Tidyverse

Marchand Jean-Louis

07/01/2021

Lien vers la vidéo

Pour celles et ceux qui voudraient voir ou revoir la séance du jeudi 7 janvier 2021, voici le lien vers l'enregistrement :

https://bbb-rc.agrocampus-ouest.fr/playback/presentation/2.0/playback.html?meetingId=bf789842969ba8bc2f1a95c8d2e81c408316390e-1610024143121

Description approximative

Le paquet **tidyverse** est un regroupement de plusieurs paquets (cf https://www.tidyverse.org/ pour des descriptions et surtout les **cheatsheets**). Grossièrement :

- **tibble** pour la structure des données (nouveau format différent d'un data.frame ou .table)
- **dplyr** pour la manipulation et l'analyse proprement dite
- **forcats** pour des fonctionnalités propres aux facteurs
- **tidyr** pour la préparation des données (pivots et autres)
- **lubridate** pour la gestion des dates
- **stringr** pour la gestion des chaînes de caractères
- *readr* pour l'importation (format tibble en sortie)
- *ggplot2* pour la visualisation
- *purrr* pour la programmation
- ...

Chargement des paquets

```
> library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------- tidyverse 1.3.
```

```
## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------- tidyverse_conflicts(
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
> library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
> library(kableExtra)
```



```
##
```

Début de discussion

- boîte à outils standard
- lisibilité du code (/base)
- prise en main plus naturelle ? (format par défaut dans RStudio)
- comme toujours : coût d'entrée
- plus orienté utilisateur que performance (/data.table)

Présentation basée sur une lecture du tutoriel

https://juba.github.io/tidyverse/

ainsi que

https://haozhu233.github.io/kableExtra/awesome_table_in_html.html

Les tibbles

```
> library(palmerpenguins)
> data(package = 'palmerpenguins')
> penguins
```

```
## # A tibble: 344 x 8
##    species island bill_length_mm bill_depth_mm flipper_length_~ body_mass_g
##    <fct>   <fct>           <dbl>         <dbl>             <int>       <int>
##  1 Adelie  Torge~           39.1          18.7               181        3750
##  2 Adelie  Torge~           39.5          17.4               186        3800
##  3 Adelie  Torge~           40.3          18                 195        3250
##  4 Adelie  Torge~           NA            NA                  NA          NA
##  5 Adelie  Torge~           36.7          19.3               193        3450
##  6 Adelie  Torge~           39.3          20.6               190        3650
##  7 Adelie  Torge~           38.9          17.8               181        3625
##  8 Adelie  Torge~           39.2          19.6               195        4675
##  9 Adelie  Torge~           34.1          18.1               193        3475
## 10 Adelie  Torge~           42            20.2               190        4250
## # ... with 334 more rows, and 2 more variables: sex <fct>, year <int>
```

Premières manipulations

```
> glimpse(penguins)
```

```
## Rows: 344
## Columns: 8
## $ species           <fct> Adelie, Adelie, Adelie, Adelie, Adelie, Adelie, A...
## $ island            <fct> Torgersen, Torgersen, Torgersen, Torgersen, Torge...
## $ bill_length_mm    <dbl> 39.1, 39.5, 40.3, NA, 36.7, 39.3, 38.9, 39.2, 34....
## $ bill_depth_mm     <dbl> 18.7, 17.4, 18.0, NA, 19.3, 20.6, 17.8, 19.6, 18....
## $ flipper_length_mm <int> 181, 186, 195, NA, 193, 190, 181, 195, 193, 190, ...
## $ body_mass_g       <int> 3750, 3800, 3250, NA, 3450, 3650, 3625, 4675, 347...
## $ sex               <fct> male, female, female, NA, female, male, female, m...
## $ year              <int> 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2...
```

