Manipulating data

This guide is partly based on online material from Amy Willis, Kiirsti Owen and Amelia McNamara, and the book “R for Data Science” by Hadley Wickham and Garrett Grolemund. Thank you amazing R community!

### Load packages

We will be using the readr, tidyr and dplyr packages from the Tidyverse family of packages. We will also load the “here” package that we will use to read in our data.

library(readr)

## Warning: package 'readr' was built under R version 4.4.3

library(tidyr)

## Warning: package 'tidyr' was built under R version 4.4.3

library(dplyr)

## Warning: package 'dplyr' was built under R version 4.4.3

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

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

library(here)

## Warning: package 'here' was built under R version 4.4.3

## here() starts at C:/Users/ECliffe ABDN/OneDrive/Documents/IntroHDS/GitHub/Intro2hdsR

### First, let’s practice with pivoting

We will start with a toy non-tidy dataset:

patient\_ID <- c(1,2)  
test\_result\_month1 <- c("a1" ,"a2")  
test\_result\_month2 <- c("b1" ,"b2")  
test\_result\_month3 <- c("c1" ,"c2")  
  
patient\_tests <- data.frame(patient\_ID,test\_result\_month1,test\_result\_month2,test\_result\_month3)  
  
patient\_tests

## patient\_ID test\_result\_month1 test\_result\_month2 test\_result\_month3  
## 1 1 a1 b1 c1  
## 2 2 a2 b2 c2

The dataset is not tidy because each row contains three observations, one per month. A tidy dataset has one observation per row. To do this, we use pivot\_longer.

* The first argument is the dataset to reshape, but as we are using the pipe (%>%) we are skipping the first argument.
* The next argument describes which columns need to be reshaped. In this case, it’s every column apart from patient\_ID.
* The names\_to gives the name of the variable that will be created from the data stored in the column names, in this case the month.
* The values\_to gives the name of the variable that will be created from the data stored in the cell value, in this case the test result.

tidy\_patient\_tests <- patient\_tests %>%  
 pivot\_longer(  
 c('test\_result\_month1','test\_result\_month2','test\_result\_month3'),  
 names\_to= 'month',  
 values\_to='test\_result'  
 )  
  
tidy\_patient\_tests

## # A tibble: 6 × 3  
## patient\_ID month test\_result  
## <dbl> <chr> <chr>   
## 1 1 test\_result\_month1 a1   
## 2 1 test\_result\_month2 b1   
## 3 1 test\_result\_month3 c1   
## 4 2 test\_result\_month1 a2   
## 5 2 test\_result\_month2 b2   
## 6 2 test\_result\_month3 c2

As you can see, the data frame is now tidy (one observation per row), but it would be better if the “month” column just contained the month number (1,2,3). To do this we can add the arguments names\_prefix to strip off the test\_result\_month prefix, and names\_transform to convert month into an integer:

tidy\_patient\_tests <- patient\_tests %>%  
 pivot\_longer(  
 c('test\_result\_month1','test\_result\_month2','test\_result\_month3'),  
 names\_to= 'month',  
 names\_prefix = 'test\_result\_month',  
 names\_transform = list(month = as.integer),  
 values\_to='test\_result'  
 )  
tidy\_patient\_tests

## # A tibble: 6 × 3  
## patient\_ID month test\_result  
## <dbl> <int> <chr>   
## 1 1 1 a1   
## 2 1 2 b1   
## 3 1 3 c1   
## 4 2 1 a2   
## 5 2 2 b2   
## 6 2 3 c2

### Reading in the FEV data

We will use the same data as last week. So read in the data from file fev.csv and save it in an object called fev\_data:

# Add your code here!  
fev\_data <- read\_csv(here("./Inputs/fev.csv"))

## Rows: 654 Columns: 7  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## dbl (7): seqnbr, subjid, age, fev, height, sex, smoke  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

Tip: If you got an error that “fev.csv” does not exist, check that you are working in the correct directory!

