Fonctions R

Dupliqués et niveaux

Commande	Utilité	Résultat	Exemple
`duplicated()`	identifie les dupliqués	retourne un vecteur booléen identifiant les dupliqués	`duplicated(c(1, 1, 2, 4, 6, 4))` `## [1] FALSE TRUE FALSE FALSE FALSE TRUE`
`unique()`	extrais les valeurs uniques	retourne un vecteur contenant les valeurs uniques du vecteur donné en argument	`unique(c(1, 1, 2, 4, 6, 4))` `## [1] 1 2 4 6`
`dplyr::distinct()`	exclure les lignes dupliquées d'une base de données	retourne la BD conservant la première occurrence de dupliqués	`data %>%` `  distinct()`

`levels()`	identifie les niveaux d'un facteur	retourne une list des niveaux du facteur	`fac <- factor(x = c("one", "two", "two"))`
			`levels(fac)`
			`## [1] "one" "two"`

`droplevels()`	identifie les niveaux d'un facteur non utilisés	retourne le facteur en enlevant les niveaux sans observations	`fac_filt <- fac[-1]`
			`droplevels(fac_filt)`
			`## [1] two two`
			`## Levels: two`

`which()`	identifie la **position** d'objets rencontrant la condition	retourne les indices des objets	`data_wt <- c(76, 87, NA, 47, 55, 42, 666, NA)`
			`which(is.na(data_wt))`
			`## [1] 3 8`

Agrégation

`table()` et `prop.table()` Retourne un tableau de fréquence et de proportion.
À 2 dimensions, il faut s'assurer d'avoir les mêmes dimensions pour les 2 vecteurs.
À 3 dimensions, la fonction retourne un tableau par niveau du troisième argument.
`prop.table()` s'enchaîne à une table et ne pas être utilisé directement.
L'argument `margin` spécifie la dimension sur laquelle sommer (1 pour les rangées, 2 pour les colonnes).

```
facto <- factor(x = c("deux", "two", "one", "one", "un", "deux"))
table(facto)

## facto
## deux  one  two   un
##    2    2    1    1

age <- c(18, 20, 19, 18, 20)
weight <- c(200, 150, 175, 190, 220)
table(age, weight)

##     weight
## age  150 175 190 200 220
##   18   0   0   1   1   0
##   19   0   1   0   0   0
##   20   1   0   0   0   1

table(age, weight) %>%
    prop.table(margin = 1)

##     weight
## age  150 175 190 200 220
##   18 0.0 0.0 0.5 0.5 0.0
##   19 0.0 1.0 0.0 0.0 0.0
##   20 0.5 0.0 0.0 0.0 0.5
```

`aggregate()` Calcule une fonction par groupe pour une base de données.
Il faut faire attention à la fonction donnée en argument. Si l'on veut calculer une moyenne et la BD contient des facteurs, la fonction retourne `NA` pour cette colonne avec un message d'erreur.
Il faut donner l'argument à `by` en forme de liste même si nous n'avons qu'une seule variable par laquelle grouper.

```
BD <- data.frame(age = c(18, 20, 19, 18, 20),
                 weight = c(200, 150, 175, 190, 220))
sex <- factor(c("F", "F", "H", "F", "H"))
student <- as.factor(c(T, F, T, T, F))
aggregate(x = BD, by = list(sex), FUN = median)
```

```
##   Group.1  age weight
## 1       F 18.0  190.0
## 2       H 19.5  197.5

aggregate(x = BD, by = list(sex, student), FUN = median)

##   Group.1 Group.2 age weight
## 1       F   FALSE  20    150
## 2       H   FALSE  20    220
## 3       F    TRUE  18    195
## 4       H    TRUE  19    175
```

Modifications de variables

Commande	Utilité	Résultat	Exemple

`tolower()` et `toupper()`	convertis les chaînes de caractères en minuscules et majuscules	retourne le vecteur avec les chaînes de caractères modifiées	`tolower(c("ALec", "JAmES"))`
			`## [1] "alec" "james"`
			`toupper(c("ALec", "JAmES"))`
			`## [1] "ALEC" "JAMES"`

Imputation et prévision

`element_text()`

Paramètre	Description	Possiblités	Exemple
`face`	type de police	"plain", "italic", "bold"	`face = "plain"`
`colour`	couleur du texte		`colour = "blue"`
`size`	taille du texte	pts	`size = 8`
`angle`	angle du texte	de 0 à 360 degrés	`angle = 90`
`hjust` et `vjust`	justification du text selon l'aire du graphique (et non selon les axes)	entre 0 et 1	`hjust = 0.5`