Changement de structure

```
> penguins.df <- as.data.frame(penguins)
> str(penguins.df)
```

```
## 'data.frame':    344 obs. of  8 variables:
##  $ species          : Factor w/ 3 levels "Adelie","Chinstrap",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ island           : Factor w/ 3 levels "Biscoe","Dream",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ bill_length_mm   : num  39.1 39.5 40.3 NA 36.7 39.3 38.9 39.2 34.1 42 ...
##  $ bill_depth_mm    : num  18.7 17.4 18 NA 19.3 20.6 17.8 19.6 18.1 20.2 ...
##  $ flipper_length_mm: int  181 186 195 NA 193 190 181 195 193 190 ...
##  $ body_mass_g      : int  3750 3800 3250 NA 3450 3650 3625 4675 3475 4250 ...
##  $ sex              : Factor w/ 2 levels "female","male": 2 1 1 NA 1 2 1 2 NA NA ...
##  $ year             : int  2007 2007 2007 2007 2007 2007 2007 2007 2007 2007 ...
```

```
> penguins.df <- as_tibble(penguins.df)
> str(penguins.df)
```

```
## tibble [344 x 8] (S3: tbl_df/tbl/data.frame)
##  $ species          : Factor w/ 3 levels "Adelie","Chinstrap",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ island           : Factor w/ 3 levels "Biscoe","Dream",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ bill_length_mm   : num [1:344] 39.1 39.5 40.3 NA 36.7 39.3 38.9 39.2 34.1 42 ...
##  $ bill_depth_mm    : num [1:344] 18.7 17.4 18 NA 19.3 20.6 17.8 19.6 18.1 20.2 ...
##  $ flipper_length_mm: int [1:344] 181 186 195 NA 193 190 181 195 193 190 ...
##  $ body_mass_g      : int [1:344] 3750 3800 3250 NA 3450 3650 3625 4675 3475 4250 ...
##  $ sex              : Factor w/ 2 levels "female","male": 2 1 1 NA 1 2 1 2 NA NA ...
##  $ year             : int [1:344] 2007 2007 2007 2007 2007 2007 2007 2007 2007 2007 ...
```

Remarque sur les colonnes

```
> str(penguins$species)

##  Factor w/ 3 levels "Adelie","Chinstrap",..: 1 1 1 1 1 1 1 1 1 1 ...
> str(penguins[,1])

## tibble [344 x 1] (S3: tbl_df/tbl/data.frame)
##  $ species: Factor w/ 3 levels "Adelie","Chinstrap",..: 1 1 1 1 1 1 1 1 1 1 ...
> str(penguins[[1]])

##  Factor w/ 3 levels "Adelie","Chinstrap",..: 1 1 1 1 1 1 1 1 1 1 ...
> str(simplify(penguins[,1]))

##  Factor w/ 3 levels "Adelie","Chinstrap",..: 1 1 1 1 1 1 1 1 1 1 ...
##  - attr(*, "names")= chr [1:344] "species1" "species2" "species3" "species4" ...
```

Les fonctions dplyr : rename

Changement des noms de variable

```
> penguins <- rename(penguins, bill_l = bill_length_mm,
+                   bill_d = bill_depth_mm,
+                   flip_l = flipper_length_mm,
+                   mass = body_mass_g)
> glimpse(penguins)
```

```
## Rows: 344
## Columns: 8
## $ species <fct> Adelie, Adelie, Adelie, Adelie, Adelie, Adelie, Adelie, Ade.
## $ island  <fct> Torgersen, Torgersen, Torgersen, Torgersen, Torgersen, Torg.
## $ bill_l  <dbl> 39.1, 39.5, 40.3, NA, 36.7, 39.3, 38.9, 39.2, 34.1, 42.0, 3.
## $ bill_d  <dbl> 18.7, 17.4, 18.0, NA, 19.3, 20.6, 17.8, 19.6, 18.1, 20.2, 1.
## $ flip_l  <int> 181, 186, 195, NA, 193, 190, 181, 195, 193, 190, 186, 180, .
## $ mass    <int> 3750, 3800, 3250, NA, 3450, 3650, 3625, 4675, 3475, 4250, 3.
## $ sex     <fct> male, female, female, NA, female, male, female, male, NA, N.
## $ year    <int> 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007,.
```

Les fonctions dplyr : slice

Sélection de lignes/individus par leur position

```
> slice(penguins, 123)
```

```
## # A tibble: 1 x 8
##   species island    bill_l bill_d flip_l  mass sex     year
##   <fct>   <fct>      <dbl>  <dbl>  <int> <int> <fct>  <int>
## 1 Adelie  Torgersen   40.2     17    176  3450 female  2009
```

```
> slice(penguins, 3:6)
```

```
## # A tibble: 4 x 8
##   species island    bill_l bill_d flip_l  mass sex     year
##   <fct>   <fct>      <dbl>  <dbl>  <int> <int> <fct>  <int>
## 1 Adelie  Torgersen   40.3   18      195  3250 female  2007
## 2 Adelie  Torgersen   NA     NA       NA    NA <NA>    2007
## 3 Adelie  Torgersen   36.7   19.3    193  3450 female  2007
## 4 Adelie  Torgersen   39.3   20.6    190  3650 male    2007
```