### Operating on data: subsets

To select subsets of the data (not just columns with $) use square brackets:

fev\_data$fev[32] # 32nd element of the fev column

## [1] 3

fev\_data[32,3] # 32nd element of the 3rd column

## # A tibble: 1 × 1  
## age  
## <dbl>  
## 1 9

fev\_data[32,"age"] # Same thing, but using the name of the 3rd column - better, as it is more readable and robust

## # A tibble: 1 × 1  
## age  
## <dbl>  
## 1 9

fev\_data[32, ] # Everything in the 3rd row

## # A tibble: 1 × 7  
## seqnbr subjid age fev height sex smoke  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 32 7201 9 3 65.5 1 0

fev\_data[32,1:3]

## # A tibble: 1 × 3  
## seqnbr subjid age  
## <dbl> <dbl> <dbl>  
## 1 32 7201 9

fev\_data[32,-5]

## # A tibble: 1 × 6  
## seqnbr subjid age fev sex smoke  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 32 7201 9 3 1 0

fev\_data[32,-1:-2]

## # A tibble: 1 × 5  
## age fev height sex smoke  
## <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 9 3 65.5 1 0

fev\_data[32,c(1,3,5)] #c(1,3,5) is a vector of numbers (c means "combine")

## # A tibble: 1 × 3  
## seqnbr age height  
## <dbl> <dbl> <dbl>  
## 1 32 9 65.5

c(1,3,5) %>%  
 length

## [1] 3

–> How would you drop the 1st, 3rd and 5th column?

# Add your code here!  
fev\_data[,c(2,4,6,7)]

## # A tibble: 654 × 4  
## subjid fev sex smoke  
## <dbl> <dbl> <dbl> <dbl>  
## 1 301 1.71 0 0  
## 2 451 1.72 0 0  
## 3 501 1.72 0 0  
## 4 642 1.56 1 0  
## 5 901 1.90 1 0  
## 6 1701 2.34 0 0  
## 7 1752 1.92 0 0  
## 8 1753 1.42 0 0  
## 9 1901 1.99 0 0  
## 10 1951 1.94 0 0  
## # ℹ 644 more rows

### Logicals

Besides numbers and strings of characters, R also stores logicals - TRUE and FALSE

Example: a new vector with elements that are TRUE if height is above 72 cm and FALSE otherwise:

is\_tall <- fev\_data$height > 72

Useful summary command:

table(is\_tall)

## is\_tall  
## FALSE TRUE   
## 647 7

Which subjects in fev\_data are tall?

fev\_data[is\_tall,]

## # A tibble: 7 × 7  
## seqnbr subjid age fev height sex smoke  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 401 18841 14 4.27 72.5 1 0  
## 2 450 32741 13 4.22 74 1 0  
## 3 464 37241 13 4.88 73 1 0  
## 4 517 49541 13 5.08 74 1 0  
## 5 550 59941 14 4.27 72.5 1 0  
## 6 632 37441 17 5.63 73 1 0  
## 7 636 44241 16 3.64 73.5 1 0

### Filtering (selecting rows)

fev\_data %>%  
 filter(height > 72)

## # A tibble: 7 × 7  
## seqnbr subjid age fev height sex smoke  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 401 18841 14 4.27 72.5 1 0  
## 2 450 32741 13 4.22 74 1 0  
## 3 464 37241 13 4.88 73 1 0  
## 4 517 49541 13 5.08 74 1 0  
## 5 550 59941 14 4.27 72.5 1 0  
## 6 632 37441 17 5.63 73 1 0  
## 7 636 44241 16 3.64 73.5 1 0

fev\_data %>%  
 filter(age == 6)

## # A tibble: 37 × 7  
## seqnbr subjid age fev height sex smoke  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 7 1752 6 1.92 58 0 0  
## 2 8 1753 6 1.42 56 0 0  
## 3 11 1952 6 1.60 53 0 0  
## 4 18 3551 6 1.88 53 0 0  
## 5 49 10841 6 1.65 55 1 0  
## 6 55 12241 6 1.63 54 1 0  
## 7 63 14251 6 1.48 51 0 0  
## 8 66 14541 6 1.75 57.5 1 0  
## 9 80 16151 6 1.72 53 0 0  
## 10 82 16252 6 1.70 53 0 0  
## # ℹ 27 more rows

fev\_data %>%  
 filter(age != 20)

## # A tibble: 654 × 7  
## seqnbr subjid age fev height sex smoke  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 1 301 9 1.71 57 0 0  
## 2 2 451 8 1.72 67.5 0 0  
## 3 3 501 7 1.72 54.5 0 0  
## 4 4 642 9 1.56 53 1 0  
## 5 5 901 9 1.90 57 1 0  
## 6 6 1701 8 2.34 61 0 0  
## 7 7 1752 6 1.92 58 0 0  
## 8 8 1753 6 1.42 56 0 0  
## 9 9 1901 8 1.99 58.5 0 0  
## 10 10 1951 9 1.94 60 0 0  
## # ℹ 644 more rows

fev\_data %>%  
 filter(age <= 20)

