Lab 3: Modeling correlation and regression

Practice session covering topics discussed in Lecture 3

M. Chiara Mimmi, Ph.D. | Università degli Studi di Pavia

July 26, 2024

GOAL OF TODAY'S PRACTICE SESSION

- Review the basic questions we can ask about ASSOCIATION between any two variables:
  - does it exist?
  - how strong is it?
  - what is its direction?
- Introduce a widely used analytical tool: REGRESSION

The examples and code from this lab session follow very closely the open access book:

- Vu, J., & Harrington, D. (2021). **Introductory Statistics for the Life and Biomedical Sciences**. https://www.openintro.org/book/biostat/

Topics discussed in Lecture # 3

Lecture 3: topics

- Testing and summarizing relationship between 2 variables (**correlation**)
  - Pearson's ***r*** analysis (param)
  - Spearman test (no param)
- Measures of **association**
  - Chi-Square test of independence
  - Fisher's Exact Test
    - alternative to the Chi-Square Test of Independence
- From correlation/association to **prediction/causation**
  - The purpose of observational and experimental studies
- Widely used analytical tools
  - Simple linear regression models
  - Multiple Linear Regression models
- Shifting the emphasis on **empirical prediction**
  - Introduction to Machine Learning (ML)
  - Distinction between Supervised & Unsupervised algorithms

R ENVIRONMENT SET UP & DATA

Needed R Packages

- We will use functions from packages `base`, `utils`, and `stats` (pre-installed and pre-loaded)
- We will also use the packages below (specifying `package::function` for clarity).

```
# Load pckgs for this R session

# -- General
library(fs)      # file/directory interactions
library(here)    # tools find your project's files, based on working directory
library(paint) # paint data.frames summaries in colour
library(janitor) # tools for examining and cleaning data
library(dplyr)   # {tidyverse} tools for manipulating and summarizing tidy data
library(forcats) # {tidyverse} tool for handling factors
library(openxlsx) # Read, Write and Edit xlsx Files
library(flextable) # Functions for Tabular Reporting
# -- Statistics
library(rstatix) # Pipe-Friendly Framework for Basic Statistical Tests
library(lmtest) # Testing Linear Regression Models
library(broom) # Convert Statistical Objects into Tidy Tibbles
#library(tidymodels) # not installed on this machine
library(performance) # Assessment of Regression Models Performance
# -- Plotting
library(ggplot2) # Create Elegant Data Visualisations Using the Grammar of Graphics
```

DATASETS FOR TODAY

We will use examples (with adapted datasets) from real clinical studies, provided among the learning materials of the open access books:

- Vu, J., & Harrington, D. (2021). **Introductory Statistics for the Life and Biomedical Sciences**. https://www.openintro.org/book/biostat/
- Çetinkaya-Rundel, M., & Hardim, J. (2023). **Introduction to Modern Statistics (1st Ed)**. https://openintro-ims.netlify.app/

Importing Dataset 1 (NHANES)

Name: NHANES (National Health and Nutrition Examination Survey) combines interviews and physical examinations to assess the health and nutritional status of adults and children in the United States. Sterted in the 1960s, it became a continuous program in 1999.
Documentation: dataset1
Sampling details: Here we use a sample of 500 adults from NHANES 2009-2010 & 2011-2012 (`nhanes.samp.adult.500` in the R `oibiostat` package, which has been adjusted so that it can be viewed as a random sample of the US population)

```
# Check my working directory location
# here::here()

# Use `here` in specifying all the subfolders AFTER the working directory
nhanes_samp <- read.csv(file = here::here("practice", "data_input", "03_datasets",
                                          "nhanes.samp.csv"),
                        header = TRUE, # 1st line is the name of the variables
                        sep = ",", # which is the field separator character.
                        na.strings = c("?","NA" ), # specific MISSING values
                        row.names = NULL)
```

- Adapting the function `here` to match your own folder structure

NHANES Variables and their description

[EXCERPT: see complete file in Input Data Folder]

Variable	Type	Description
X	int	xxxx
ID	int	xxxxx
SurveyYr	chr	yyyy_mm. Ex. 2011_12
Gender	chr	Gender (sex) of study participant coded as male or female
Age	int	##
AgeDecade	chr	yy-yy es 20-29
Education	chr	[>= 20 yro]. Ex. 8thGrade, 9-11thGrade, HighSchool, SomeCollege, or CollegeGrad.
Weight	dbl	Weight in kg
Height	dbl	Standing height in cm. Reported for participants aged 2 years or older.
BMI	dbl	Body mass index (weight/height2 in kg/m2). Reported for participants aged 2 years or older
Pulse	int	60 second pulse rate
DirectChol	dbl	Direct HDL cholesterol in mmol/L. Reported for participants aged 6 years or older
TotChol	dbl	Total HDL cholesterol in mmol/L. Reported for participants aged 6 years or older
Diabetes	chr	Study participant told by a doctor or health professional that they have diabetes
DiabetesAge	int	Age of study participant when first told they had diabetes
HealthGen	chr	Self-reported rating of health: Excellent, Vgood, Good, Fair, or Poor Fair
Alcohol12PlusYr	chr	Participant has consumed at least 12 drinks of any type of alcoholic beverage in any one year
...	...	...

Importing Dataset 2 (PREVEND)

Name: PREVEND (**Prevention of REnal and Vascular END-stage Disease**) is a study which took place in the Netherlands starting in the 1990s, with subsequent follow-ups throughout the 2000s. This dataset is from the third survey, which participants completed in 2003-2006; data is provided for 4,095 individuals who completed cognitive testing.
Documentation: dataset2 and sample dataset variables' codebook
Sampling details: Here we use a sample of 500 adults taken from 4,095 individuals who completed cognitive testing (i.e. the `prevend.samp` dataset in the R `oibiostat` package)

```
# Check my working directory location
# here::here()

# Use `here` in specifying all the subfolders AFTER the working directory
prevend_samp <- read.csv(file = here::here("practice", "data_input", "03_datasets",
                                           "prevend.samp.csv"),
                         header = TRUE, # 1st line is the name of the variables
                         sep = ",", # which is the field separator character.
                         na.strings = c("?","NA" ), # specific MISSING values
                         row.names = NULL)
```

PREVEND Variables and their description

[EXCERPT: see complete file in Input Data Folder]

Variable	Type	Description
X	int	Patient ID
Age	int	Age in years
Gender	int	Expressed as: 0 = males; 1 = females
RFFT	int	Performance on the Ruff Figural Fluency Test. Scores range from 0 (worst) to 175 (best)
VAT	int	Visual Association Test score. Scores may range from 0 (worst) to 12 (best)
Chol	dbl	Total cholesterol, in mmol/L.
HDL	dbl	HDL cholesterol, in mmol/L.
Statin	int	Statin use at enrollment. Numeric vector: 0 = No; 1 = Yes.
CVD	int	History of cardiovascular event. Numeric vector: 0 = No; 1 = Yes
DM	int	Diabetes mellitus status at enrollment. Numeric vector: 0 = No; 1 = Yes
Education	int	Highest level of education. Numeric: 0 primary school; 1 = lower secondary education; 3 = university
Smoking	int	Smoking at enrollment. numeric vector: 0 = No; 1 = Yes
Hypertension	int	Status of hypertension at enrollment. Numeric vector: 0 = No; 1 = Yes
Ethnicity	int	Expressed as: 0 = Western European; 1 = African; 2 = Asian; 3 = Other
...	...	...

Importing Dataset 3 (FAMuSS)

Name: FAMuSS (Functional SNPs Associated with Muscle Size and Strength) examine the association of demographic, physiological and genetic characteristics with muscle strength – including data on race and genotype at a specific locus on the ACTN3 gene (the “sports gene”).
Documentation: dataset3
Sampling details: the DATASET includes 595 observations on 9 variables (`famuss` in the R `oibiostat` package)

```r
# Check my working directory location
# here::here()

# Use `here` in specifying all the subfolders AFTER the working directory
famuss <- read.csv(file = here::here("practice", "data_input", "03_datasets",
                                     "famuss.csv"),
                           header = TRUE, # 1st line is the name of the variables
                           sep = ",", # which is the field separator character.
                           na.strings = c("?","NA" ), # specific MISSING values
                           row.names = NULL)
```

FAMuSS Variables and their description

[See complete file in Input Data Folder]

Variable	Description
X	id
ndrm.ch	Percent change in strength in the non-dominant arm
drm.ch	Percent change in strength in the dominant arm
sex	Sex of the participant
age	Age in years
race	Recorded as African Am (African American), Caucasian, Asian, Hispanic, Other
height	Height in inches
weight	Weight in pounds
actn3.r577x	Genotype at the location r577x in the ACTN3 gene.
bmi	Body Mass Index

CORRELATION

[Using NHANES and FAMuSS datasets]

Explore relationships between two variables

Approaches for summarizing relationships between two variables vary depending on variable types…

- Two **numerical** variables
- Two **categorical** variables
- One **numerical** variable and one **categorical** variable

Two variables x and y are

- *positively associated* if y increases as x increases.
- *negatively associated* if y decreases as x increases.

TWO NUMERICAL VARIABLES (NHANES)

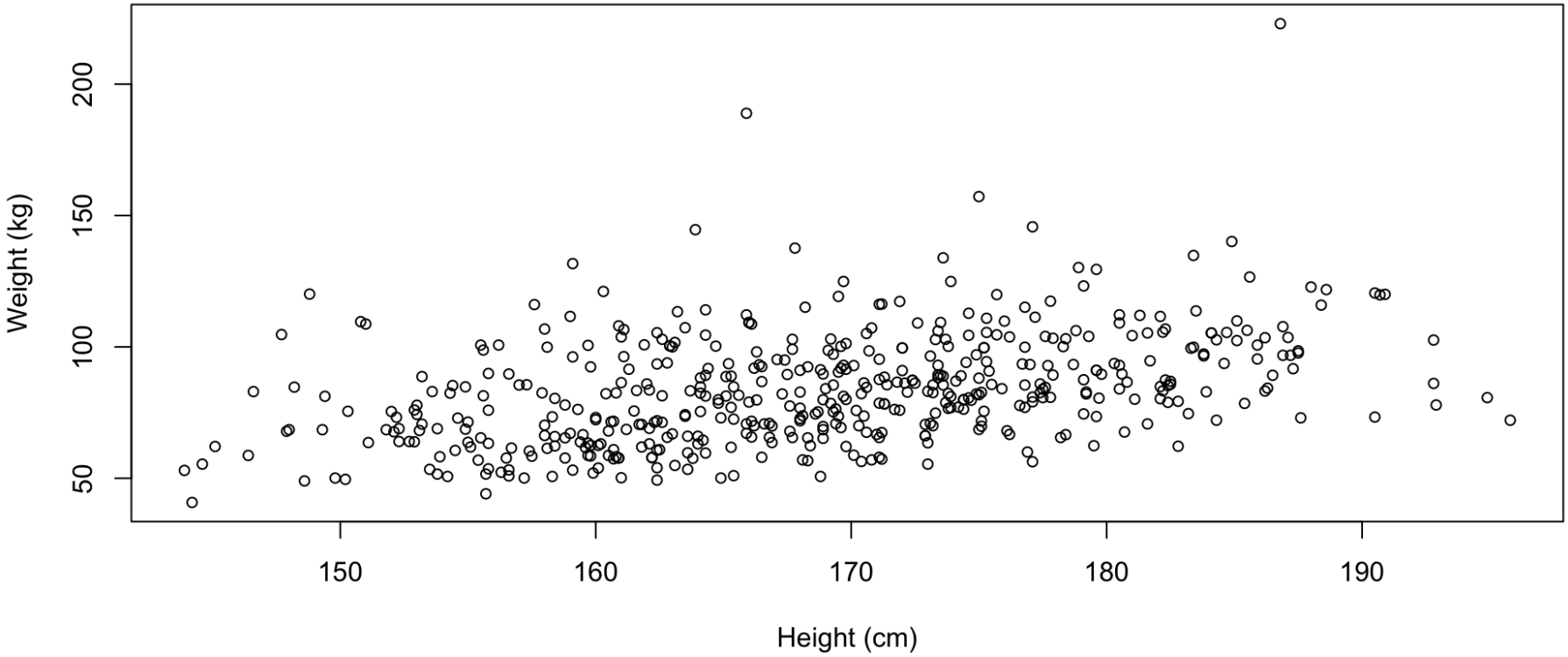
Two numerical variables (plot)

Height and **weight** (taken from the `nhanes_samp` dataset) are positively associated.

- notice we can also use the generic base R function `plot` for a quick scatter plot

```r
# rename for convenience
nhanes <- nhanes_samp %>%
  janitor::clean_names()

# basis plot
plot(nhanes$height, nhanes$weight,
     xlab = "Height (cm)", ylab = "Weight (kg)", cex = 0.8)
```

Two numerical variables (plot)



Two numerical variables: correlation (with `stats::cor`)

Correlation is a numerical summary that measures the strength of a linear relationship between two variables.

