnes et trouver la moyenne

```
summarise(group_by(iris, Species), Mean.Petal.Size = mean(Petal.Width + Petal.Length), Mean
.Sepal.Size = mean(Sepal.Width + Sepal.Length))
```

```
## # A tibble: 3 x 3
##   Species    Mean.Petal.Size Mean.Sepal.Size
##   <fct>                <dbl>           <dbl>
## 1 setosa                1.71            8.43
## 2 versicolor            5.59            8.71
## 3 virginica             7.58            9.56
```

Nous pouvons également écrire comme ceci

```
tibble::as_tibble(iris) %>% group_by(Species) %>% summarise(Mean.Petal.Size = mean(Petal.Wi
dth + Petal.Length), Mean.Sepal.Size = mean(Sepal.Width + Sepal.Length))
```

```
## # A tibble: 3 x 3
##   Species    Mean.Petal.Size Mean.Sepal.Size
##   <fct>                <dbl>           <dbl>
## 1 setosa                1.71            8.43
## 2 versicolor            5.59            8.71
## 3 virginica             7.58            9.56
```

rename

Permet de renommer des colonnes

```
rename(iris, S.W = Sepal.Width)
```