## # A tibble: 654 × 7  
## seqnbr subjid age fev height sex smoke  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 1 301 9 1.71 57 0 0  
## 2 2 451 8 1.72 67.5 0 0  
## 3 3 501 7 1.72 54.5 0 0  
## 4 4 642 9 1.56 53 1 0  
## 5 5 901 9 1.90 57 1 0  
## 6 6 1701 8 2.34 61 0 0  
## 7 7 1752 6 1.92 58 0 0  
## 8 8 1753 6 1.42 56 0 0  
## 9 9 1901 8 1.99 58.5 0 0  
## 10 10 1951 9 1.94 60 0 0  
## # ℹ 644 more rows

You can also filter by whether data are not a number (na):

fev\_data %>%  
 filter(is.na(age)) # opposite: !is.na(age)

## # A tibble: 0 × 7  
## # ℹ 7 variables: seqnbr <dbl>, subjid <dbl>, age <dbl>, fev <dbl>,  
## # height <dbl>, sex <dbl>, smoke <dbl>

You can combine multiple expressions with Boolean operators: & is “and”, | is “or”, and ! is “not”

fev\_data %>%  
 filter(age == 14 & smoke !=0) # age is 14 AND smoker

## # A tibble: 7 × 7  
## seqnbr subjid age fev height sex smoke  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 332 4952 14 2.24 66 0 1  
## 2 358 10053 14 3.43 64 0 1  
## 3 370 11642 14 3.96 72 1 1  
## 4 384 15751 14 3.07 65 0 1  
## 5 439 30042 14 4.31 69 1 1  
## 6 556 61941 14 2.28 66 1 1  
## 7 602 82743 14 4.76 68 1 1

fev\_data %>%  
 filter(age < 5 | height < 50) # younger than 5 OR shorter than 50 cm

## # A tibble: 18 × 7  
## seqnbr subjid age fev height sex smoke  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 21 4351 5 1.4 49 0 0  
## 2 23 5152 4 0.839 48 0 0  
## 3 26 5642 3 1.40 51.5 1 0  
## 4 31 6851 5 1.28 49 0 0  
## 5 59 13751 4 1.57 50 0 0  
## 6 64 14252 4 1.58 49 0 0  
## 7 104 23841 4 0.796 47 1 0  
## 8 118 28551 5 1.20 46.5 0 0  
## 9 157 38242 6 1.54 48 1 0  
## 10 173 40541 4 1.79 52 1 0  
## 11 181 43242 7 1.16 47 1 0  
## 12 216 49551 4 1.10 48 0 0  
## 13 222 50951 3 1.07 46 0 0  
## 14 225 51341 6 1.42 49.5 1 0  
## 15 233 54751 4 1.39 48 0 0  
## 16 286 75951 4 1.42 49 0 0  
## 17 299 80841 4 1.00 48 1 0  
## 18 300 81241 6 1.43 49.5 1 0

Rules for filtering for categorical data: sex == “F” or sex != “F” sex %in% c(“M”,“F”)

### Selecting columns

fev\_data %>%  
 select(fev, height, age)

## # A tibble: 654 × 3  
## fev height age  
## <dbl> <dbl> <dbl>  
## 1 1.71 57 9  
## 2 1.72 67.5 8  
## 3 1.72 54.5 7  
## 4 1.56 53 9  
## 5 1.90 57 9  
## 6 2.34 61 8  
## 7 1.92 58 6  
## 8 1.42 56 6  
## 9 1.99 58.5 8  
## 10 1.94 60 9  
## # ℹ 644 more rows

fev\_data %>%  
 select(-seqnbr, -subjid)

## # A tibble: 654 × 5  
## age fev height sex smoke  
## <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 9 1.71 57 0 0  
## 2 8 1.72 67.5 0 0  
## 3 7 1.72 54.5 0 0  
## 4 9 1.56 53 1 0  
## 5 9 1.90 57 1 0  
## 6 8 2.34 61 0 0  
## 7 6 1.92 58 0 0  
## 8 6 1.42 56 0 0  
## 9 8 1.99 58.5 0 0  
## 10 9 1.94 60 0 0  
## # ℹ 644 more rows