- The correlation coefficient r takes on values between -1 and 1.
- The closer r is to ± 1, the stronger the linear association.
- Here we compute the **Pearson rho (parametric)**, with base R function `stats::cor`
  - the `use` argument let us choose how to deal with missing values (in this case only using **all complete pairs**)

```r
is.numeric(nhanes$height)
```

```
[1] TRUE
```

```r
is.numeric(nhanes$weight)
```

```
[1] TRUE
```

```r
# using `stats` package
stats::cor(x = nhanes$height, y =  nhanes$weight,
    # argument for dealing with missing values
    use = "pairwise.complete.obs",
    method = "pearson")
```

```
[1] 0.4102269
```

Two numerical variables: correlation (with `stats::cor.test`)

- Here we compute the **Pearson rho (parametric)**, with the function `cor.test` (the same we used for testing paired samples)
  - implicitely takes care on NAs

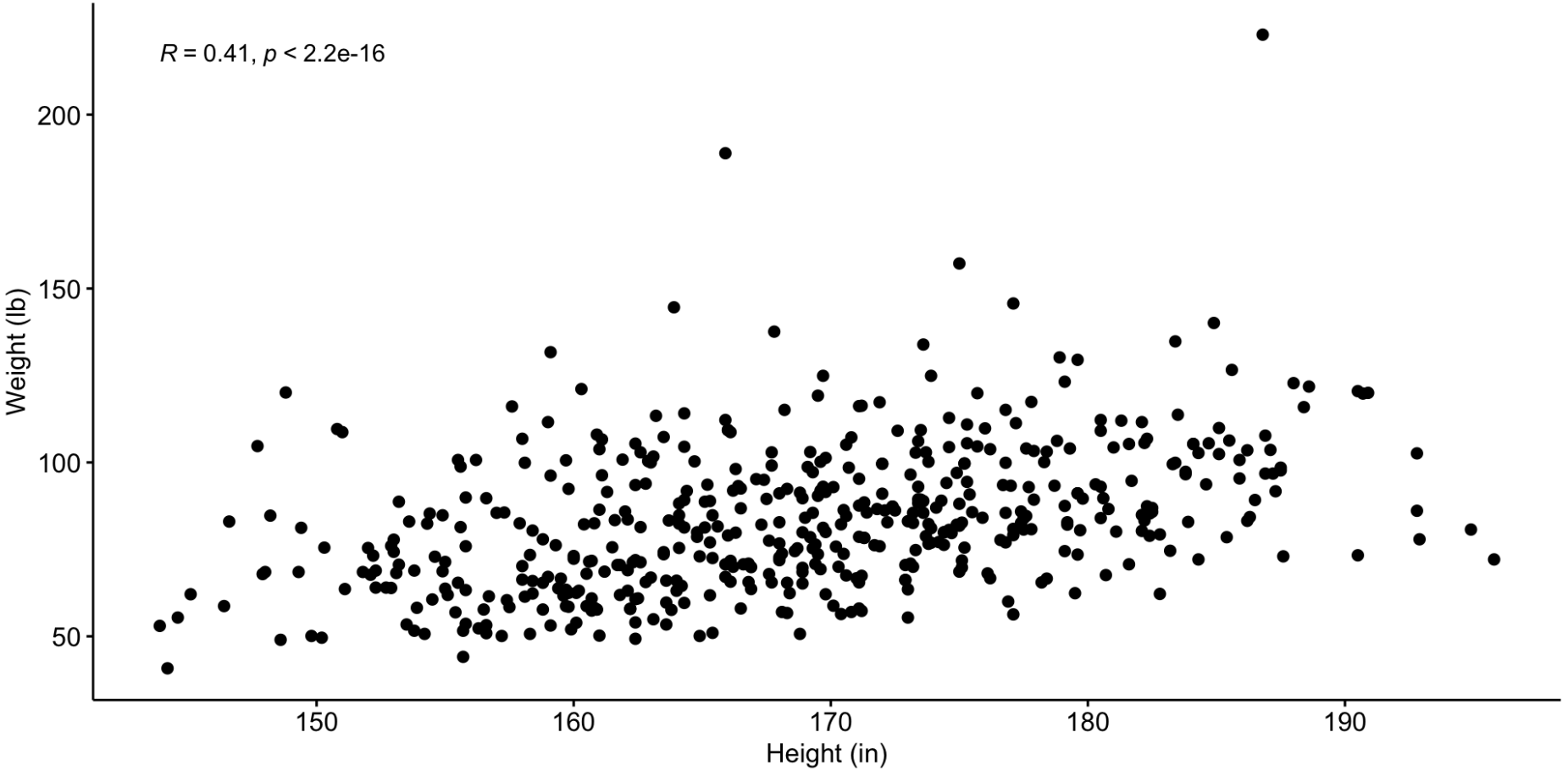
```
# using `stats` package
cor_test_result <- cor.test(x = nhanes$height, y =  nhanes$weight,
                            method = "pearson")

# looking at the cor estimate
cor_test_result[["estimate"]][["cor"]]
```

```
[1] 0.4102269
```

- The function `ggpubr::ggscatter` gives us all in one (scatter plot + r ("R"))! 🤯

```
library("ggpubr") # 'ggplot2' Based Publication Ready Plots
ggpubr::ggscatter(nhanes, x = "height", y = "weight",
                  cor.coef = TRUE, cor.method = "pearson", #cor.coef.coord = 2,
                  xlab = "Height (in)", ylab = "Weight (lb)")
```

Two numerical variables: correlation (with `stats::cor.test`)



Spearman rank-order correlation

The **Spearman's rank-order correlation is the nonparametric version** of the `Pearson` correlation.

Spearman's correlation coefficient, (ϱ, also signified by r_s) measures the strength and direction of association between two ranked variables.

- used when 2 variables have a **non-linear** relationship
- excellent for **ordinal** data (when Pearson's is not appropriate), i.e. Likert scale items

> To compute it, we simply calculate Pearson's correlation of the `rankings` of the raw data (instead of the data).

Spearman rank-order correlation (example)

Let's say we want to get Spearman's correlation with ordinal factors `Education` and `HealthGen` in the `NHANES` sample.

- We have to convert them to their underlying numeric code, to compare rankings.

```
tabyl(nhanes$education)
```

```
nhanes$education   n percent valid_percent
       8th Grade  32   0.064    0.06412826
 9 - 11th Grade  68   0.136    0.13627255
    College Grad 157   0.314    0.31462926
     High School  94   0.188    0.18837675
    Some College 148   0.296    0.29659319
            <NA>   1   0.002            NA
```

```
tabyl(nhanes$health_gen)
```

```
nhanes$health_gen   n percent valid_percent
        Excellent  47   0.094    0.10444444
             Fair  53   0.106    0.11777778
             Good 177   0.354    0.39333333
             Poor  11   0.022    0.02444444
            Vgood 162   0.324    0.36000000
             <NA>  50   0.100            NA
```

```
nhanes <- nhanes %>%
  # reorder education
  mutate (edu_ord = factor (education,
                            levels = c("8th Grade", "9 - 11th Grade",
                                       "High School", "Some College",
                                       "College Grad" , NA))) %>%
  # create edu_rank
  mutate (edu_rank = as.numeric(edu_ord)) %>%
  # reorder health education
  mutate (health_ord = factor (health_gen,
                            levels = c( NA, "Poor", "Fair",
                                       "Good", "Vgood",
                                       "Excellent"))) %>%
  # create health_rank
  mutate (health_rank = as.numeric(health_ord))
```

Spearman rank-order correlation (example), cont.

- Let's check out the `..._rank` version of the 2 categorical variables of interest:
  - **education** from `edu_ord` to `edu_rank`

```
table(nhanes$edu_ord, useNA = "ifany" )
```

```
8th Grade 9 - 11th Grade    High School   Some College   College Grad
       32              68             94            148            157
     <NA>
        1
```

```
table(nhanes$edu_rank, useNA = "ifany" )
```

```
  1   2   3   4   5 <NA>
 32  68  94 148 157    1
```

- **general health** from `health_ord` to `health_rank`

```
table(nhanes$health_ord, useNA = "ifany" )
```

```
     Poor      Fair      Good     Vgood Excellent      <NA>
       11        53       177       162        47        50
```

```
table(nhanes$health_rank,  useNA = "ifany" )
```

```
  1   2   3   4   5 <NA>
 11  53 177 162  47  50
```

Spearman rank-order correlation (example cont.)

After setting up the variables in the correct (numerical rank) format, now we can actually compute it: + same function call `stats::cor.test` + but specifying argument `method = "spearman"`

```
# -- using `stats` package
cor_test_result_sp <- cor.test(x = nhanes$edu_rank,
                               y = nhanes$health_rank,
                               method = "spearman",
                               exact = FALSE) # removes the Ties message warning
# looking at the cor estimate
cor_test_result_sp
```

```
    Spearman's rank correlation rho

data:  nhanes$edu_rank and nhanes$health_rank
S = 10641203, p-value = 1.915e-10
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.2946493
```

```
# -- only print Spearman rho
#cor_test_result_sp[["estimate"]][["rho"]]
```

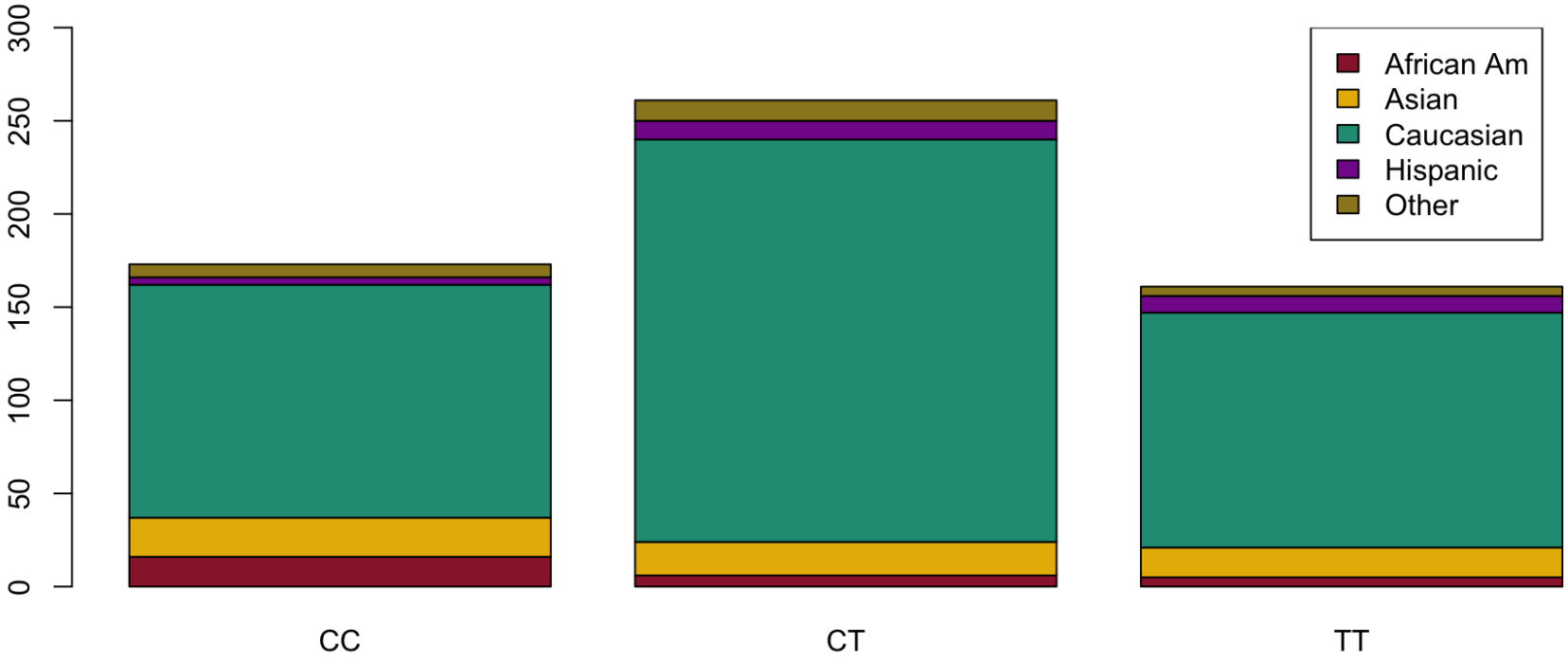
TWO CATEGORICAL VARIABLES (FAMuSS)

Two categorical variables (plot)

In the `famuss` dataset, the variables `race`, and `actn3.r577x` are categorical variables.

- we can use the generic base R function `graphics::barplot`

```
mycolors_contrast <- c("#9b2339", "#E7B800","#239b85", "#85239b", "#9b8523","#23399b", "#d8e600", "#0084e6","#399B23"

## genotypes as columns
genotype.race = matrix(table(famuss$actn3.r577x, famuss$race), ncol=3, byrow=T)
colnames(genotype.race) = c("CC", "CT", "TT")
rownames(genotype.race) = c("African Am", "Asian", "Caucasian", "Hispanic", "Other")

# using generic base::barplot
graphics::barplot(genotype.race, col = mycolors_contrast[1:5], ylim=c(0,300), width=2)
legend("topright", inset=c(.05, 0), fill=mycolors_contrast[1:5],
       legend=rownames(genotype.race))
```


0
50
100
150
200
250
300
CC
CT
TT
African Am
Asian
Caucasian
Hispanic
Other

Two categorical variables (contingency table)

Specifically, the variable `actn3.r577x` takes on three possible levels (`CC`, `CT`, or `TT`) which indicate the distribution of genotype at location `r577x` on the `ACTN3` gene for the FAMuSS study participants.

A **contingency table** summarizes data for two categorical variables.

- the function `stats::addmargins` puts arbitrary *Margins* on multidimensional tables
  - The extra column & row `"Sum"` provide the *marginal totals* across each row and each column, respectively

```
# levels of actn3.r577x
table(famuss$actn3.r577x)
```

```
 CC  CT  TT
173 261 161
```

```
# contingency table to summarize race and actn3.r577x
addmargins(table(famuss$race, famuss$actn3.r577x))
```

```
             CC  CT  TT Sum
  African Am 16   6   5  27
  Asian      21  18  16  55
  Caucasian 125 216 126 467
  Hispanic    4  10   9  23
  Other       7  11   5  23
  Sum       173 261 161 595
```

Two categorical variables (contingency table prop)

Contingency tables can also be converted to show *proportions*. Since there are 2 variables, it is necessary to specify whether the proportions are calculated according to the row variable or the column variable.

- using the `margin =` argument in the `base::prop.table` function (1 indicates rows, 2 indicates columns)

```
# adding row proportions
addmargins(prop.table(table(famuss$race, famuss$actn3.r577x), margin =  1))
```

```
                 CC        CT        TT       Sum
African Am 0.5925926 0.2222222 0.1851852 1.0000000
Asian      0.3818182 0.3272727 0.2909091 1.0000000
Caucasian  0.2676660 0.4625268 0.2698073 1.0000000
Hispanic   0.1739130 0.4347826 0.3913043 1.0000000
Other      0.3043478 0.4782609 0.2173913 1.0000000
Sum        1.7203376 1.9250652 1.3545972 5.0000000
```

```
# adding column proportions
addmargins(prop.table(table(famuss$race, famuss$actn3.r577x),margin =  2))
```

```
                 CC         CT         TT        Sum
African Am 0.09248555 0.02298851 0.03105590 0.14652996
Asian      0.12138728 0.06896552 0.09937888 0.28973168
Caucasian  0.72254335 0.82758621 0.78260870 2.33273826
Hispanic   0.02312139 0.03831418 0.05590062 0.11733618
Other      0.04046243 0.04214559 0.03105590 0.11366392
Sum        1.00000000 1.00000000 1.00000000 3.00000000
```

Chi Squared test of `independence`

The **Chi-squared test** is a hypothesis test used to determine whether there is a relationship between **two categorical variables**.