Les fonctions dplyr : filter

Sélection de lignes/individus par condition(s)

```
> filter(penguins, (species != "Adelie")&(bill_l >= 39))
```

```
## # A tibble: 191 x 8
##    species island bill_l bill_d flip_l  mass sex     year
##    <fct>   <fct>   <dbl>  <dbl>  <int> <int> <fct>  <int>
##  1 Gentoo  Biscoe   46.1   13.2    211  4500 female  2007
##  2 Gentoo  Biscoe   50     16.3    230  5700 male    2007
##  3 Gentoo  Biscoe   48.7   14.1    210  4450 female  2007
##  4 Gentoo  Biscoe   50     15.2    218  5700 male    2007
##  5 Gentoo  Biscoe   47.6   14.5    215  5400 male    2007
##  6 Gentoo  Biscoe   46.5   13.5    210  4550 female  2007
##  7 Gentoo  Biscoe   45.4   14.6    211  4800 female  2007
##  8 Gentoo  Biscoe   46.7   15.3    219  5200 male    2007
##  9 Gentoo  Biscoe   43.3   13.4    209  4400 female  2007
## 10 Gentoo  Biscoe   46.8   15.4    215  5150 male    2007
## # ... with 181 more rows
```

Les fonctions dplyr : select

Sélection de colonnes par nom ou condition

```
> select(penguins, species, bill_l)
```

```
## # A tibble: 344 x 2
##    species bill_l
##    <fct>    <dbl>
##  1 Adelie    39.1
##  2 Adelie    39.5
##  3 Adelie    40.3
##  4 Adelie    NA
##  5 Adelie    36.7
##  6 Adelie    39.3
##  7 Adelie    38.9
##  8 Adelie    39.2
##  9 Adelie    34.1
## 10 Adelie    42
## # ... with 334 more rows
```

```
> select(penguins, species:bill_l)
> select(penguins, starts_with("bill"))
> select(penguins, ends_with("_l")) ## aussi contains() et matches()
> select_if(penguins, is.numeric) ## par booléen
```

Les fonctions dplyr : arrange

Tri des observations suivant des valeurs des variables

```
> arrange(penguins,bill_d, desc(bill_l))
```

```
## # A tibble: 344 x 8
##    species island bill_l bill_d flip_l  mass sex     year
##    <fct>   <fct>   <dbl>  <dbl>  <int> <int> <fct>  <int>
##  1 Gentoo  Biscoe   42.9   13.1    215  5000 female  2007
##  2 Gentoo  Biscoe   46.1   13.2    211  4500 female  2007
##  3 Gentoo  Biscoe   44.9   13.3    213  5100 female  2008
##  4 Gentoo  Biscoe   43.3   13.4    209  4400 female  2007
##  5 Gentoo  Biscoe   46.5   13.5    210  4550 female  2007
##  6 Gentoo  Biscoe   42     13.5    210  4150 female  2007
##  7 Gentoo  Biscoe   44     13.6    208  4350 female  2008
##  8 Gentoo  Biscoe   47.2   13.7    214  4925 female  2009
##  9 Gentoo  Biscoe   45.5   13.7    214  4650 female  2007
## 10 Gentoo  Biscoe   45.3   13.7    210  4300 female  2008
## # ... with 334 more rows
```

Les fonctions dplyr : mutate

Création d'une nouvelle variable

```
> mutate(penguins, bill_length_cm = bill_l/10)
```

```
## # A tibble: 344 x 9
##    species island    bill_l bill_d flip_l  mass sex     year bill_length_cm
##    <fct>   <fct>      <dbl>  <dbl>  <int> <int> <fct>  <int>          <dbl>
##  1 Adelie  Torgersen   39.1   18.7    181  3750 male    2007           3.91
##  2 Adelie  Torgersen   39.5   17.4    186  3800 female  2007           3.95
##  3 Adelie  Torgersen   40.3   18      195  3250 female  2007           4.03
##  4 Adelie  Torgersen   NA     NA       NA    NA <NA>    2007          NA
##  5 Adelie  Torgersen   36.7   19.3    193  3450 female  2007           3.67
##  6 Adelie  Torgersen   39.3   20.6    190  3650 male    2007           3.93
##  7 Adelie  Torgersen   38.9   17.8    181  3625 female  2007           3.89
##  8 Adelie  Torgersen   39.2   19.6    195  4675 male    2007           3.92
##  9 Adelie  Torgersen   34.1   18.1    193  3475 <NA>    2007           3.41
## 10 Adelie  Torgersen   42     20.2    190  4250 <NA>    2007           4.2
## # ... with 334 more rows
```