### Summarising data

fev\_data %>%  
 filter(age == 14 & smoke != 0) %>%  
 summarise(mean(fev))

## # A tibble: 1 × 1  
## `mean(fev)`  
## <dbl>  
## 1 3.43

You can name the summary variable:

fev\_data %>%  
 filter(age == 14 & smoke != 0) %>%  
 summarise(my\_mean = mean(fev))

## # A tibble: 1 × 1  
## my\_mean  
## <dbl>  
## 1 3.43

fev\_data %>%  
 filter(age == 14 & smoke != 0) %>%  
 summarise(mean(fev), sd(fev))

## # A tibble: 1 × 2  
## `mean(fev)` `sd(fev)`  
## <dbl> <dbl>  
## 1 3.43 0.976

To get the average FEV for both smokers and non-smokers we don’t need to repeat for smoke==0. We can create a grouping variable:

fev\_data %>%  
 group\_by(smoke)

## # A tibble: 654 × 7  
## # Groups: smoke [2]  
## seqnbr subjid age fev height sex smoke  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 1 301 9 1.71 57 0 0  
## 2 2 451 8 1.72 67.5 0 0  
## 3 3 501 7 1.72 54.5 0 0  
## 4 4 642 9 1.56 53 1 0  
## 5 5 901 9 1.90 57 1 0  
## 6 6 1701 8 2.34 61 0 0  
## 7 7 1752 6 1.92 58 0 0  
## 8 8 1753 6 1.42 56 0 0  
## 9 9 1901 8 1.99 58.5 0 0  
## 10 10 1951 9 1.94 60 0 0  
## # ℹ 644 more rows

(Same exact data, it just prints the two groups)

fev\_data %>%  
 group\_by(smoke) %>%  
 summarise(mean(fev), sd(fev))

## # A tibble: 2 × 3  
## smoke `mean(fev)` `sd(fev)`  
## <dbl> <dbl> <dbl>  
## 1 0 2.57 0.851  
## 2 1 3.28 0.750

But what is the size of each group? n() gives us the number of observations in each group:

fev\_data %>%  
 group\_by(smoke) %>%  
 summarise(n = n(), mean = mean(fev), sd = sd(fev))

## # A tibble: 2 × 4  
## smoke n mean sd  
## <dbl> <int> <dbl> <dbl>  
## 1 0 589 2.57 0.851  
## 2 1 65 3.28 0.750

You can also group by your own variables:

fev\_data %>%  
 group\_by(height < 60) %>%  
 summarise(n(), mean(fev))

## # A tibble: 2 × 3  
## `height < 60` `n()` `mean(fev)`  
## <lgl> <int> <dbl>  
## 1 FALSE 409 3.10  
## 2 TRUE 245 1.86

A useful function: arrange

fev\_data %>%  
 group\_by(age) %>%  
 summarise(n(), mean(fev)) %>%  
 arrange(age) # arrange by increasing age

## # A tibble: 17 × 3  
## age `n()` `mean(fev)`  
## <dbl> <int> <dbl>  
## 1 3 2 1.24  
## 2 4 9 1.28  
## 3 5 28 1.55  
## 4 6 37 1.66  
## 5 7 54 1.87  
## 6 8 85 2.12  
## 7 9 94 2.43  
## 8 10 81 2.69  
## 9 11 90 3.04  
## 10 12 57 3.22  
## 11 13 43 3.48  
## 12 14 25 3.58  
## 13 15 19 3.48  
## 14 16 13 3.67  
## 15 17 8 4.30  
## 16 18 6 3.59  
## 17 19 3 3.99

fev\_data %>%  
 group\_by(age) %>%  
 summarise(n(), mean(fev)) %>%  
 arrange(desc(age)) # arrange by decreasing age

## # A tibble: 17 × 3  
## age `n()` `mean(fev)`  
## <dbl> <int> <dbl>  
## 1 19 3 3.99  
## 2 18 6 3.59  
## 3 17 8 4.30  
## 4 16 13 3.67  
## 5 15 19 3.48  
## 6 14 25 3.58  
## 7 13 43 3.48  
## 8 12 57 3.22  
## 9 11 90 3.04  
## 10 10 81 2.69  
## 11 9 94 2.43  
## 12 8 85 2.12  
## 13 7 54 1.87  
## 14 6 37 1.66  
## 15 5 28 1.55  
## 16 4 9 1.28  
## 17 3 2 1.24