- categorical vars. can have *nominal* or *ordinal* measurement scale
- the *observed* frequencies are compared with the *expected* frequencies and their deviations are examined.

```r
# Chi-squared test
# (Test of association to see if
# H0: the 2 cat var (race  & actn3.r577x ) are independent
# H1: the 2 cat var are correlated in __some way__

tab <- table(famuss$race, famuss$actn3.r577x)
test_chi <- chisq.test(tab)
```

the obtained result (`test_chi`) is a list of objects…

You try…

…run `View(test_chi)` to check

Chi Squared test of `independence` (cont)

Within `test_chi` results there are:

- `Observed frequencies` = how often a combination occurs in our sample
- `Expected frequencies` = what would it be if the 2 vars were PERFECTLY INDEPENDENT

```
1 # Observed frequencies
2 test_chi$observed
```

```
            CC  CT  TT
African Am  16   6   5
Asian       21  18  16
Caucasian  125 216 126
Hispanic     4  10   9
Other        7  11   5
```

```
1 # Expected frequencies
2 round(test_chi$expected  , digits = 1
```

```
              CC    CT    TT
African Am   7.9  11.8   7.3
Asian       16.0  24.1  14.9
Caucasian  135.8 204.9 126.4
Hispanic     6.7  10.1   6.2
Other        6.7  10.1   6.2
```

Chi Squared test of `independence` (results)

- Recall that:
  - H_0: the 2 cat. var. are **independent**
  - H_1: the 2 cat. var. are **correlated** in some way
- The result of Chi-Square test represents a comparison of the above two tables (*observed* v. *expected*):
  - p-value = 0.01286 smaller than α = 0.05 so **we REJECT the null hypothesis** (i.e. there's likely an association between race and ACTN3 gene)

```
1 test_chi
```

```
    Pearson's Chi-squared test

data:  tab
X-squared = 19.4, df = 8, p-value = 0.01286
```

Computing Cramer's V after test of independence

Recall that **Crammer's V** allows to measure the *effect size* of the test of independence (i.e. the **strength of association** between two nominal variables)

- V ranges from [0 1] (the smaller V, the lower the correlation)

$$V = \sqrt{\frac{\chi^2}{n(k-1)}}$$

where:

- V denotes Cramér's V
- χ^2 is the Pearson chi-square statistic from the prior test
- n is the sample size involved in the test
- k is the lesser number of categories of either variable

Computing Cramer's V after test of independence (2 ways)

- ✍🏻 "By hand" first to see the steps

```r
# Compute Creamer's V by hand

# inputs
chi_calc <- test_chi$statistic
n <- nrow(famuss) # N of obd
n_r <- nrow(test_chi$observed) # number of rows in the contingency table
n_c <- ncol(test_chi$observed) # number of columns in the contingency table

# Cramer's V
sqrt(chi_calc / (n*min(n_r -1, n_c -1)) )
```

```
X-squared
0.1276816
```

- 👩🏻‍💻 Using an R function `rstatix::cramer_v`

```r
# Cramer's V with rstatix
rstatix::cramer_v(test_chi$observed)
```

```
[1] 0.1276816
```

Cramer's V = 0.12, which indicates a relatively weak association between the two categorical variables. It suggests that while there may be some relationship between the variables, it is not particularly strong.

Chi Squared test of `goodness of fit`

In some cases the `Chi-square test` examines **whether or not an observed frequency distribution matches an expected theoretical distribution**.

Here, we are conducting a type of `Chi-square Goodness of Fit Test` which:

- serves to test whether the observed distribution of a categorical variable differs from your expectations
- interprets the statistic based on the discrepancies between observed and expected counts

Chi Squared test of goodness of fit (example)

Since the participants of the **FAMuSS study** where *volunteers* at a university, they did not come from a "representative" sample of the US population, we can use the χ^2 goodness of fit test to test against:

- H_0: the study participants (1st row below) are racially representative of the general population (2nd row below)

Race	African.American	Asian	Caucasian	Other	Total
FAMuSS (Observed)	27	55	467	46	595
US Census (Expected)	76.16	5.95	478.38	34.51	595

We use the formula

$$\chi^2 = \sum_{k} \frac{(\text{Observed} - \text{Expected})^2}{\text{Expected}}$$

Under H_0, the sample proportions should equal the population proportions.

Chi Squared test of goodness of fit (example)

```r
# Subset the vectors of frequencies from the 2 rows
observed <- c(27,  55,  467, 46)
expected <- c(76.2,  5.95, 478.38,  34.51)

# Calculate Chi-Square statistic manually
chi_sq_statistic <- sum((observed - expected)^2 / expected)
df <- length(observed) - 1
p_value <- 1 - pchisq(chi_sq_statistic, df)

# Print results
chi_sq_statistic
```

```
[1] 440.2166
```

```r
df
```

```
[1] 3
```

```r
p_value
```

```
[1] 0
```

The calculated χ^2 statistic is very large, and the `p_value` is close to 0. Hence, there is more than sufficient evidence to **reject the null hypothesis** that the sample is representative of the general population.
Comparing the observed and expected values (or the residuals), we find the **largest discrepancy with the over-representation of Asian study participants**.

SIMPLE LINEAR REGRESSION

[Using NHANES dataset]

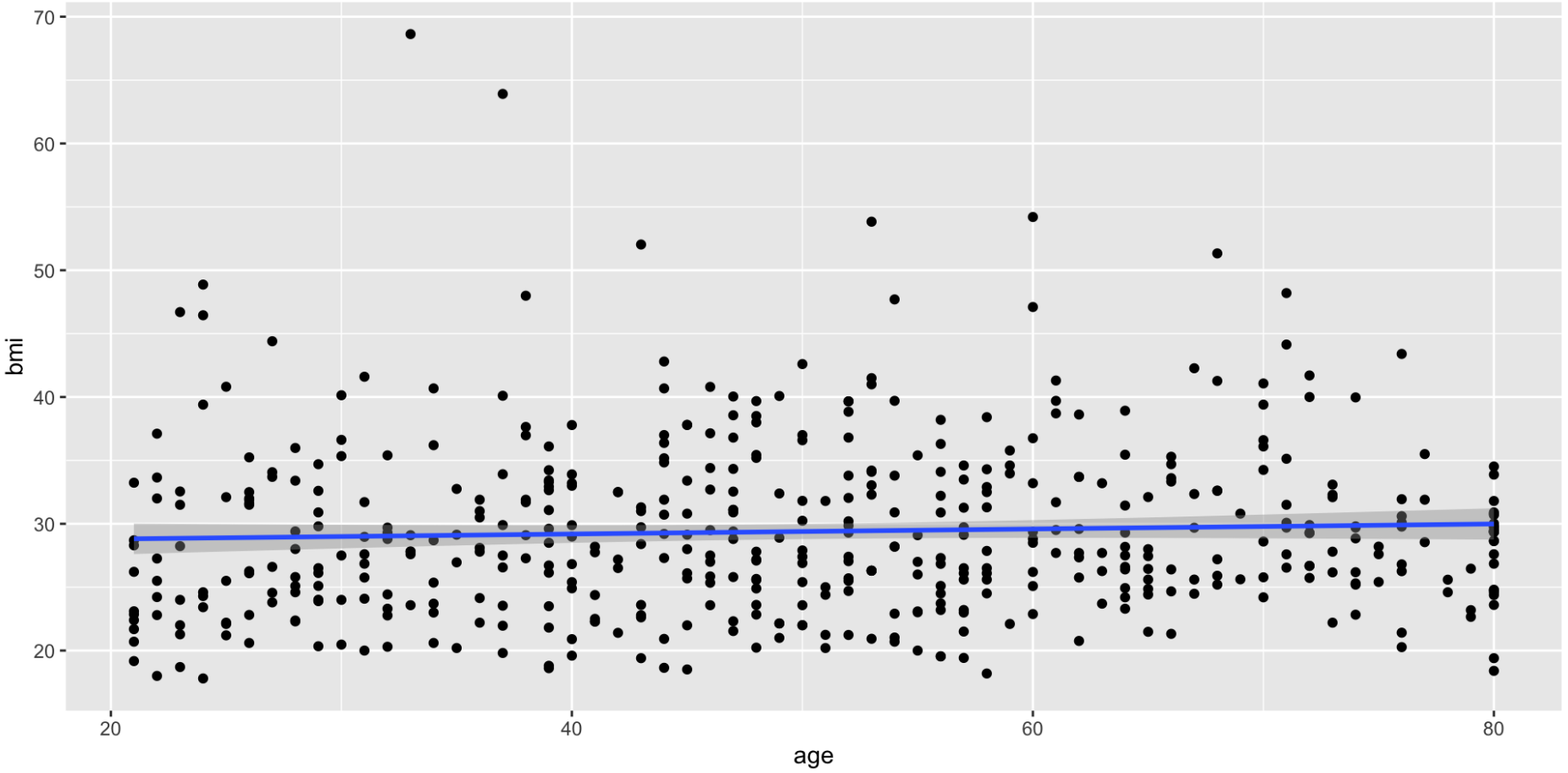
Visualize the data: BMI and age

We are mainly looking for a “vaguely” linear shape here

- `ggplot2` gives us a visual confirmation with `geom_point()`
- Essentially, `geom_smooth()` adds a trend line over an existing plot
  - inside the function, we have different options with the `method` argument (default is LOESS (locally estimated scatterplot smoothing))
  - with `method = lm` we get the linear best fit (the **least squares regression line**) & its 95% CI

```
ggplot(nhanes, aes (x = age,
                            y = bmi)) +
  geom_point() +
  geom_smooth(method = lm,
              #se = FALSE
              )
```

Visualize the data: BMI and age



Linear regression model

The `lm()` function is used to fit linear models has the following generic structure:

```
lm(y ~ x, data)
```

where:

- the 1st argument `y ~ x` specifies the variables used in the model (here the model regresses a **response variable** y against an **explanatory variable** x.
- The 2nd argument `data` is used only when the dataframe name is not already specified in the first argument.

Linear regression models syntax

The following example shows fitting a linear model that predicts **BMI** from **age (in years)** using data from `nhanes` adult sample (individuals 21 years of age or older from the NHANES data).

```r
# fitting linear model
lm(nhanes$bmi ~ nhanes$age)
```

```r
# or equivalently...
lm(bmi ~ age, data = nhanes)
```

```
Call:
lm(formula = bmi ~ age, data = nhanes)

Coefficients:
(Intercept)          age
   28.40113      0.01982
```

- Running the function creates an *object* (of class `lm`) that contains several components (model coefficients, etc), either directly displayed or accessible with `summary()` notation or specific functions.

Linear regression models syntax

We can save the model and then extract individual output elements from it using the `$` syntax

```r
# name the model object
lr_model <- lm(bmi ~ age, data = nhanes)

# extract model output elements
lr_model$coefficients
lr_model$residuals
lr_model$fitted.values
```

The command `summary` returns these elements

- `Call`: reminds the equation used for this regression model
- `Residuals`: a 5 number summary of the distribution of residuals from the regression model
- `Coefficients`:displays the estimated coefficients of the regression model and relative hypothesis testing, given for:
  - intercept
  - explanatory variable(s) slope

Linear regression models interpretation: coefficients

- The model tests the null hypothesis H_0 that a coefficient is 0
- `coefficients` outputs are: `estimate`, `std. error`, `t-statistic`, and `p-value` correspondent to the t-statistic for:
  - *intercept*
  - *explanatory variable(s)* slope
- In regression, the population **parameter of interest** is typically the *slope* parameter
  - in this model, `age` doesn't appear significantly ≠ 0

```
summary(lr_model)$coefficients
```

```
              Estimate Std. Error   t value      Pr(>|t|)
(Intercept) 28.40112932 0.96172389 29.531480 2.851707e-111
age          0.01981675 0.01824641  1.086063  2.779797e-01
```

Linear regression models interpretation: Coefficients 2

For the the estimated coefficients of the regression model, we get:

- `Estimate` = the average increase in the response variable associated with a one unit increase in the predictor variable, (assuming all other predictor variables are held constant).
- `Std. Error` = a measure of the uncertainty in our estimate of the coefficient.
- `t value` = the t-statistic for the predictor variable, calculated as (Estimate) / (Std. Error).
- `Pr(>|t|)` = the p-value that corresponds to the t-statistic. If less than some alpha level (e.g. 0.05). the predictor variable is said to be *statistically significant*.

Linear regression models outputs: fitted values

Here we see $\hat{y_i}$, i.e. the **fitted y value for the i-th individual**

```r
fit_val <- lr_model$fitted.values

# print the first 6 elements
head(fit_val)
```

```
       1        2        3        4        5        6
29.39197 29.33252 29.31270 28.95600 29.39197 29.17398
```

Linear regression models outputs: residuals

Here we see $e_i = y_i - \hat{y_i}$, i.e. the **residual value for the i-th individual**

```r
resid_val <- lr_model$residuals

# print the first 6 elements
head(resid_val)
```

```
          1           2           3           4           5           6
-1.49196704  0.06748322 -3.96270002 -3.15599844 -2.49196704  3.75601726
```

Linear regression model's fit: Residual standard error

- The `Residual standard error` (an estimate of the parameter σ) tells the average distance that the observed values fall from the regression line (we are assuming constant variance).
  - *The smaller it is, the better the model fits the dataset!*

We can compute it manually as:

$$SE_{resid} = \sqrt{\frac{\sum_{i=1}^{n} (y_i - \hat{y_i})^2}{df_{resid}}}$$

```r
# Residual Standard error (Like Standard Deviation)

# ---  inputs
# sample size
n =length(lr_model$residuals)
# n of parameters in the model
k = length(lr_model$coefficients)-1 #Subtract one to ignore intercept
# degrees of freedom of the the residuals
df_resid = n-k-1
# Squared Sum of Errors
SSE =sum(lr_model$residuals^2) # 22991.19

# --- Residual Standard Error
ResStdErr <- sqrt(SSE/df_resid)  # 6.815192
ResStdErr
```

```
[1] 6.815192
```

Linear regression model's fit: : R^2 and $\text{Adj.}\, R^2$

The R^2 tells us the **proportion of the variance in the response variable** that can be explained by the predictor variable(s).

- if R^2 close to 0 -> data more spread
- if R^2 close to 1 -> data more tight around the regression line

```
# --- R^2
summary(lr_model)$r.squared
```

```
[1] 0.00237723
```

The $\text{Adj.}\, R^2$ is a **modified version of** R^2 that has been adjusted for the number of predictors in the model.

- It is always lower than the R-squared
- It can be useful for comparing the fit of different regression models that use different numbers of predictor variables.

```
# --- Adj. R^2
summary(lr_model)$adj.r.squared
```

```
[1] 0.0003618303
```

Linear regression model's fit: : F statistic

The **F-statistic** indicates whether the regression model provides a better fit to the data than a model that contains no independent variables. In essence, it tests if the regression model as a whole is useful.

```
# extract only F statistic
summary(lr_model)$fstatistic
```

```
    value     numdf     dendf 
 1.179533  1.000000 495.000000 
```

```
# define function to extract overall p-value of model
overall_p <- function(my_model) {
    f <- summary(my_model)$fstatistic
    p <- pf(f[1],f[2],f[3],lower.tail=F)
    attributes(p) <- NULL
    return(p)
}

# extract overall p-value of model
overall_p(lr_model)
```

```
[1] 0.2779797
```

Given the **p-value is > 0.05**, this indicate that *the predictor variable is not useful for predicting the value of the response variable*.

DIAGNOSTIC PLOTS

The following plots help us checking if (most of) the assumptions of linear regression are met!