Composition de fonctions : pipeline

Composition de gauche à droite pour plus de lisibilité !

```
> slice( arrange( select_if ( filter (penguins,  island == "Biscoe" ),
+                             is.numeric ), mass) , 1:3)
```

```
## # A tibble: 3 x 5
##   bill_l bill_d flip_l  mass  year
##    <dbl>  <dbl>  <int> <int> <int>
## 1   36.5   16.6    181  2850  2008
## 2   36.4   17.1    184  2850  2008
## 3   34.5   18.1    187  2900  2008
```

```
> penguins %>% filter( island == "Biscoe" ) %>%
+              select_if ( is.numeric ) %>%
+              arrange( mass ) %>%
+              slice(1:3)
```

```
## # A tibble: 3 x 5
##   bill_l bill_d flip_l  mass  year
##    <dbl>  <dbl>  <int> <int> <int>
## 1   36.5   16.6    181  2850  2008
## 2   36.4   17.1    184  2850  2008
## 3   34.5   18.1    187  2900  2008
```

Les fonctions dplyr : group_by

Regroupement des indvidus

```
> penguins %>% group_by(island) %>%
+              slice(1:2)
```

```
## # A tibble: 6 x 8
## # Groups:   island [3]
##   species island    bill_l bill_d flip_l  mass sex     year
##   <fct>   <fct>      <dbl>  <dbl>  <int> <int> <fct>  <int>
## 1 Adelie  Biscoe      37.8   18.3    174  3400 female  2007
## 2 Adelie  Biscoe      37.7   18.7    180  3600 male    2007
## 3 Adelie  Dream       39.5   16.7    178  3250 female  2007
## 4 Adelie  Dream       37.2   18.1    178  3900 male    2007
## 5 Adelie  Torgersen   39.1   18.7    181  3750 male    2007
## 6 Adelie  Torgersen   39.5   17.4    186  3800 female  2007
```

Les fonctions dplyr : summarise

Descriptions des données par sous-groupes

```
> penguins %>% group_by(island) %>%
+              summarise(long_moy = mean(bill_l, na.rm=T),
+                        long_max = max(bill_l, na.rm = T),
+                        nb = n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 3 x 4
##   island    long_moy long_max    nb
##   <fct>        <dbl>    <dbl> <int>
## 1 Biscoe        45.3     59.6   168
## 2 Dream         44.2     58     124
## 3 Torgersen     39.0     46      52
```

Les fonctions dplyr : count

Autre façon de compter

```
> penguins %>% count(island, species) %>%
+              arrange(desc(n))
```

```
## # A tibble: 5 x 3
##   island    species       n
##   <fct>     <fct>     <int>
## 1 Biscoe    Gentoo      124
## 2 Dream     Chinstrap    68
## 3 Dream     Adelie       56
## 4 Torgersen Adelie       52
## 5 Biscoe    Adelie       44
```

Cela revient quasiment au même que

```
> penguins %>% group_by(island, species) %>%
+              summarise(n=n()) %>%
+              arrange(desc(n))
```

Descriptions

Grouper/Dégrouper

```
> penguins %>% group_by(island, species) %>%
+              summarise(nb = n()) %>%
+              ungroup() %>%
+              mutate(pourcentage = nb / sum(nb) * 100)
```

```
## `summarise()` regrouping output by 'island' (override with `.groups` argumen
```

```
## # A tibble: 5 x 4
##   island    species      nb pourcentage
##   <fct>     <fct>     <int>       <dbl>
## 1 Biscoe    Adelie       44        12.8
## 2 Biscoe    Gentoo      124        36.0
## 3 Dream     Adelie       56        16.3
## 4 Dream     Chinstrap    68        19.8
## 5 Torgersen Adelie       52        15.1
```

