Lab 2: Statistical inference & hypothesis testing

Practice session covering topics discussed in Lecture 2

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

July 26, 2024

GOAL OF TODAY’S PRACTICE SESSION

Consolidate understanding of inferential statistic, through R coding examples conducted on real biostatistics research data.

Lecture 2: topics

- **Purpose and foundations of inferential statistics**
- **Getting to know the “language” of hypothesis testing**
- **Hypothesis testing**
  - review examples
- **A closer look at testing assumptions**
  - more examples dealing with assumptions’ violation

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).

```r
# Load them for this R session

# General
library(fs)          # file/directory interactions
library(here)        # tools find your project's files, based on working directory
library(janitor)     # tools for examining and cleaning data
library(dplyr)       # {tidyverse} tools for manipulating and summarising tidy data
library(forcats)     # {tidyverse} tool for handling factors

# Statistics
library(BSDA)
library(rstatix)     # Pipe-Friendly Framework for Basic Statistical Tests
library(car)         # Companion to Applied Regression
library(multcomp)    # Simultaneous Inference in General Parametric Models

# Plotting
library(ggplot2)     # {tidyverse} tools for plotting
library(ggstatsplot) # 'ggplot2' Based Plots with Statistical Details
library(ggpubr)      # 'ggplot2' Based Publication Ready Plots
library(patchwork)   # Functions for ""Grid" Graphics"composing" plots
library(viridis)     # Colorblind-Friendly Color Maps for R
library(ggthemes)   # Extra Themes, Scales and Geoms for 'ggplot2'
```

Our dataset for today

For the most part, we will refer to a real clinical dataset (for which a *Creative Commons license* was granted) discussed in two articles (also open access) :

- Ahmad, T., Munir, A., Bhatti, S. H., Aftab, M., & Raza, M. A. (2017). ***Survival analysis of heart failure patients: A case study***. PLOS ONE, 12(7), e0181001. https://doi.org/10.1371/journal.pone.0181001
- Chicco, D., & Jurman, G. (2020). ***Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone***. BMC Medical Informatics and Decision Making, 20(1), 16. https://doi.org/10.1186/s12911-020-1023-5

Here is the link to the dataset (or download from workshop website)

- From the UC Irvine Machine Learning Repository Heart Failure Clinical Records

Importing from your project folder (previously downloaded file)

Tip

Make sure to match your own folder structure!

- The function `here` lets me specify the complete path of the destination folder

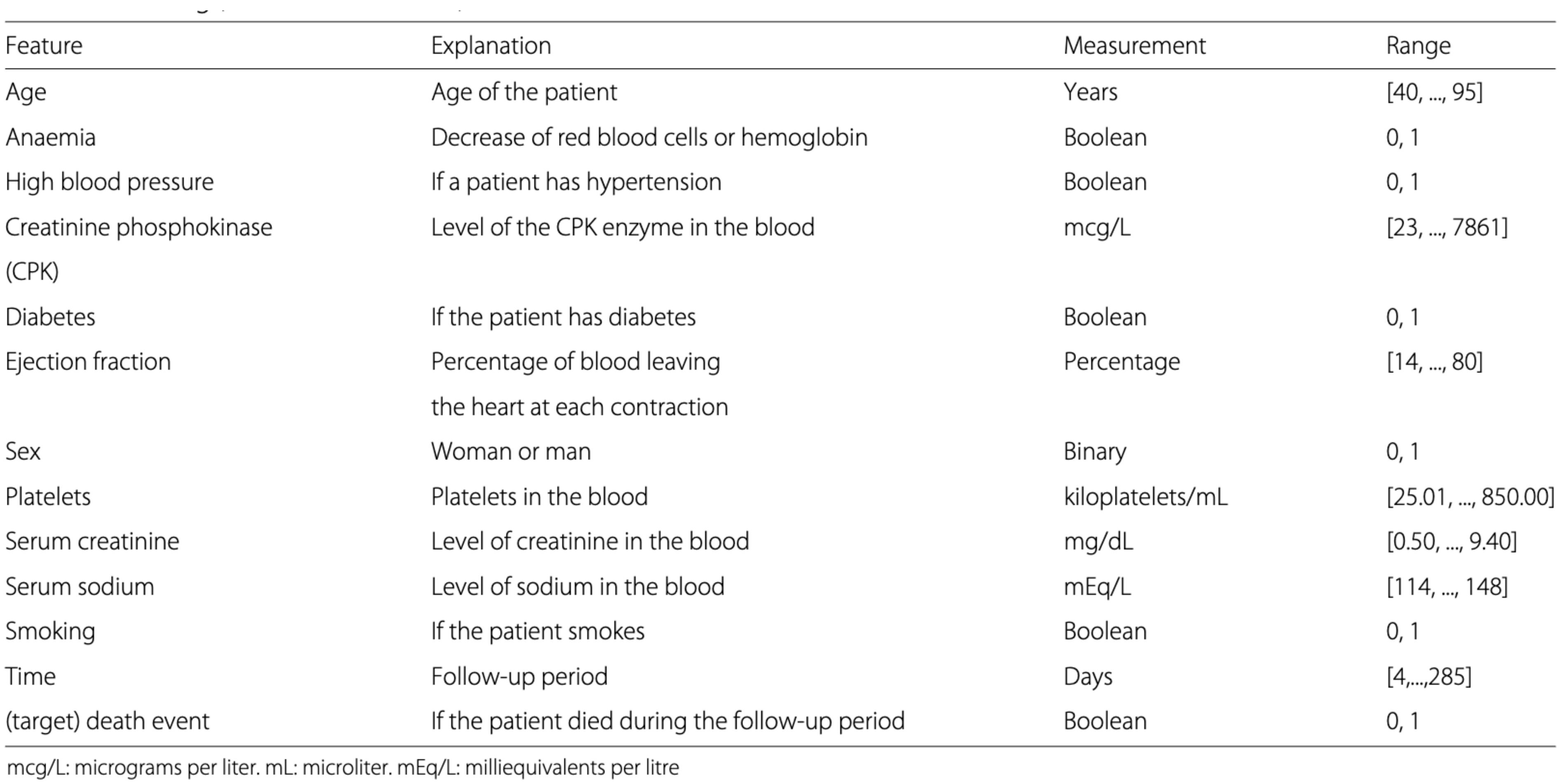
```r
# Check my working directory location
# here::here()

# Use `here` in specifying all the subfolders AFTER the working directory
heart_failure <- read.csv(file = here::here("practice", "data_input", "02_datasets",
                                      "heart_failure_clinical_records_dataset.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)
```