> (the **independence** assumption is more linked to the study design than to the data used in modeling)

Linear regression diagnostic plots: residuals 1/4

> **ASSUMPTION 1**: there exists a linear relationship between the independent variable, x, and the dependent variable, y

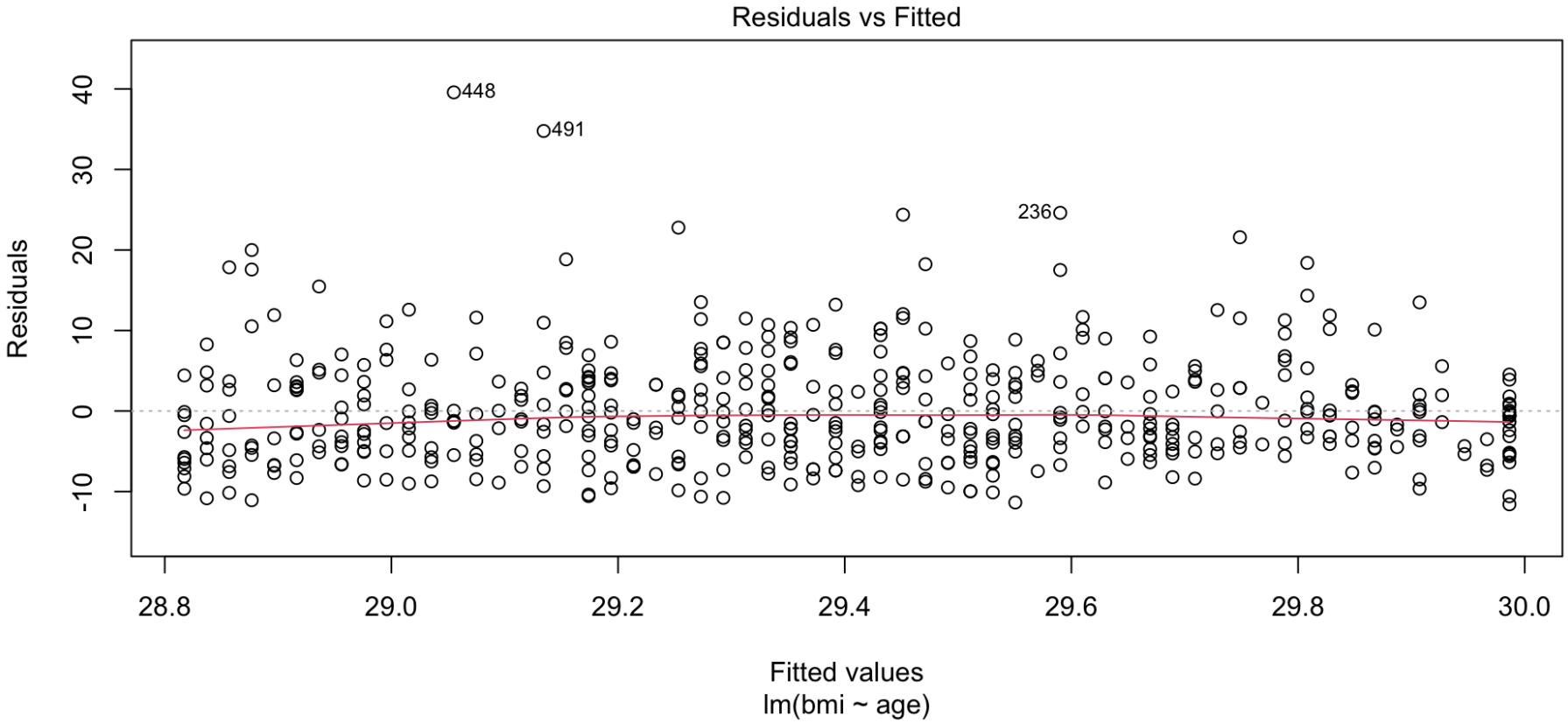
For an observation (x_i, y_i), where $\hat{y_i}$ is the `predicted value` according to the line $\hat{y} = b_0 + b_1 x$, the `residual` is the value $e_i = y_i - \hat{y_i}$

- A linear (e.g. `lr_model`) is a particularly good fit for the data when the residual plot shows random scatter above and below the horizontal line.
  - (In this R plot, we look for a red line that is fairly straight)

```
1 # residual plot
2 plot(lr_model, which = 1 )
```

- We use the argument `which` in the function `plot` so we see the plots one at a time.

Linear regression diagnostic plots: residuals 1/4



Linear regression diagnostic plots: normality of residuals 2/4

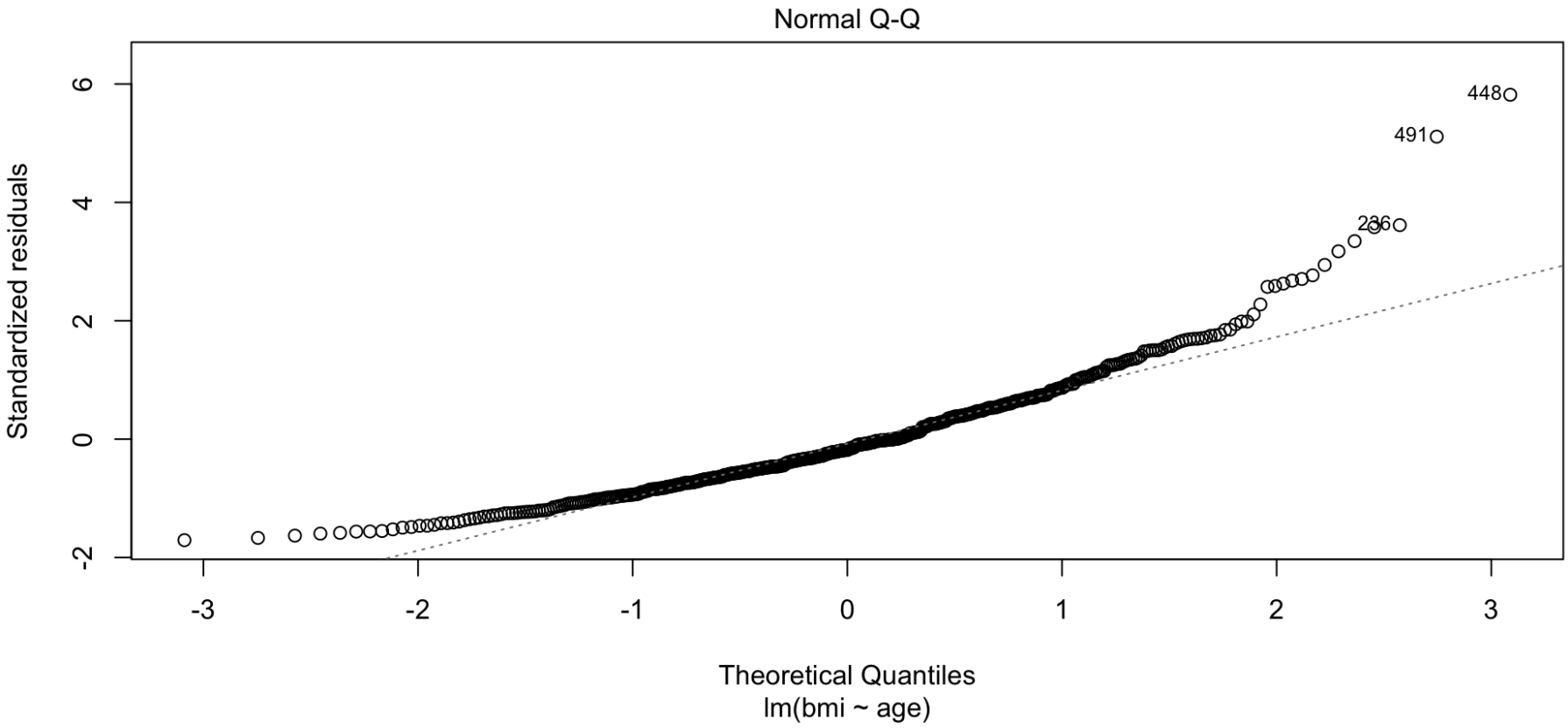
> **ASSUMPTION 2**: The residuals of the model are normally distributed

With the quantile-quantile plot (Q-Q) we can checking normality of the residuals.

```
1 # quantile-quantile plot
2 plot(lr_model, which = 2 )
```

Linear regression diagnostic plots: normality of residuals 2/4

The data appear roughly normal, but there are deviations from normality in the tails, particularly the upper tail.



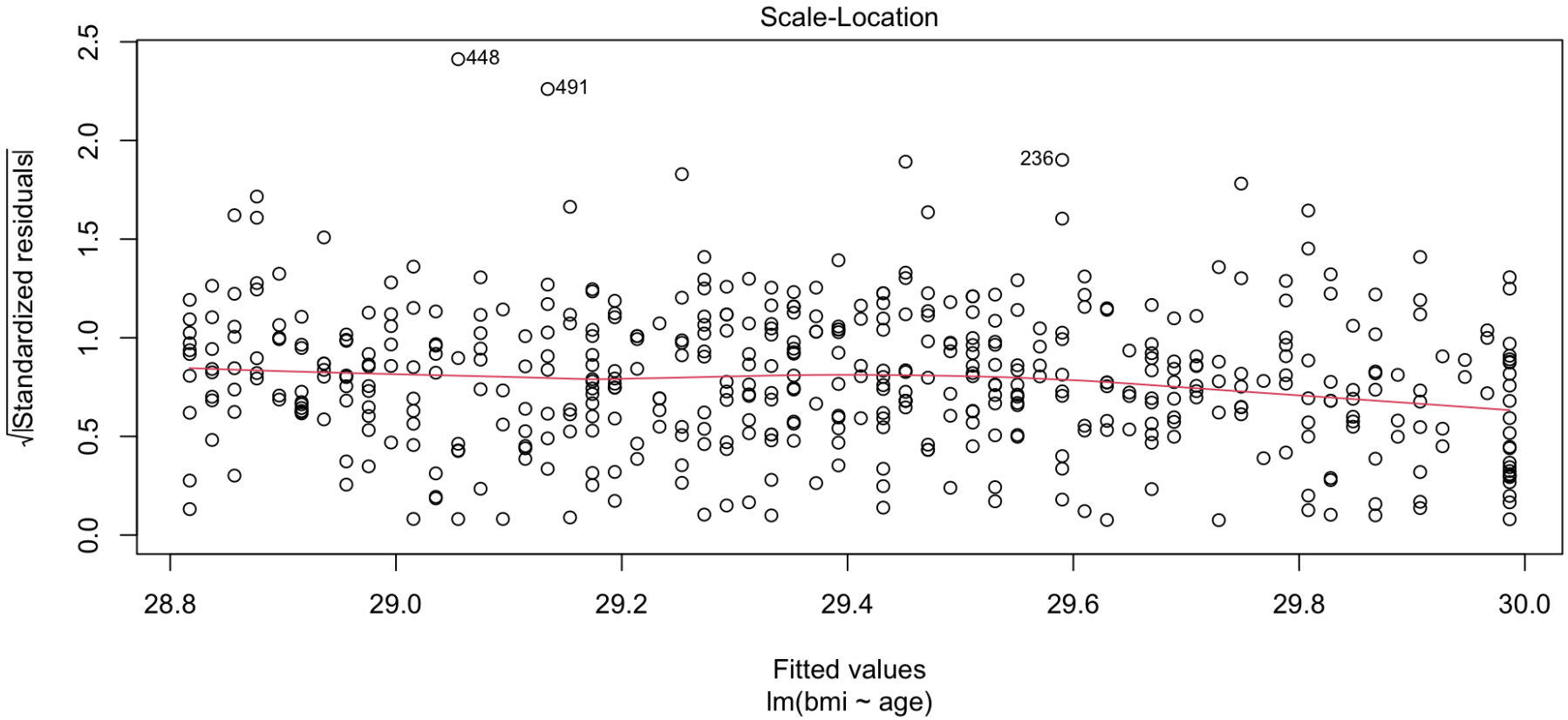
Linear regression diagnostic plots: Homoscedasticity 3/4

> **ASSUMPTION 3**: The residuals have constant variance at every level of x ("*homoscedasticity*")

This one is called a **Spread-location plot**: shows if residuals are spread equally along the ranges of predictors

```
# Spread-location plot
plot(lr_model, which = 3 )
```

Linear regression diagnostic plots: Homoscedasticity 3/4



Test for Homoscedasticity

Besides visual check, we can perform the `Breusch–Pagan test` to verify the assumption of homoscedasticity. In this case:

- H_0 : residuals are distributed with **equal variance**
- H_1 : residuals are distributed with **UNequal variance**
- we use `bptest` function from the `lmtest` package

```
# Breusch-Pagan test against heteroskedasticity
lmtest::bptest(lr_model)
```

```
    studentized Breusch-Pagan test

data:  lr_model
BP = 2.7548, df = 1, p-value = 0.09696
```

Because the test statistic (BP) is small and the p-value is not significant (p-value > 0.05): **WE DO NOT REJECT THE NULL HYPOTHESIS** (i.e. we can assume equal variance)

Linear regression diagnostic plots: leverage 4/4

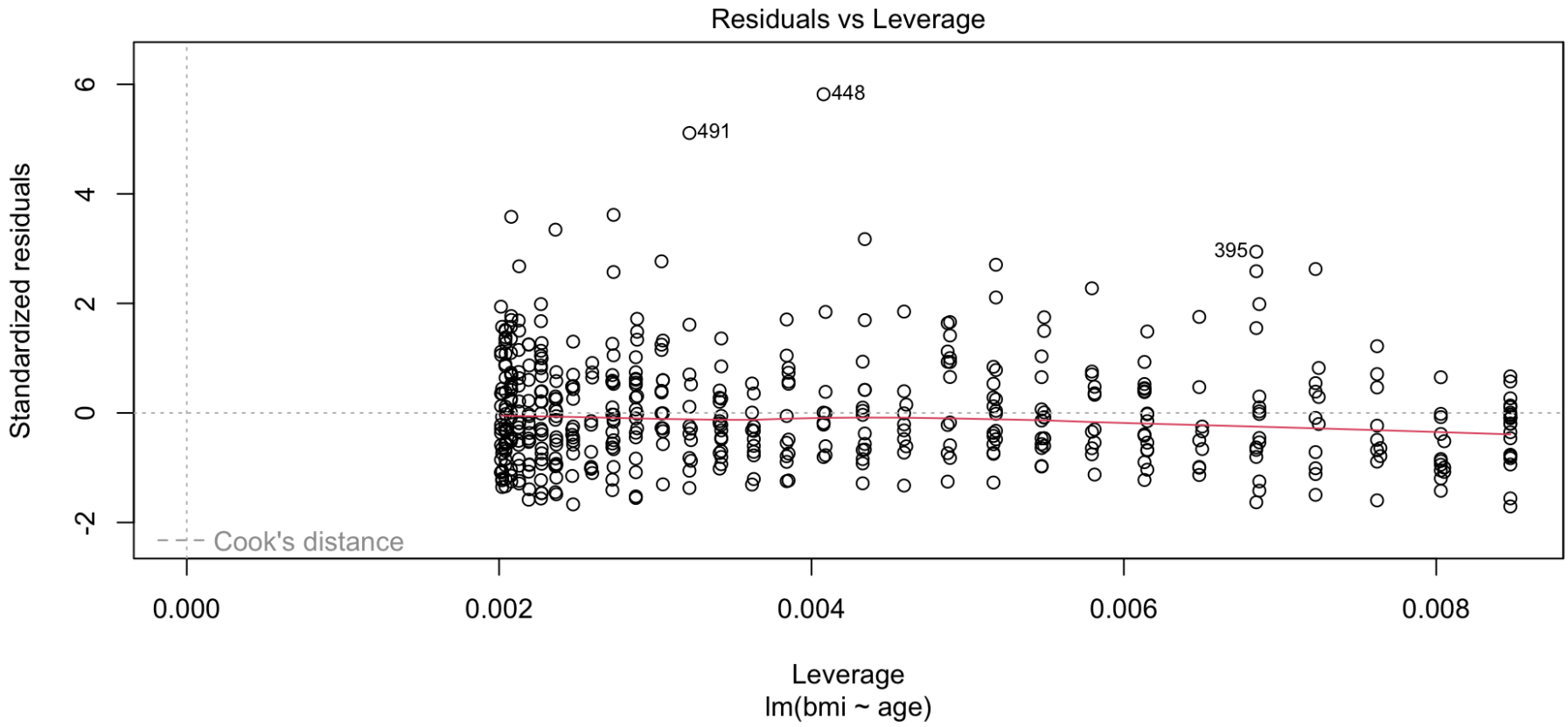
This last diagnostic plot has to do with **outliers**:

