

Remaniement de données avec dplyr et tidyr

Aide-mémoire



Syntaxe - conventions utiles

dplyr::tbl_df(iris)

Convertit le jeu de données en classe *tbl*.

Les *tbl* sont plus faciles à explorer que les data frames :

R n'affiche que les données adaptées à la taille de l'écran

Source: local data frame [150 x 5]

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
..

Variables not shown: Petal.Width (dbl), Species (fctr)

dplyr::glimpse(iris)

Fournit un résumé des jeux de données de class *tbl*

utils::View(iris)

Affiche les données dans un tableur (attention au V majuscule)

dplyr::%>%

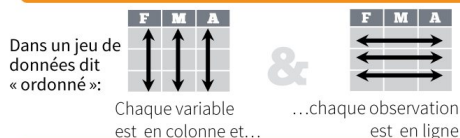
Passer l'objet se trouvant à gauche comme premier argument de la fonction se trouvant à droite.

$x \%>\% f(y)$ équivaut à $f(x, y)$
 $y \%>\% f(x, ., z)$ équivaut à $f(y, x, z)$

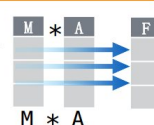
Utiliser l'opérateur `%>%` rend le code plus lisible :

```
iris %>%  
  group_by(Species) %>%  
  summarise(avg = mean(Sepal.Width)) %>%  
  arrange(avg)
```

Jeu de données ordonné - la base du remaniement de données



Les jeux de données ordonnés sont complémentaires de la vectorisation dans R. R préserve les observations quand les variables sont manipulées. Aucun autre format ne fonctionne aussi intuitivement que celui de R.



Réorganisation des données - changer la disposition des données



tidyr::gather(cases, "year", "n", 2:4)
Fusionne des colonnes en lignes.



tidyr::separate(storms, date, c("y", "m", "d"))
Divise une colonne en plusieurs.



tidyr::spread(pollution, size, amount)
Distribue les lignes dans des colonnes.



tidyr::unite(data, col, ..., sep)
Concatène plusieurs colonnes en une seule.

dplyr::data_frame(a = 1:3, b = 4:6)
Combine les vecteurs dans un *data frame* (de façon optimisée).

dplyr::arrange(mtcars, mpg)

Trie les observations par les valeurs d'une variable (ordre croissant).

dplyr::arrange(mtcars, desc(mpg))
Trie les observations par les valeurs d'une colonne (ordre décroissant).

dplyr::rename(tb, y = year)

Renomme les variables du jeu de données.

Extraction d'observations (lignes)



dplyr::filter(iris, Sepal.Length > 7)

Permet d'extraire des observations selon une condition logique

dplyr::distinct(iris)

Dédoublonne la base

dplyr::sample_frac(iris, 0.5, replace = TRUE)

Sélectionne aléatoirement une fraction d'observations

dplyr::sample_n(iris, 10, replace = TRUE)

Sélectionne aléatoirement n observations

dplyr::slice(iris, 10:15)

Sélectionne les lignes selon leur position

dplyr::top_n(storms, 2, date)

Sélectionne et ordonne les n premières observations (ou groupes si les données sont groupées)

Opérateurs logiques dans R - ?Comparison et ?base::Logic		
<	Inférieur strictement à	!=
>	Supérieur strictement à	%in%
==	Egal à	is.na
<=	Inférieur ou égal à	!is.na
>=	Supérieur ou égal à	&, , !, xor, any, all
		Opérateurs booléens

Extraction de variables (colonnes)



dplyr::select(iris, Sepal.Width, Petal.Length, Species)

Sélectionne des colonnes selon leur nom ou leur fonction assistantes

Fonctions assistantes à la sélection - ?select

select(iris, contains("."))

Sélectionne les variables contenant la chaîne de caractères "."

select(iris, ends_with("Length"))

Sélectionne les variables se terminant par la chaîne de caractères "Length"

select(iris, everything())

Sélectionne toutes les variables

select(iris, matches("."))

Sélectionne toutes les variables qui correspondent à l'expression régulière .t

select(iris, num_range("x", 1:5))

Sélectionne les variables nommées x1, x2, x3, x4, x5.

select(iris, one_of(c("Species", "Genus")))

Sélectionne les variables dans la liste de noms spécifiée

select(iris, starts_with("Sepal"))

Sélectionne les variables débutant par la chaîne de caractères "Sepal"

select(iris, Sepal.Length:Petal.Width)

Sélectionne toutes les variables de Sepal.Length à Petal.Width (incluses).

select(iris, - Species)

Sélectionne toutes les variables sauf Species.