Descriptions

Descriptions conditionnelles

```
> penguins %>% group_by(island, species) %>%
+              summarise_if(is.numeric,mean, na.rm = T)
```

```
## # A tibble: 5 x 7
## # Groups:   island [3]
##   island    species   bill_l bill_d flip_l  mass  year
##   <fct>     <fct>      <dbl>  <dbl>  <dbl> <dbl> <dbl>
## 1 Biscoe    Adelie      39.0   18.4   189. 3710. 2008.
## 2 Biscoe    Gentoo      47.5   15.0   217. 5076. 2008.
## 3 Dream     Adelie      38.5   18.3   190. 3688. 2008
## 4 Dream     Chinstrap   48.8   18.4   196. 3733. 2008.
## 5 Torgersen Adelie      39.0   18.4   191. 3706. 2008.
```

Descriptions

Descriptions de plusieurs variables

```
> penguins %>% summarise_at(vars(bill_l:flip_l),
+                           list(~ mean(., na.rm = TRUE),
+                                ~ median(., na.rm = TRUE)))
```

```
## # A tibble: 1 x 6
##   bill_l_mean bill_d_mean flip_l_mean bill_l_median bill_d_median flip_l_med
##         <dbl>       <dbl>       <dbl>          <dbl>          <dbl>          <d
## 1        43.9        17.2        201.           44.4           17.3
```

Autres façons

```
> penguins %>% summarise_at(vars(3:5),
+                           list(~ mean(., na.rm = TRUE),
+                                ~ median(., na.rm = TRUE)))
> penguins %>% summarise_at(vars(bill_l,bill_d,flip_l),
+                           list(~ mean(., na.rm = TRUE),
+                                ~ median(., na.rm = TRUE)))
```

Manipulations des facteurs : deux exemples

Changement des niveaux d'un facteur donné

```
> penguins %>%
+   mutate( species = fct_recode( species, "Delo" = "Adelie" )) %>%
+   select(1:3)
```

```
## # A tibble: 344 x 3
##    species island    bill_l
##    <fct>   <fct>      <dbl>
##  1 Delo    Torgersen   39.1
##  2 Delo    Torgersen   39.5
##  3 Delo    Torgersen   40.3
##  4 Delo    Torgersen   NA
##  5 Delo    Torgersen   36.7
##  6 Delo    Torgersen   39.3
##  7 Delo    Torgersen   38.9
##  8 Delo    Torgersen   39.2
##  9 Delo    Torgersen   34.1
## 10 Delo    Torgersen   42
## # ... with 334 more rows
```

Manipulations des facteurs : deux exemples

Changement de niveaux sur tous les facteurs

```
> pingoo <- penguins %>% mutate_if( is.factor, fct_recode,
+                                   "Delo" = "Adelie",
+                                   "Delo" = "Dream")
```

```
## Warning: Problem with `mutate()` input `species`.
## i Unknown levels in `f`: Dream
## i Input `species` is `(function (.f, ...) ...`.

## Warning: Unknown levels in `f`: Dream

## Warning: Problem with `mutate()` input `island`.
## i Unknown levels in `f`: Adelie
## i Input `island` is `(function (.f, ...) ...`.

## Warning: Unknown levels in `f`: Adelie

## Warning: Problem with `mutate()` input `sex`.
## i Unknown levels in `f`: Adelie, Dream
## i Input `sex` is `(function (.f, ...) ...`.

## Warning: Unknown levels in `f`: Adelie, Dream
```

Manipulations des facteurs : deux exemples

Changement des niveaux d'un facteur

```
> pingoo %>% select_if(is.factor) %>%
+             str()
```

```
## tibble [344 x 3] (S3: tbl_df/tbl/data.frame)
##  $ species: Factor w/ 3 levels "Delo","Chinstrap",..: 1 1 1 1 1 1 1 1 1 1 ..
##  $ island : Factor w/ 3 levels "Biscoe","Delo",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ sex    : Factor w/ 2 levels "female","male": 2 1 1 NA 1 2 1 2 NA NA ...
```