- A **residuals vs. leverage plot** allows us to identify *influential observations* in a regression model
  - The x-axis shows the "`leverage`" of each point and the y-axis shows the "`standardized residual of each point`", i.e. *"How much would the coefficients in the regression model would change if a particular observation was removed from the dataset?"*
  - `Cook's distance lines` (red dashed lines) – not visible here – should appear on the corners of the plot when there are influential cases

```
1 plot(lr_model, which = 5 )
```

Linear regression diagnostic plots: leverage 4/4

In this particular case, there is no influential case, or cases



(Digression on the `broom` package)

- The `broom` package introduces the ***tidy approach*** to regression modeling code and outputs, allowing to convert/save them in the form of `tibbles`
- The function `tidy` will turn an object into a tidy tibble
- The function `glance` will construct a single row summary "glance" of a model, fit, or other object
- The function `augment` will show a lot of results for the model attached to each observation
  - this is very useful for further use of such objects, like `ggplot2` etc.

```
# render model as a dataframe
broom::tidy(lr_model)

# see overal performance
broom::glance(lr_model)

# save an object with all the model output elements
model_aug <- broom::augment(lr_model)
```

You try…

Run these functions and then run `View(model_aug)` to check out the output

MULTIPLE LINEAR REGRESSION

[Using PREVEND dataset: a sample of 500 obs]

Visualize the data: Statin use and cognitive function

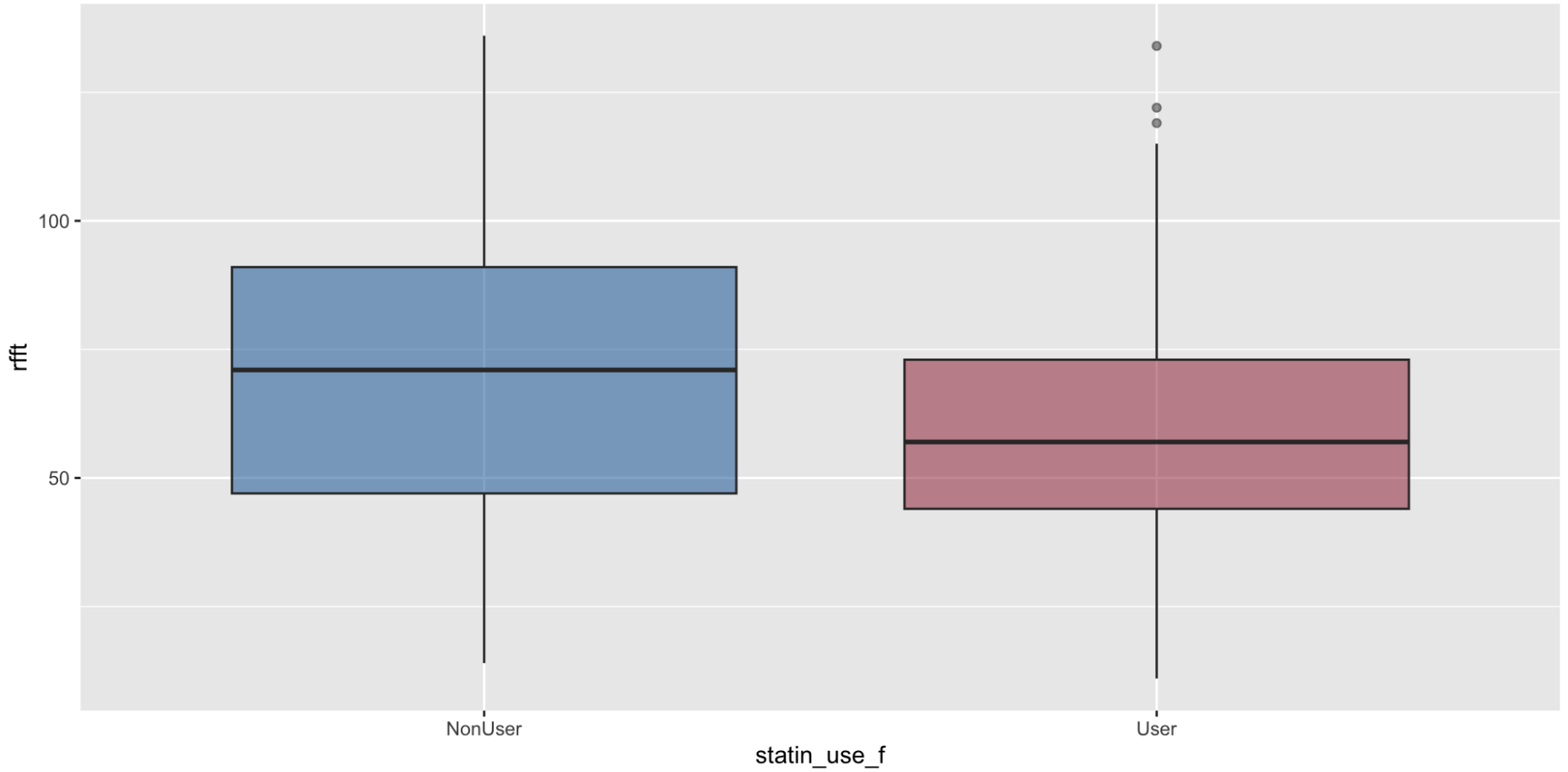
Statins are a class of drugs widely used to lower **cholesterol** (recent guidelines would lead to statin use in almost half of Americans between 40 - 75 years of age and nearly all men over 60). But a few small studies have suggested that statins may be associated with lower **cognitive ability**.

- From this sample of the PREVEND study, we can observe the relationship between **statin use** (`statin_use`) and **cognitive ability** (`rfft`).

```
# rename for convenience
prevend <- prevend_samp %>% janitor::clean_names() %>%
  #create statin.use logical + factor
  mutate(statin_use = as.logical(statin)) %>%
  mutate(statin_use_f = factor(statin, levels = c(0,1), labels = c("NonUser", "User")))

# box plot
ggplot(prevend,
       aes (x = statin_use_f, y = rfft, fill = statin_use_f)) +
  geom_boxplot(alpha=0.5) +
  scale_fill_manual(values=c("#005ca1","#9b2339" )) +
  # drop legend and Y-axis title
  theme(legend.position = "none")
```

Visualize the data: Statin use and cognitive function

The boxplot suggests that statin user (red) present lower cognitive ability score, on average



Confirm visual intuition with independent sample t-test

We could use an independent t-test to confirm what the boxplot shows

```r
t_test_w <- t.test(prevend$rfft[prevend$statin == 1],
                   prevend$rfft[prevend$statin == 0],
                   # here we specify the situation
                   var.equal = TRUE,
                   paired = FALSE, alternative = "two.sided")

t_test_w
```

```
    Two Sample t-test

data:  prevend$rfft[prevend$statin == 1] and prevend$rfft[prevend$statin == 0]
t = -3.4917, df = 498, p-value = 0.0005226
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -15.710276  -4.396556
sample estimates:
mean of x mean of y
 60.66087  70.71429
```

…statistically significant difference in means (Statin use: yes and no) do exist

Consider Simple Linear regression: Statin use and cognitive function

… and build a simple linear regression model like so:

$$E(RFFT) = b_0 + b_{statin}(\text{Statin use})$$

```r
#fit the linear model
model_1 <- lm(rfft ~ statin, data=prevend)
summary(model_1)
```

```
Call:
lm(formula = rfft ~ statin, data = prevend)

Residuals:
    Min      1Q  Median      3Q     Max
-56.714 -22.714   0.286  18.299  73.339

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   70.714      1.381  51.212  < 2e-16 ***
statin       -10.053      2.879  -3.492 0.000523 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 27.09 on 498 degrees of freedom
Multiple R-squared:  0.0239,    Adjusted R-squared:  0.02194
F-statistic: 12.19 on 1 and 498 DF,  p-value: 0.0005226
```

- This preliminary model shows that, on average, **statin** users score approximately 10 points lower on the RFFT cognitive test (and the statin coefficient is **highly significant**!)

Visualize the data: Statin use and cognitive function + age

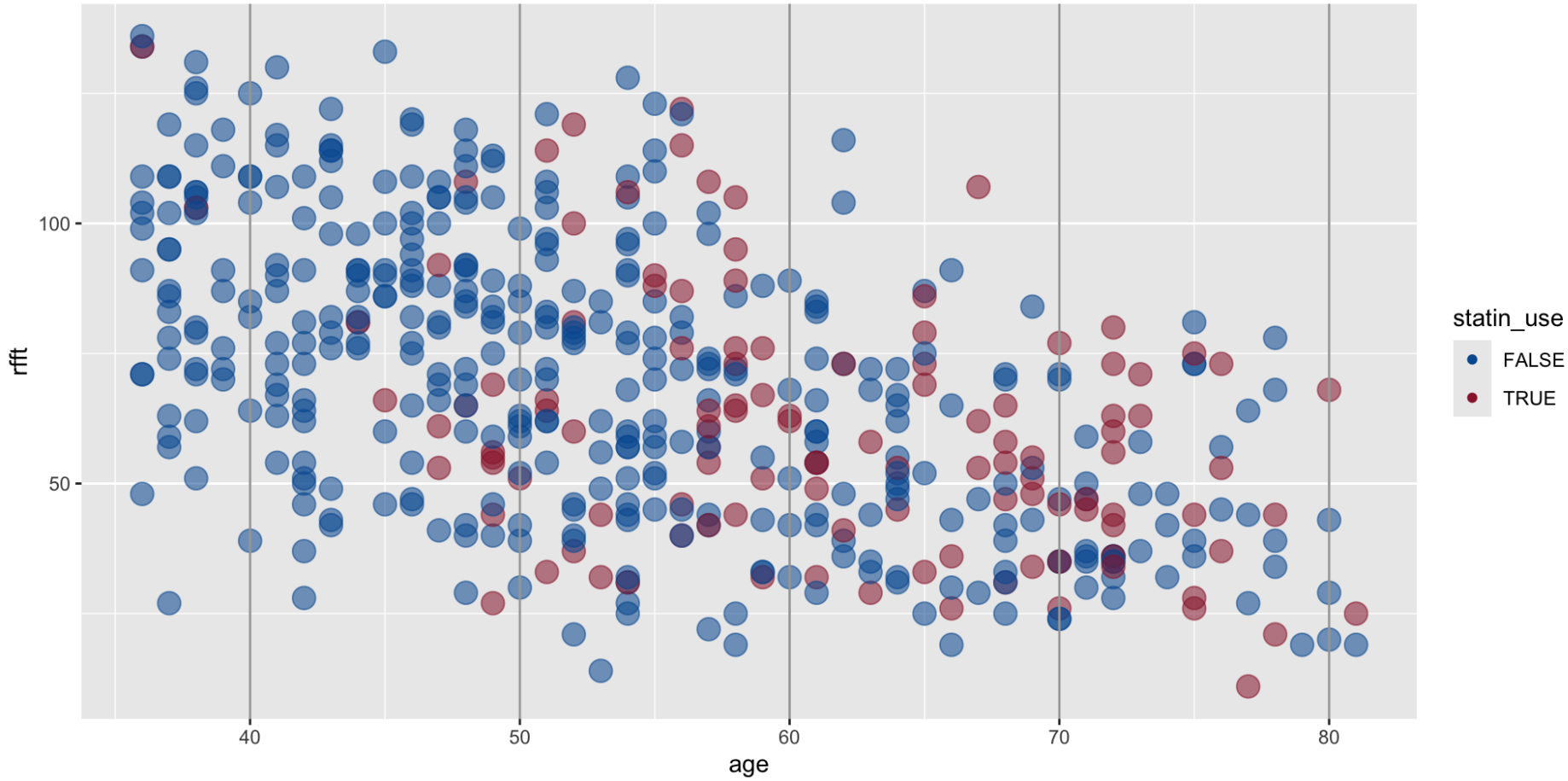
However, following the literature, this prelimary model might be misleading (`biased`) because it does not account for the underlying relationship between age and statin

- hence **age** could be a `confounder` within the **statin** -> **RFFT** relationship

```
ggplot(prevend,
       aes (x = age, y = rfft, group = statin_use)) +
  geom_point (aes(color = statin_use , size=.01, alpha = 0.75),
              show.legend = c(size = F, alpha = F) )+
  scale_color_manual(values=c("#005ca1","#9b2339" )) +
  # decades line separators
  geom_vline(xintercept = 40, color = "#A6A6A6")+
  geom_vline(xintercept = 50, color = "#A6A6A6")+
  geom_vline(xintercept = 60, color = "#A6A6A6")+
  geom_vline(xintercept = 70, color = "#A6A6A6")+
  geom_vline(xintercept = 80, color = "#A6A6A6")
```

Visualize the data: Statin use and cognitive function + age

Statin users are represented with red points; participants not using statins are shown as blue points



Multiple linear regression model

Multiple regression allows for a (richer) model that incorporates both statin use and age:

$$\mathrm{E(RFFT)} = \mathrm{b_0} + \mathrm{b_{statin}}\,(\text{Statin use}) + \mathrm{b_{age}}\,(\text{Age})$$

- or (*in statistical terms*) the association between **RFFT** and **Statin use** is being estimated `after adjusting` for **Age**

The R syntax is very easy: simply use + to add covariates

```
# fit the (multiple) linear model
model_2 <- lm(rfft ~ statin + age , data=prevend)
```

RFFT vs. statin use & age…

Although the use of statins appeared to be associated with lower RFFT scores when no adjustment was made for possible confounders, **`statin use` is not significantly associated with `RFFT score` in a regression model that adjusts for `age`.**

```
1 summary(model_2)
```

```
Call:
lm(formula = rfft ~ statin + age, data = prevend)

Residuals:
    Min      1Q  Median      3Q     Max
-63.855 -16.860  -1.178  15.730  58.751

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 137.8822     5.1221  26.919   <2e-16 ***
statin        0.8509     2.5957   0.328    0.743
age          -1.2710     0.0943 -13.478   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 23.21 on 497 degrees of freedom
Multiple R-squared:  0.2852,    Adjusted R-squared:  0.2823
F-statistic: 99.13 on 2 and 497 DF,  p-value: < 2.2e-16
```

Evaluating a multiple regression model

Assumptions for multiple regression

Similar to those of simple linear regression…

1. **Linearity**: For each predictor variable x_j, change in the predictor is linearly related to change in the response variable when the value of all other predictors is held constant.
2. **Constant variability**: The residuals have approximately constant variance.
3. **Normality of residuals**: The residuals are approximately normally distributed.
4. **Independent observations**: Each set of observations $(y, x_1, x_2, \ldots, x_p)$ is independent.
5. **No multicollinearity**: i.e. no situations when there is a strong linear correlation between the independent variables, conditional on the other variables in the model