Résumer des données



dplyr::summarise(iris, avg = mean(Sepal.Length))

Résume de l'information en une seule ligne

dplyr::summarise_each(iris, funs(mean))

Applique une fonction (de résumé) sur chaque variable

dplyr::count(iris, Species, wt = Sepal.Length)

Dénombre le nombre d'observations de chaque valeur d'une variable (avec ou sans poids)



Summarise utilise des fonctions de résumé qui prennent en entrée un vecteur de valeurs et retournent une seule valeur tel que:

dplyr::first	Première valeur d'un vecteur
dplyr::last	Dernière valeur d'un vecteur
dplyr::nth	N ^{ième} valeur d'un vecteur
dplyr::n	Nb de valeurs d'un vecteur
dplyr::n_distinct	Nb de valeurs distinctes d'un vecteur
IQR	IQR d'un vecteur
min	Valeur minimum d'un vecteur
max	Valeur maximum d'un vecteur
mean	Moyenne d'un vecteur
median	Médiane d'un vecteur
var	Variance d'un vecteur
sd	Ecart-type d'un vecteur

Grouper des données

dplyr::group_by(iris, Species)

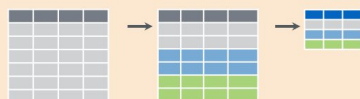
Regroupe les observations d'iris par la valeur de Species.

dplyr::ungroup(iris)

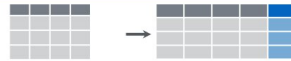
Dégroupes le jeu de données

iris %>% group_by(Species) %>% summarise(...)

Construit un *tbl* résumant chaque groupe



Construire de nouvelles variables



dplyr::mutate(iris, sepal = Sepal.Length + Sepal.Width)

Calcule et ajoute une ou plusieurs nouvelles variables

dplyr::mutate_each(iris, funs(min_rank))

Applique une fonction window à chaque variable

dplyr::transmute(iris, sepal = Sepal.Length + Sepal.Width)

Construit une ou plusieurs variables en supprimant les originales

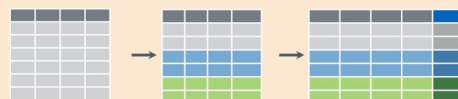


Mutate utilise des fonctions window qui prennent en entrée un vecteur et retournent un vecteur tel que:

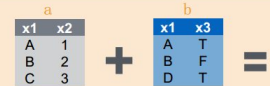
dplyr::lead	Copier avec des valeurs décalées à gauche
dplyr::lag	Copier avec des valeurs décalées à droite
dplyr::dense_rank	Ordonne sans sauts de rangs
dplyr::min_rank	Ordonne avec sauts de rangs
dplyr::percent_rank	Rangs de (min_rank) entre [0, 1].
dplyr::row_number	Ordonne en affectant aux liens la première position.
dplyr::ntile	Divise en n groupes.
dplyr::between	Les valeurs sont-elles entre a et b?
dplyr::cume_dist	Distribution cumulée
dplyr::cumall	Cumul tant que vrai
dplyr::cumany	Cumul dès que vrai
dplyr::cummean	Moyenne glissante
cumsum	Somme cumulée
cummax	Maximum cumulé
cummin	Minimum cumulé
cumprod	Produit cumulé
pmax	Maximum par élément
pmin	Minimum par élément

iris %>% group_by(Species) %>% mutate(...)

Construit de nouvelles variables, par groupe



Fusionner des jeux de données



Jointures transformantes

x1	x2	x3
A	1	T
B	2	F
C	3	NA

dplyr::left_join(a, b, by = "x1")
Joindre à a les variables de b selon x1

x1	x3	x2
A	T	1
B	F	2
D	NA	NA

dplyr::right_join(a, b, by = "x1")
Joindre à b les variables de a selon x1

x1	x2	x3
A	1	T
B	2	F

dplyr::inner_join(a, b, by = "x1")
Joindre a et b en ne gardant que les observations des deux tableaux

x1	x2	x3
A	1	T
B	2	F
D	NA	NA

dplyr::full_join(a, b, by = "x1")
Joindre a et b en gardant toutes les observations

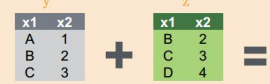
Jointures filtrantes

x1	x2
A	1
B	2

dplyr::semi_join(a, b, by = "x1")
Toutes les observations de a ayant des valeurs correspondantes dans b

x1	x2
A	1
C	3

dplyr::anti_join(a, b, by = "x1")
Toutes les observations de a n'ayant aucune correspondance dans b.



Opérations ensemblistes

x1	x2
B	2
C	3

dplyr::intersect(y, z)
Observations appartenant à y et z

x1	x2
A	1
B	2
C	3
D	4

dplyr::union(y, z)
Observations appartenant à y et z ou l'un des 2

x1	x2
A	1

dplyr::setdiff(y, z)
Observations appartenant à y et pas à z

Assemblages

x1	x2
A	1
B	2
C	3
D	4

dplyr::bind_rows(y, z)
Ajoute z à y comme nouvelles lignes.

x1	x2	x1	x2
A	1	B	2
B	2	C	3
C	3	D	4

dplyr::bind_cols(y, z)
Ajoute z à y comme nouvelles colonnes.
NB: matches rows by position.