Sorting columns

fev\_data$age %>% sort #Sort a column

## [1] 3 3 4 4 4 4 4 4 4 4 4 5 5 5 5 5 5 5 5 5 5 5 5 5 5  
## [26] 5 5 5 5 5 5 5 5 5 5 5 5 5 5 6 6 6 6 6 6 6 6 6 6 6  
## [51] 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6  
## [76] 6 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7  
## [101] 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7  
## [126] 7 7 7 7 7 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8  
## [151] 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8  
## [176] 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8  
## [201] 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 9 9 9 9 9 9 9 9 9 9  
## [226] 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9  
## [251] 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9  
## [276] 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9  
## [301] 9 9 9 9 9 9 9 9 9 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10  
## [326] 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10  
## [351] 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10  
## [376] 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 11 11 11 11 11 11 11 11 11 11  
## [401] 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11  
## [426] 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11  
## [451] 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11  
## [476] 11 11 11 11 11 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12  
## [501] 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12  
## [526] 12 12 12 12 12 12 12 12 12 12 12 12 13 13 13 13 13 13 13 13 13 13 13 13 13  
## [551] 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13  
## [576] 13 13 13 13 13 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  
## [601] 14 14 14 14 14 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 16  
## [626] 16 16 16 16 16 16 16 16 16 16 16 16 17 17 17 17 17 17 17 17 18 18 18 18 18  
## [651] 18 19 19 19

fev\_data$age %>% unique %>% sort # Sort unique values in a column

## [1] 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

table() gives you a count of a particular factor or combination of factor levels:

table(fev\_data$age)

##   
## 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19   
## 2 9 28 37 54 85 94 81 90 57 43 25 19 13 8 6 3

table(fev\_data$age,fev\_data$smoke)

##   
## 0 1  
## 3 2 0  
## 4 9 0  
## 5 28 0  
## 6 37 0  
## 7 54 0  
## 8 85 0  
## 9 93 1  
## 10 76 5  
## 11 81 9  
## 12 50 7  
## 13 30 13  
## 14 18 7  
## 15 9 10  
## 16 6 7  
## 17 6 2  
## 18 4 2  
## 19 1 2

–> **Problem 1**: Which subjects are male and which are female? (i.e. what does sex == 1 mean?)

fev\_data %>% filter(age>=18) %>% group\_by(sex) %>% summarise(mean(height))

## # A tibble: 2 × 2  
## sex `mean(height)`  
## <dbl> <dbl>  
## 1 0 64.4  
## 2 1 69.4

#I think that sex == 1 is the males

–> **Problem 2**: Why do smokers appear to have better lung function (higher forced expiratory volume - FEV)?

# Add your code here!  
library(ggplot2)

## Warning: package 'ggplot2' was built under R version 4.4.3

fev\_data <- read\_csv(here("./Inputs/fev.csv"))

## Rows: 654 Columns: 7  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## dbl (7): seqnbr, subjid, age, fev, height, sex, smoke  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

fev\_data %>% group\_by(smoke,age) %>% summarise(mean(fev))

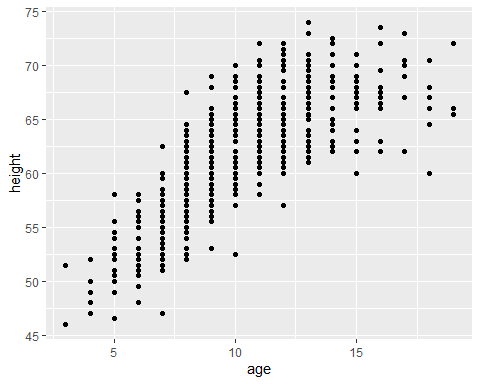
## `summarise()` has grouped output by 'smoke'. You can override using the  
## `.groups` argument.

## # A tibble: 28 × 3  
## # Groups: smoke [2]  
## smoke age `mean(fev)`  
## <dbl> <dbl> <dbl>  
## 1 0 3 1.24  
## 2 0 4 1.28  
## 3 0 5 1.55  
## 4 0 6 1.66  
## 5 0 7 1.87  
## 6 0 8 2.12  
## 7 0 9 2.44  
## 8 0 10 2.67  
## 9 0 11 3.03  
## 10 0 12 3.24  
## # ℹ 18 more rows