Using residual plots to assess LINEARITY: age

> **ASSUMPTION 1**: there exists a linear relationship between the independent variables, $(x_1, x_2, \ldots, x_p)$, and the dependent variable, y

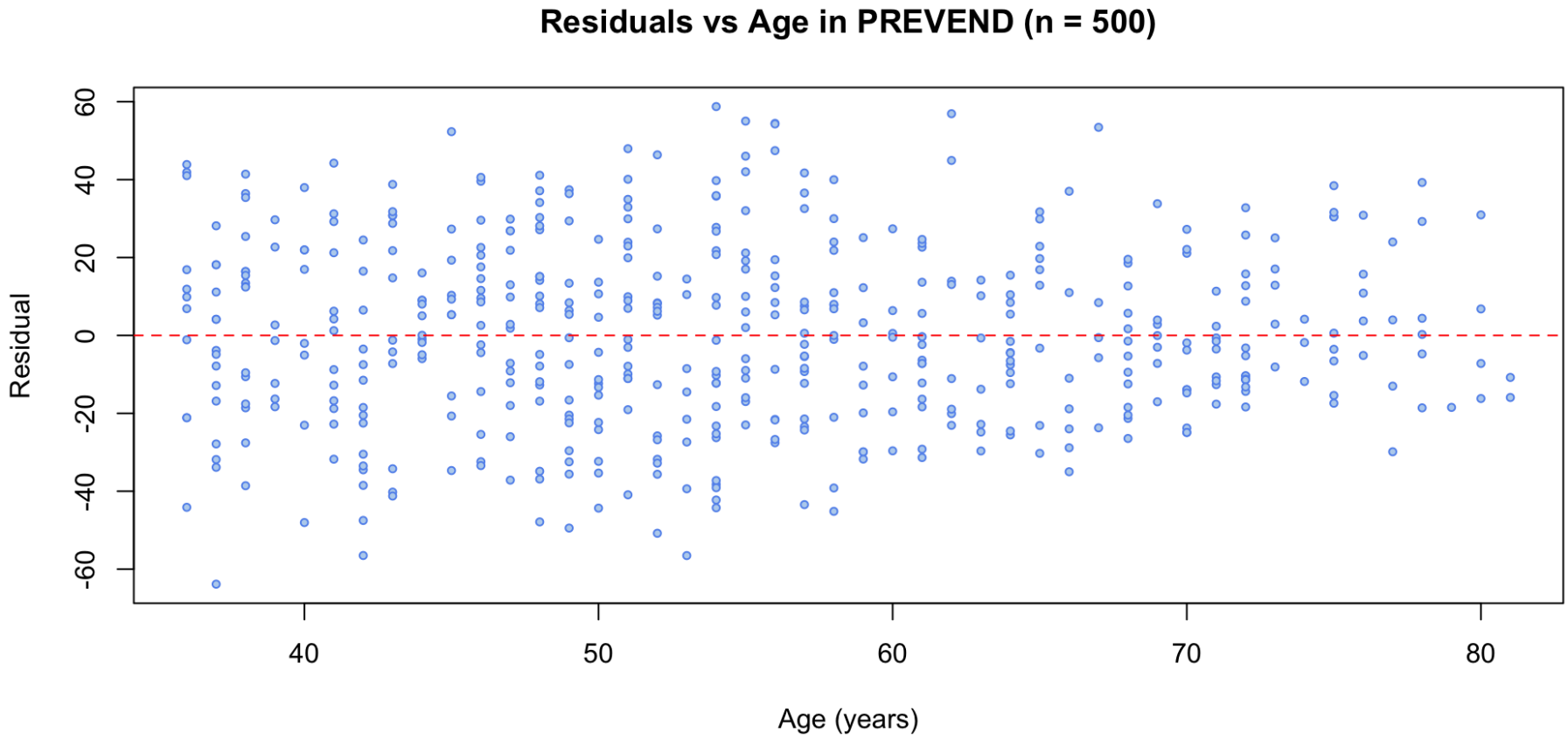
It is not possible to make a scatterplot of a response against several simultaneous predictors. Instead, use a `modified residual plot` to assess linearity:

- For **each** (numerical) predictor, plot the residuals on the y-axis and the predictor values on the x-axis.
- Patterns/curvature are indicative of non-linearity.

```r
# recall
model_2 <- lm(rfft ~ statin + age , data=prevend)

# assess linearity
plot(residuals(model_2) ~ prevend$age,
     main = "Residuals vs Age in PREVEND (n = 500)",
     xlab = "Age (years)", ylab = "Residual",
     pch = 21, col = "cornflowerblue", bg = "slategray2",
     cex = 0.60)
abline(h = 0, col = "red", lty = 2)
```

Using residual plots to assess LINEARITY: age

There are no apparent trends; the data scatter evenly above and below the horizontal line. There does not seem to be remaining nonlinearity with respect to age after the model is fit.



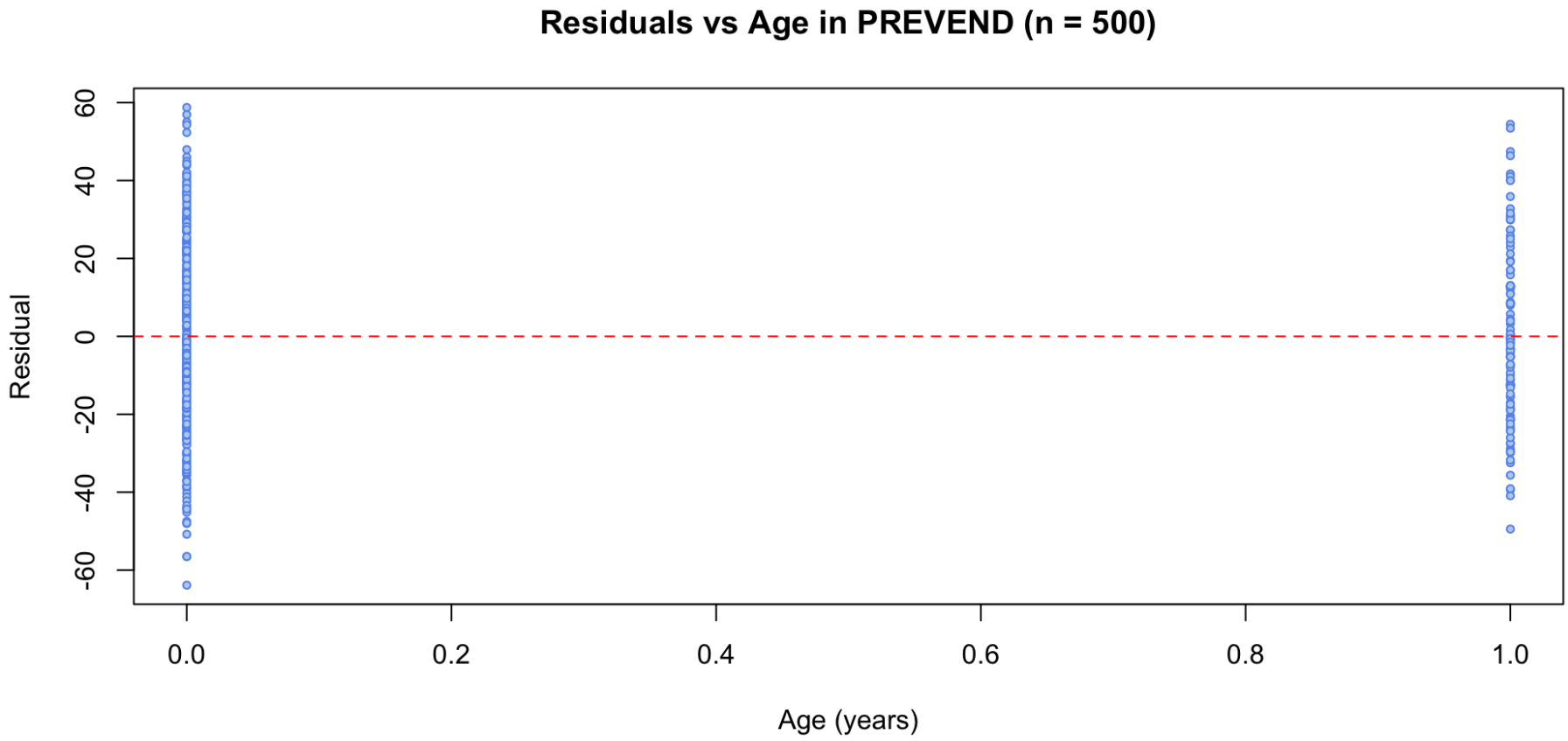
Using residual plots to assess LINEARITY: statin use

Should we be testing linearity of residuals also against a **categorical variable** (`statin use`)? (not really, because not meaningful)

```r
# recall
model_2 <- lm(rfft ~ statin + age , data=prevend)

#assess linearity
plot(residuals(model_2) ~ prevend$statin,
     main = "Residuals vs Age in PREVEND (n = 500)",
     xlab = "Age (years)", ylab = "Residual",
     pch = 21, col = "cornflowerblue", bg = "slategray2",
     cex = 0.60)
abline(h = 0, col = "red", lty = 2)
```

Using residual plots to assess LINEARITY: statin use

It is not necessary to assess linearity with respect to statin use since statin use is measured as a categorical variable. A line drawn through two points (that is, the mean of the two groups defined by a binary variable) is necessarily linear



Using residual plots to assess CONSTANT VARIABILITY

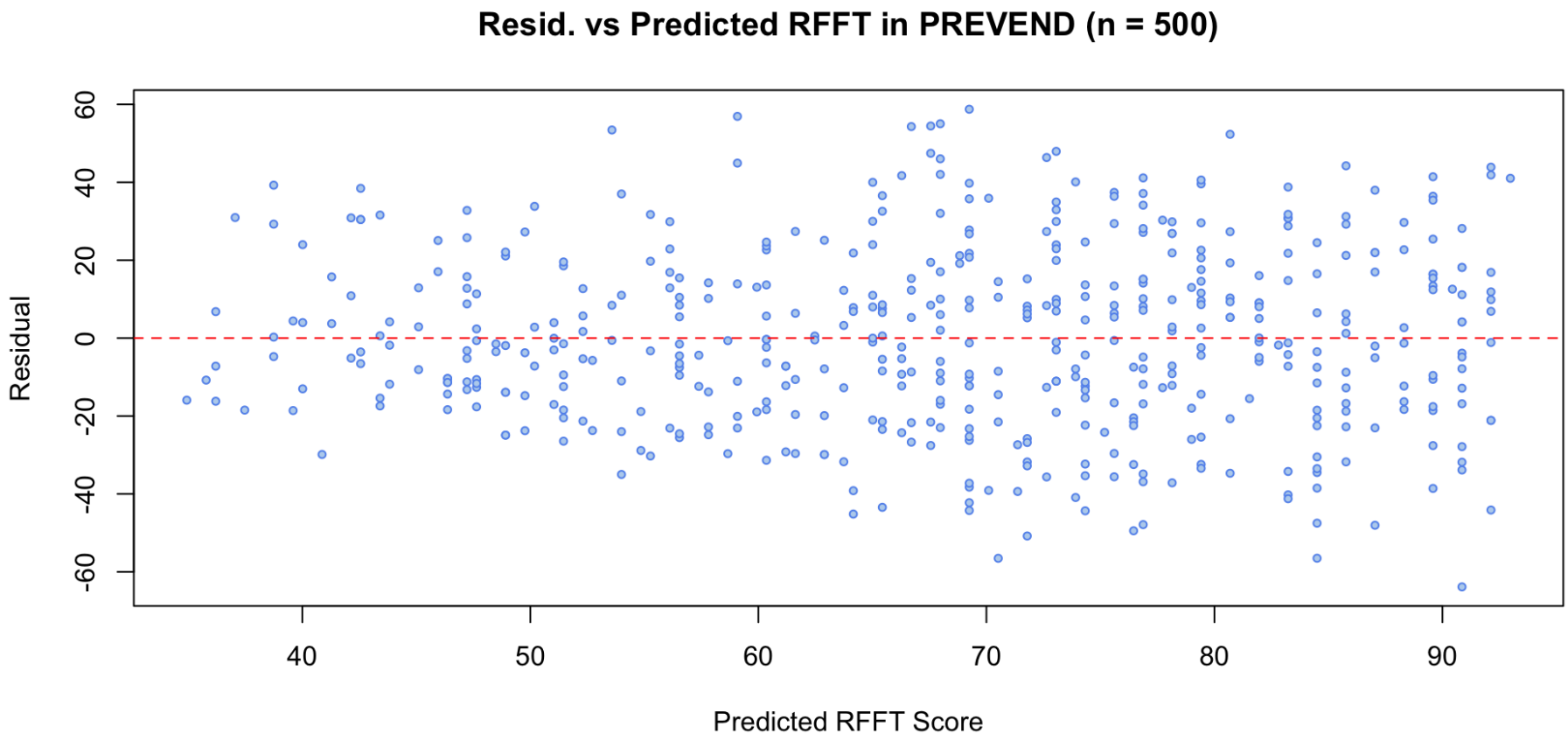
> **ASSUMPTION 2**: The residuals have constant variance at every level of x ("*homoscedasticity*")

- Constant variability: plot the residual values on the y-axis and the predicted values on the x-axis

```
#assess constant variance of residuals
plot(residuals(model_2) ~ fitted(model_2),
     main = "Resid. vs Predicted RFFT in PREVEND (n = 500)",
     xlab = "Predicted RFFT Score", ylab = "Residual",
     pch = 21, col = "cornflowerblue", bg = "slategray2",
     cex = 0.60)
abline(h = 0, col = "red", lty = 2)
```

Using residual plots to assess CONSTANT VARIABILITY

The variance of the residuals is somewhat smaller for lower predicted values of RFFT score, but this may simply be an artifact from observing few individuals with relatively low predicted scores. It seems reasonable to assume approximately constant variance.