Transformations des données : un exemple

Pivots

```
> pays <- c("Belgique", "France")
> pop1992 <- c(10045622,    57374179)
> pop1997 <- c(10199787,    58623428)
> d <- tibble(pays, pop1992, pop1997)
> d %>% pivot_longer(c("pop1992","pop1997"))
```

```
## # A tibble: 4 x 3
##   pays     name       value
##   <chr>    <chr>      <dbl>
## 1 Belgique pop1992 10045622
## 2 Belgique pop1997 10199787
## 3 France   pop1992 57374179
## 4 France   pop1997 58623428
```

Manipulations des chaînes de caractères : un exemple

Concaténation et séparations simples

```
> d <- penguins %>% unite(new , island, species)
> d %>% slice(1:3)
```

```
## # A tibble: 3 x 7
##   new               bill_l bill_d flip_l  mass sex     year
##   <chr>              <dbl>  <dbl>  <int> <int> <fct>  <int>
## 1 Torgersen_Adelie   39.1   18.7     181  3750 male    2007
## 2 Torgersen_Adelie   39.5   17.4     186  3800 female  2007
## 3 Torgersen_Adelie   40.3   18       195  3250 female  2007
```

```
> d %>% separate(new, c("ile", "espece")) %>% slice(1:3)
```

```
## # A tibble: 3 x 8
##   ile       espece bill_l bill_d flip_l  mass sex     year
##   <chr>     <chr>   <dbl>  <dbl>  <int> <int> <fct>  <int>
## 1 Torgersen Adelie   39.1   18.7     181  3750 male    2007
## 2 Torgersen Adelie   39.5   17.4     186  3800 female  2007
## 3 Torgersen Adelie   40.3   18       195  3250 female  2007
```

Manipulations des dates : un exemple

Transcription facile

```
> ymd(20101215)

## [1] "2010-12-15"
> ymd(101215)

## [1] "2010-12-15"
> mdy("4/1/17")

## [1] "2017-04-01"
> mdy("4-1-17")

## [1] "2017-04-01"
> mdy("04-1-2017")

## [1] "2017-04-01"
```

Mise en forme de tableaux : un exemple

```
> anvmod1 <- anova( lm(data= penguins, flip_l ~ .) )
> anvmod1 <- as_tibble(cbind(Effect = row.names(anvmod1),anvmod1))
> anvmod1 %>% mutate_if(is.numeric, list(~as.character(signif(., 3)))) %>%
+   kbl() %>%
+   kable_paper(full_width = F) %>%
+   column_spec(1,  background = 'cyan') %>%
+   row_spec(which.max(anvmod1$`Pr(>F)`), color  = 'white',
+            background = 'teal') %>%
+   column_spec(4, color  = spec_color(anvmod1$`Mean Sq`))
```

Effect	Df	Sum Sq	Mean Sq	F value	Pr(>F)
species	2	50500	25300	1080	1.13e-143
island	2	173	86.4	3.69	0.0261
bill_l	1	3490	3490	149	2.02e-28
bill_d	1	1100	1100	47	3.55e-11
mass	1	1150	1150	48.9	1.55e-11
sex	1	14.4	14.4	0.614	0.434
year	1	1200	1200	51.2	5.6e-12
Residuals	323	7570	23.4		

Cheat sheets

- **readr** https://raw.githubusercontent.com/rstudio/cheatsheets/master/pngs/thumbnails/data-import-cheatsheet-thumbs.png
- **tidyr** https://raw.githubusercontent.com/rstudio/cheatsheets/master/pngs/thumbnails/data-import-cheatsheet-thumbs.png
- **stringr** https://raw.githubusercontent.com/rstudio/cheatsheets/master/pngs/thumbnails/strings-cheatsheet-thumbs.png
- **forcats** https://raw.githubusercontent.com/rstudio/cheatsheets/master/pngs/thumbnails/forcats-cheatsheet-thumbs.png
- **dplyr** https://raw.githubusercontent.com/rstudio/cheatsheets/master/pngs/thumbnails/data-transformation-cheatsheet-thumbs.png
- **ggplot2** https://raw.githubusercontent.com/rstudio/cheatsheets/master/pngs/thumbnails/data-visualization-cheatsheet-thumbs.png
- **purrr** https://raw.githubusercontent.com/rstudio/cheatsheets/master/pngs/thumbnails/purrr-cheatsheet-thumbs.png
- **lubridate** https://raw.githubusercontent.com/rstudio/cheatsheets/master/pngs/thumbnails/lubridate-cheatsheet-thumbs.png