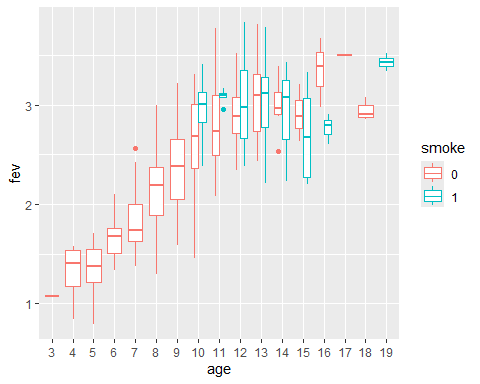
fev\_data %>% ungroup()

## # A tibble: 654 × 7  
## seqnbr subjid age fev height sex smoke  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 1 301 9 1.71 57 0 0  
## 2 2 451 8 1.72 67.5 0 0  
## 3 3 501 7 1.72 54.5 0 0  
## 4 4 642 9 1.56 53 1 0  
## 5 5 901 9 1.90 57 1 0  
## 6 6 1701 8 2.34 61 0 0  
## 7 7 1752 6 1.92 58 0 0  
## 8 8 1753 6 1.42 56 0 0  
## 9 9 1901 8 1.99 58.5 0 0  
## 10 10 1951 9 1.94 60 0 0  
## # ℹ 644 more rows

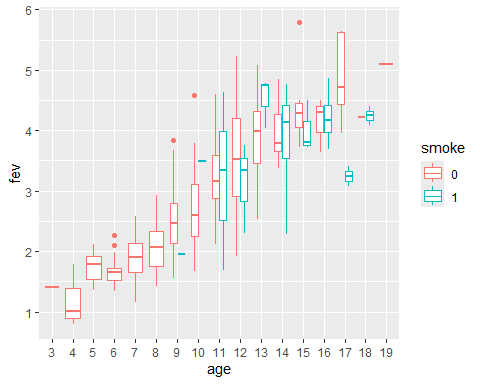
fev\_data %>%   
 ggplot() +   
 geom\_point(aes(x=age, y=height))



fev\_data$age <- factor(fev\_data$age)  
fev\_data$smoke <- factor(fev\_data$smoke)  
fev\_data %>% filter(sex==0) %>%   
 ggplot() +  
 geom\_boxplot(aes(x = age, y = fev, colour = smoke))



fev\_data %>% filter(sex==1) %>%   
 ggplot() +  
 geom\_boxplot(aes(x = age, y = fev, colour = smoke))



#Because they are older and taller and have larger lungs

Useful function: rename

fev\_data %>%  
 rename(ID = subjid)

## # A tibble: 654 × 7  
## seqnbr ID age fev height sex smoke  
## <dbl> <dbl> <fct> <dbl> <dbl> <dbl> <fct>  
## 1 1 301 9 1.71 57 0 0   
## 2 2 451 8 1.72 67.5 0 0   
## 3 3 501 7 1.72 54.5 0 0   
## 4 4 642 9 1.56 53 1 0   
## 5 5 901 9 1.90 57 1 0   
## 6 6 1701 8 2.34 61 0 0   
## 7 7 1752 6 1.92 58 0 0   
## 8 8 1753 6 1.42 56 0 0   
## 9 9 1901 8 1.99 58.5 0 0   
## 10 10 1951 9 1.94 60 0 0   
## # ℹ 644 more rows

Mutate: compute new column

fev\_data %>%  
 mutate(heightdiff = height - mean(height))

## # A tibble: 654 × 8  
## seqnbr subjid age fev height sex smoke heightdiff  
## <dbl> <dbl> <fct> <dbl> <dbl> <dbl> <fct> <dbl>  
## 1 1 301 9 1.71 57 0 0 -4.14   
## 2 2 451 8 1.72 67.5 0 0 6.36   
## 3 3 501 7 1.72 54.5 0 0 -6.64   
## 4 4 642 9 1.56 53 1 0 -8.14   
## 5 5 901 9 1.90 57 1 0 -4.14   
## 6 6 1701 8 2.34 61 0 0 -0.144  
## 7 7 1752 6 1.92 58 0 0 -3.14   
## 8 8 1753 6 1.42 56 0 0 -5.14   
## 9 9 1901 8 1.99 58.5 0 0 -2.64   
## 10 10 1951 9 1.94 60 0 0 -1.14   
## # ℹ 644 more rows

Remember that to save these changes you need to assign to a new tibble:

new\_fev\_data <- fev\_data %>%  
 rename(id = subjid) %>%  
 mutate(heightdiff = height - mean(height))