Using residual plots to assess NORMALITY of residuals

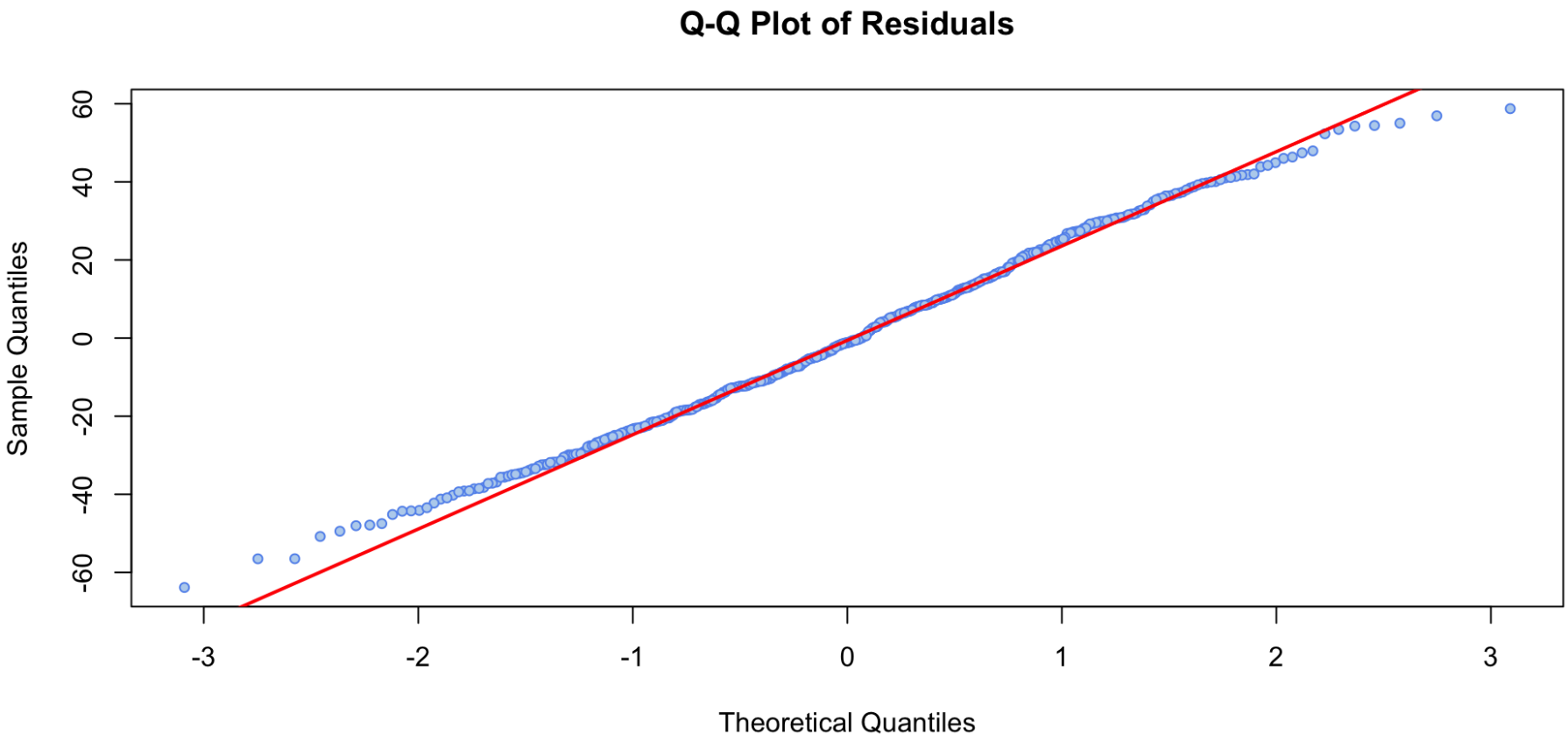
> **ASSUMPTION 3**: The residuals of the model are normally distributed - Normality of residuals: use Q-Q plots

```
#assess normality of residuals
qqnorm(resid(model_2),
       pch = 21, col = "cornflowerblue", bg = "slategray2", cex = 0.75,
       main = "Q-Q Plot of Residuals")
qqline(resid(model_2), col = "red", lwd = 2)
```

In our example, we see that most data points are OK, except some observations at the tails. However, if all other plots indicate no violation of assumptions, some deviation of normality, particularly at the tails, can be less critical.

Using residual plots to assess NORMALITY of residuals

The residuals are reasonably normally distributed, with only slight departures from normality in the tails.



Assumption of INDEPENDENCE of observations

> **ASSUMPTION 4**: Each set of observations $(y, x_1, x_2, \ldots, x_p)$ is independent.

Is it reasonable to assume that each set of observations is independent of the others?

> Using the PREVEND data, it is reasonable to assume that the observations in this dataset are independent. The participants were recruited from a large city in the Netherlands for a study focusing on factors associated with renal and cardiovascular disease.

Assumption of NO MULTICOLLINEARITY

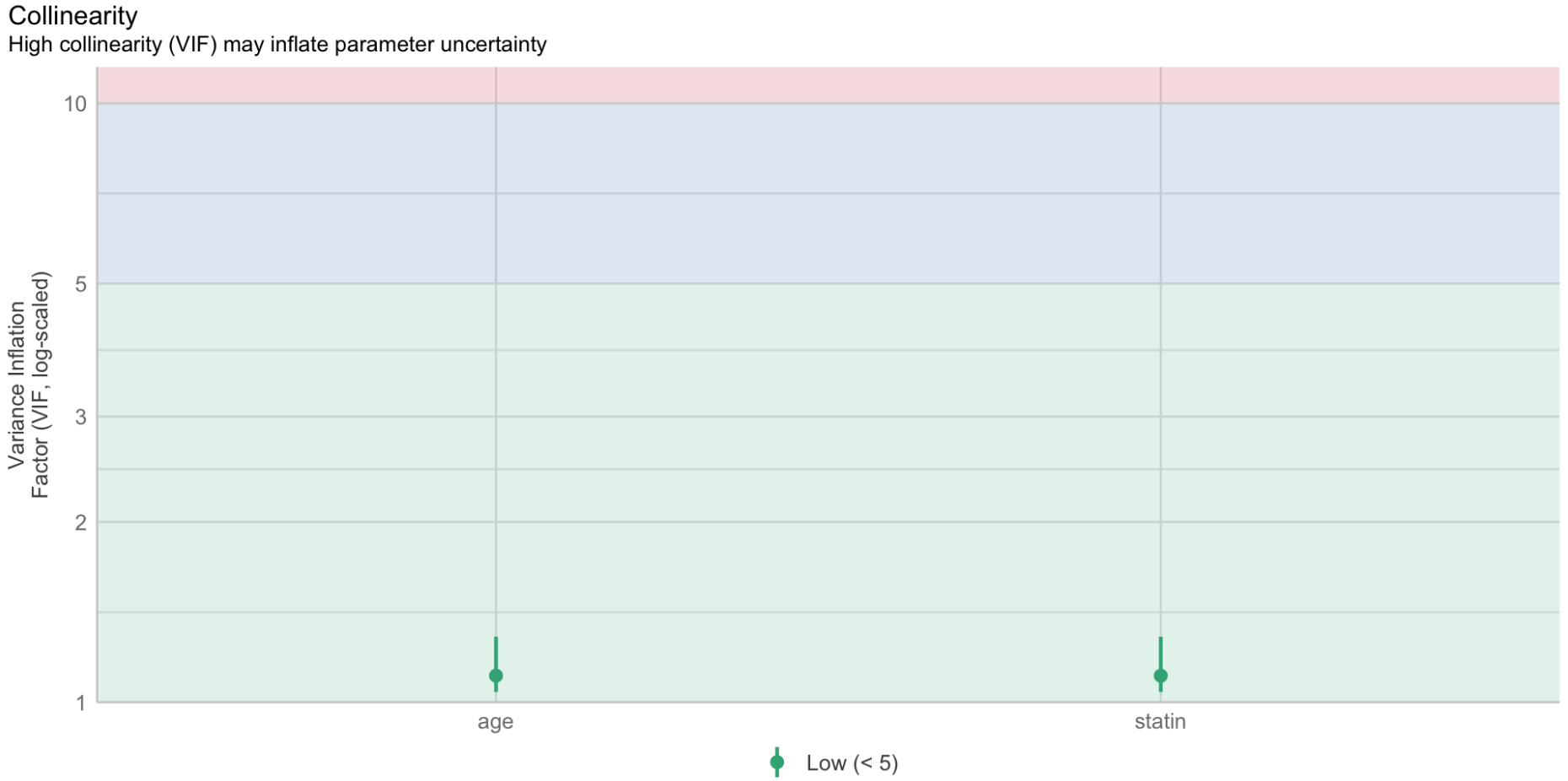
> **ASSUMPTION 5**: Each set of observations $(y, x_1, x_2, \ldots, x_p)$ is independent.

The R package `performance` actually provides a very helpful function `check_model()` which tests these assumptions all at the same time

- **Multicollinearity** is not an issue (based on a general threshold of 10 for VIF, all of them are below 10)

```
# return and store a list of single plots
diagnostic_plots <- plot(performance::check_model(model_2, panel = FALSE))

# see multicollinearity plot
diagnostic_plots[[5]]
```

Assumption of NO MULTICOLLINEARITY



Checking out the `performance` R package

- Find more info on the helpful `performance` R package here for verifying assumptions and model's quality and goodness of fit.

You try…

Run also the following commands

- Diagnostic plot of linearity `diagnostic_plots[[2]]`
- Diagnostic plot of influential observations - outliers `diagnostic_plots[[4]]`
- Diagnostic plot of normally distributed residuals `diagnostic_plots[[6]]`

R^2 with multiple regression

As in simple regression, R^2 represents the proportion of variability in the response variable explained by the model.

- As variables are added, R^2 always increases.

In the `summary(lm( ))` output, `Multiple R-squared` is R^2.

```
#extract R^2 of a model
summary(model_2)$r.squared
```

```
[1] 0.2851629
```

> The R^2 is 0.285; **the model explains 28.5% of the observed variation in RFFT score**. The moderately low R^2 suggests that the model is missing other predictors of RFFT score.

Adjusted R^2 as a tool for model assessment

The **adjusted** R^2 is computed as:

$$R^2_{adj} = 1 - \left(\frac{Var(e_i)}{Var(y_i)} \times \frac{n-1}{n-p-1} \right)$$

- where n is the number of cases and p is the number of predictor variables.

Adjusted R^2 incorporates a penalty for including predictors that do not contribute much towards explaining observed variation in the response variable.

- It is often used to balance predictive ability with model complexity.
- Unlike R^2, R^2_{adj} does not have an inherent interpretation.

```
#extract adjusted R^2 of a model
summary(model_2)$adj.r.squared
```

```
[1] 0.2822863
```

INTRODUCING SPECIAL KINDS OF PREDICTORS

Categorical predictor in regression - (example)

Is RFFT score associated with **education**? The variable `Education` in the `PREVEND` dataset indicates the highest level of education an individual completed in the Dutch educational system:

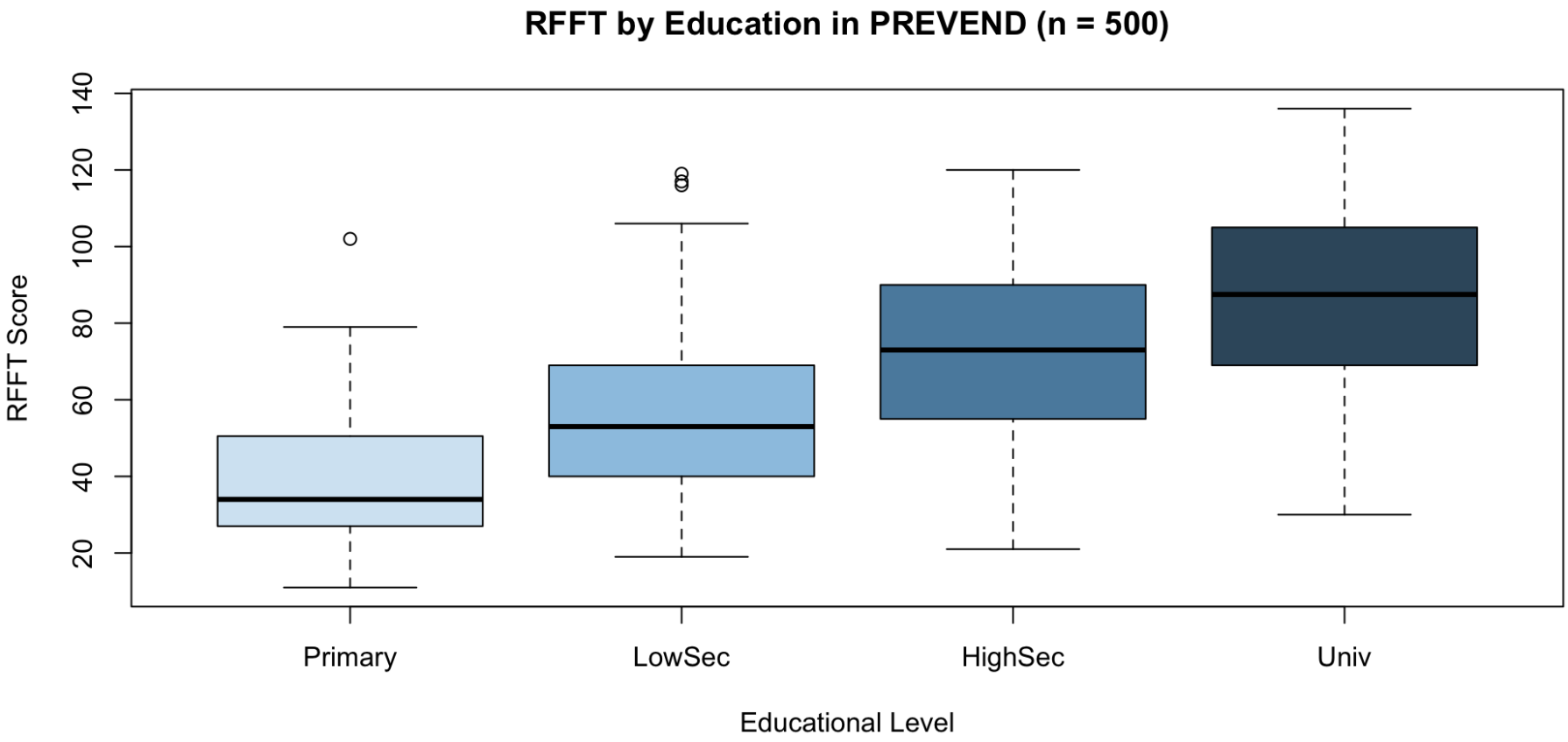
- 0: primary school
- 1: lower secondary school
- 2: higher secondary education
- 3: university education

```r
# convert Education to a factor
prevend <- prevend %>%
  mutate(educ_f = factor(education,
                         levels = c(0, 1, 2, 3),
                         labels = c("Primary", "LowerSecond",
                                    "HigherSecond", "Univ")))
```

```r
# load 4 color palette
pacificharbour_shades <- c( "#d4e6f3",  "#9cc6e3", "#5b8bac", "#39576c",  "#16222b")

# create plot
plot(rfft ~ educ_f, data = prevend,
     xlab = "Educational Level", ylab = "RFFT Score",
     main = "RFFT by Education in PREVEND (n = 500)",
     names = c("Primary", "LowSec", "HighSec", "Univ"),
     col = pacificharbour_shades[1:4])
```

Categorical predictor in regression - (example)

A very clear association seems to exist between education level and average RFFT score in the sample



Categorical predictor in regression - model

Calculate the average RFFT score in the sample across education levels

```
# calculate group means
prevend %>%
  group_by(educ_f) %>%
  summarise(avg_RFFT_score = mean(rfft))
```

```
# A tibble: 4 × 2
  educ_f       avg_RFFT_score
  <fct>                 <dbl>
1 Primary                40.9
2 LowerSecond            55.7
3 HigherSecond           73.1
4 Univ                   85.9
```

Fitting a model with `education` as a predictor

```
# fit a model
model_cat <- lm(rfft ~ educ_f, data = prevend)
model_cat$coefficients
```

```
      (Intercept)  educ_fLowerSecond educ_fHigherSecond         educ_fUniv
         40.94118           14.77857           32.13345           44.96389
```

- Notice how `Primary` level of `educ_f` does NOT appear as a coefficient

Categorical predictor in regression - model interpretation

```
Call:
lm(formula = rfft ~ educ_f, data = prevend)

Residuals:
    Min      1Q  Median      3Q     Max
-55.905 -15.975  -0.905  16.068  63.280

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)         40.941      3.203  12.783  < 2e-16 ***
educ_fLowerSecond   14.779      3.686   4.009 7.04e-05 ***
educ_fHigherSecond  32.133      3.763   8.539  < 2e-16 ***
educ_fUniv          44.964      3.684  12.207  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 22.87 on 496 degrees of freedom
Multiple R-squared:  0.3072,    Adjusted R-squared:  0.303
F-statistic:  73.3 on 3 and 496 DF,  p-value: < 2.2e-16
```

The baseline category represents individuals who at most completed primary school `Education = 0`. The coefficients represent the change in estimated average RFFT relative to the baseline category.

- `(Intercept)` is the sample mean RFFT score for these individuals, 40.94 points
- An increase of 14.78 points is predicted for `LowerSecond` level, $40.94 + 14.78 = 55.72$ points
- An increase of 32.13 points is predicted for `HigherSecond` level, $40.94 + 32.13 = 73.07$ points
- An increase of 44.96 points is predicted for `Univ` level, $40.94 + 44.96 = 85.90$ points

Interaction in regression - (example) - NHANES

Let's go back to the NHANES dataset and consider a linear model that predicts **total cholesterol level (mmol/L)** from **age (yrs.)** and **diabetes status**.

The multiple regression model:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p + \epsilon$$

assumes that when one of the predictors x_j is changed by 1 unit and the values of the other variables remain constant, the predicted response changes by β_j, *regardless of the values of the other variables*.

- With statistical **interaction**, this assumption is not true, such that *the effect of one explanatory variable* x_j *on the* y *depends on the particular value(s) of one or more other explanatory variables*.

Interaction in regression - visual

Fitting a model with `age` and `diabetes` as independent predictors (i.e. WITHOUT interaction terms)

```r
# fit a model
model_NOinterac <- lm(tot_chol ~ age + diabetes, data = nhanes)
model_NOinterac$coefficients
```

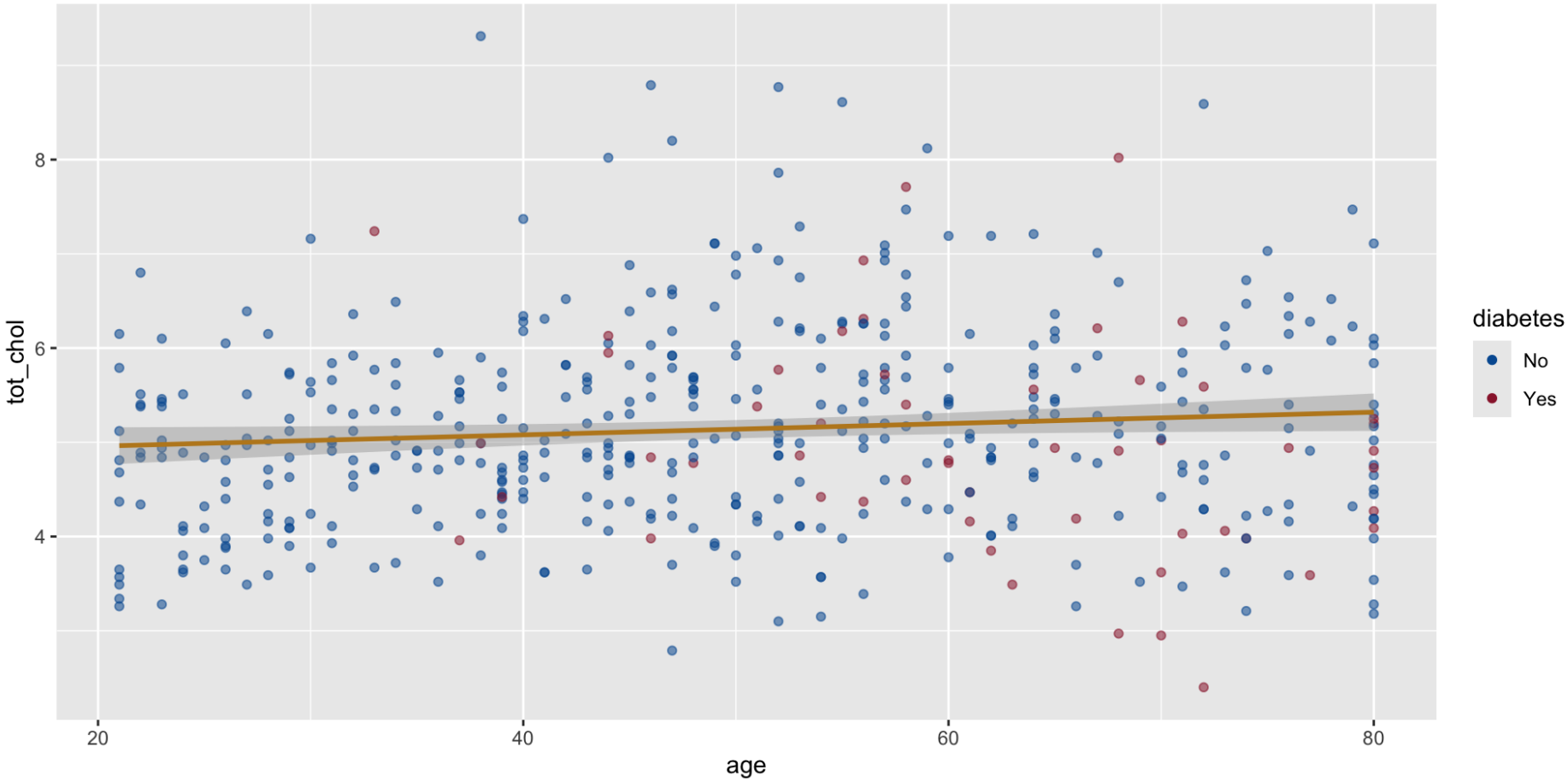
```
(Intercept)          age  diabetesYes
4.800011340  0.007491805 -0.317665963
```

- Using `geom_smooth` for a visual intuition of a linear relationship
  - ⚠️ here I consider sample DATA **as a whole** for plotting a smooth line

```r
ggplot(nhanes,
       aes (x = age, y = tot_chol)) +
  # For POINTS I split by category (category)
  geom_point (aes(color = diabetes,
                  alpha = 0.75),
              show.legend = c(size = F, alpha = F) )+
  scale_color_manual(values=c("#005ca1","#9b2339" )) +
  # For SMOOTHED LINES I take ALL data
  geom_smooth(colour="#BD8723",  method = lm)
```

Interaction in regression - visual

Users in two categories are represented points; linear relationship is representated by ONE golden line for ALL SAMPLE



Interaction in regression - visual (RETHINKING)

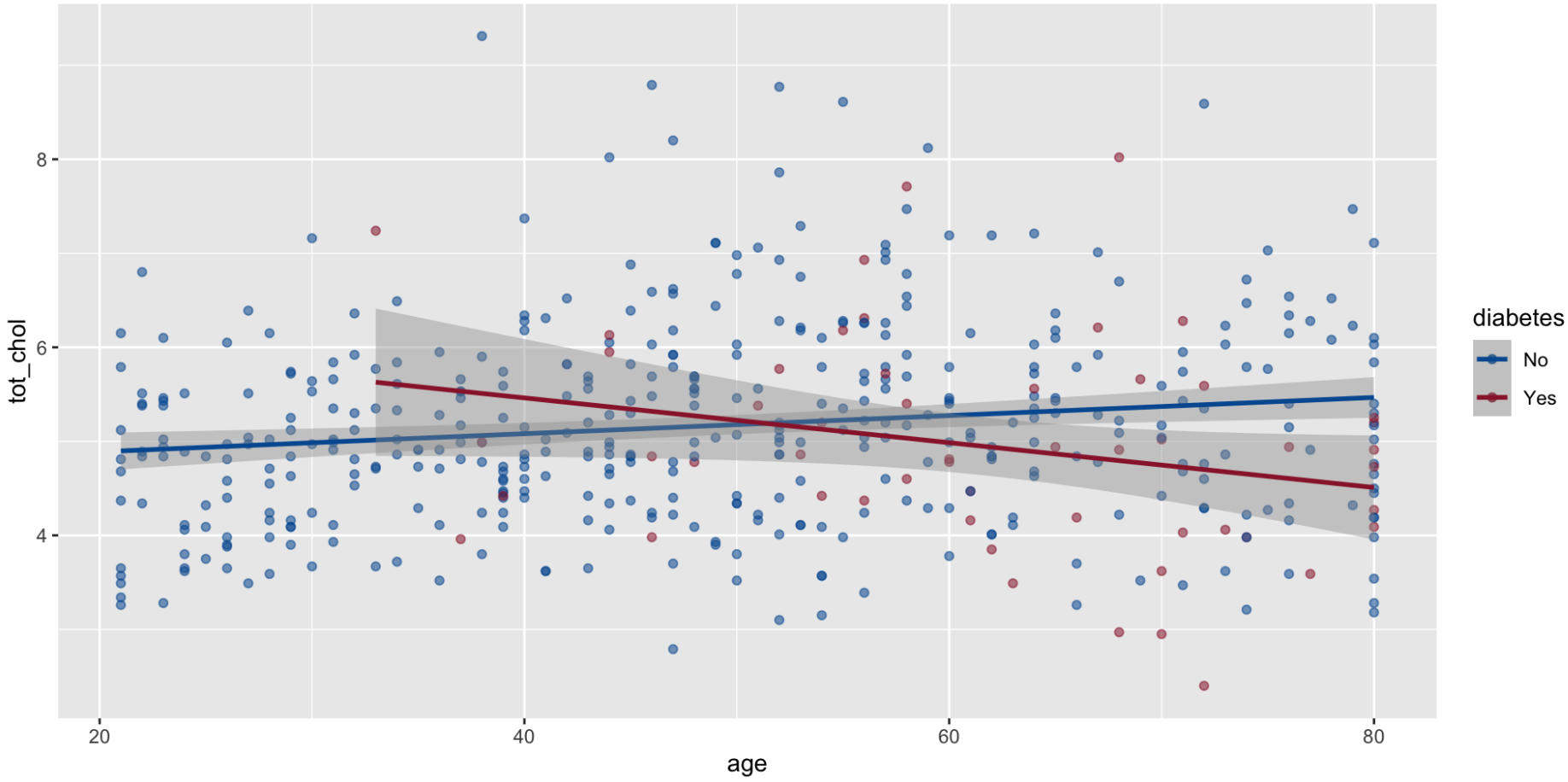
Suppose two separate models were fit for the relationship between total cholesterol and age; one in diabetic individuals and one in non-diabetic individuals.

- Using `geom_smooth` for a visual intuition of a linear relationship
  - ⚠️ here I consider sample DATA **as 2 separate groups** for plotting a smooth line

```
ggplot(nhanes,
       # For *both POINTS & LINES* I split by category (category)
       aes (x = age, y = tot_chol, color = diabetes)) +
  geom_point (aes(alpha = 0.75),
              show.legend = c(size = F, alpha = F) )+
  geom_smooth(method = lm)+
  scale_color_manual(values=c("#005ca1","#9b2339" ))
```

Interaction in regression - visual (RETHINKING)

Users in two categories are represented points; linear relationship is representated by 2 respective line according to diabetes status… the association has DIFFERENT DIRECTION!



Interaction in regression - adding term in model

Let's rethink the model and consider this new *specification*:

$$\mathrm{E}(\mathrm{TotChol}) = \beta_0 + \beta_1(\mathrm{Age}) + \beta_2(\mathrm{Diabetes}) + \beta_3(\mathrm{Diabetes} \times \mathrm{Age}).$$

Where: + the term $(\mathrm{Diabetes} \times \mathrm{Age})$ is the `interaction term` between `diabetes` status and `age`, and β_3 is the coefficient of such interaction term.

- notice the use of `...*...` in the model syntax

```
#fit a model
model_interac2 <- lm(tot_chol ~ age*diabetes, data = nhanes)
model_interac2$coefficients
```

```
(Intercept)             age     diabetesYes age:diabetesYes
4.695702513     0.009638183     1.718704342    -0.033451562
```

Interaction in regression - prediction model

We obtained this predictive model:

$$\widehat{\text{TotChol}} = 4.70 + 0.0096(\text{Age}) + 0.1.72(\text{Diabetes}) - 0.033(\text{Age} \times \text{Diabetes})$$

```
summary(model_interac2)
```

```
Call:
lm(formula = tot_chol ~ age * diabetes, data = nhanes)

Residuals:
    Min      1Q  Median      3Q     Max
-2.3587 -0.7448 -0.0845  0.6307  4.2480

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)      4.695703   0.159691  29.405  < 2e-16 ***
age              0.009638   0.003108   3.101  0.00205 **
diabetesYes      1.718704   0.763905   2.250  0.02492 *
age:diabetesYes -0.033452   0.012272  -2.726  0.00665 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.061 on 469 degrees of freedom
  (27 observations deleted due to missingness)
Multiple R-squared:  0.03229,   Adjusted R-squared:  0.0261
F-statistic: 5.216 on 3 and 469 DF,  p-value: 0.001498
```

Interaction in regression - interactive term interpretation

Given:

$$\widehat{\text{TotChol}} = 4.70 + 0.0096(\text{Age}) + 0.1.72(\text{Diabetes}) - 0.033(\text{Age} \times \text{Diabetes})$$

For diabetics (`DiabetesYes = 1`), the model equation is:

$$\text{TotChol}_{\text{diab}} = 4.70 + 0.0096(\text{Age}) + 1.72(1) - 0.034(\text{Age})(1) \text{ i.e.}$$
$$\text{TotChol}_{\text{diab}} = 6.42 - 0.024(\text{Age})$$

For non-diabetics (`DiabetesYes = 0`), the model equation is:

$$\text{TotChol}_{\text{NOdiab}} = 4.70 + 0.0096(\text{Age}) + 1.72(0) - 0.034(\text{Age})(0) \text{ i.e.}$$
$$\text{TotChol}_{\text{NOdiab}} = 4.70 + 0.0096(\text{Age})$$

Final thoughts/recommendations

- The analyses proposed in this Lab are very similar to the process we go through in real life. The following steps are always included:
  - Thorough **understanding of the input data** and the data collection process
  - Bivariate **analysis of correlation / association** to form an intuition of which explanatory variable(s) may or may not affect the response variable
  - **Diagnostic plots** to verify if the necessary assumptions are met for a linear model to be suitable
  - Upon verifying the assumptions, we **fit data** to hypothesized (linear) model
  - **Assessment of the model performance** (R^2, $\text{Adj.}\, R^2$, $\text{F} - \text{Statistic}$, etc.)
- As we saw with hypothesis testing, the **assumptions** we make (and require) for regression are of utter importance
- Clearly, we only scratched the surface in terms of all the possible predictive models, but we got a hang of the **fundamental steps** and some **useful tools** that might serve us also in more advanced analysis
  - e.g. `broom` (within `tidymodels`), `performace` `rstatix`, `lmtest`