INSPECTING THE “HEART FAILURE” DATASET

What are the variables and their levels of measurement?

The data, containing the medical records of 299 heart failure patient, were collected at the Faisalabad Institute of Cardiology and at the Allied Hospital in Faisalabad (Punjab, Pakistan), during April–December 2015.

Table 1 from the second article (Chicco & Jurman, 2020, p. 3) offers a synthetic explanation of hte observed variables.



Feature	Explanation	Measurement	Range
Age	Age of the patient	Years	[40, ..., 95]
Anaemia	Decrease of red blood cells or hemoglobin	Boolean	0, 1
High blood pressure	If a patient has hypertension	Boolean	0, 1
Creatinine phosphokinase (CPK)	Level of the CPK enzyme in the blood	mcg/L	[23, ..., 7861]
Diabetes	If the patient has diabetes	Boolean	0, 1
Ejection fraction	Percentage of blood leaving the heart at each contraction	Percentage	[14, ..., 80]
Sex	Woman or man	Binary	0, 1
Platelets	Platelets in the blood	kiloplatelets/mL	[25.01, ..., 850.00]
Serum creatinine	Level of creatinine in the blood	mg/dL	[0.50, ..., 9.40]
Serum sodium	Level of sodium in the blood	mEq/L	[114, ..., 148]
Smoking	If the patient smokes	Boolean	0, 1
Time	Follow-up period	Days	[4,...,285]
(target) death event	If the patient died during the follow-up period	Boolean	0, 1

mcg/L: micrograms per liter. mL: microliter. mEq/L: milliequivalents per litre

Look into the dataset just loaded in the R environment

Recall some `base R` functions from Lab 1

```r
# What variables are included in this dataset?
colnames(heart_failure)
```

```
 [1] "age"                      "anaemia"                 
 [3] "creatinine_phosphokinase" "diabetes"                
 [5] "ejection_fraction"        "high_blood_pressure"     
 [7] "platelets"                "serum_creatinine"        
 [9] "serum_sodium"             "sex"                     
[11] "smoking"                  "time"                    
[13] "DEATH_EVENT"             
```

```r
# How many observations & variables?
nrow(heart_failure)
```

```
[1] 299
```

```r
# How many rows & columns?
dim(heart_failure)
```

```
[1] 299  13
```

Inspect the dataframe structure (base R)

```r
# What does the dataframe look like?
str(heart_failure)
```

```
'data.frame':	299 obs. of  13 variables:
 $ age                     : num  75 55 65 50 65 90 75 60 65 80 ...
 $ anaemia                 : int  0 0 0 1 1 1 1 1 0 1 ...
 $ creatinine_phosphokinase: int  582 7861 146 111 160 47 246 315 157 123 ...
 $ diabetes                : int  0 0 0 0 1 0 0 1 0 0 ...
 $ ejection_fraction       : int  20 38 20 20 20 40 15 60 65 35 ...
 $ high_blood_pressure     : int  1 0 0 0 0 1 0 0 0 1 ...
 $ platelets               : num  265000 263358 162000 210000 327000 ...
 $ serum_creatinine        : num  1.9 1.1 1.3 1.9 2.7 2.1 1.2 1.1 1.5 9.4 ...
 $ serum_sodium            : int  130 136 129 137 116 132 137 131 138 133 ...
 $ sex                     : int  1 1 1 1 0 1 1 1 0 1 ...
 $ smoking                 : int  0 0 1 0 0 1 0 1 0 1 ...
 $ time                    : int  4 6 7 7 8 8 10 10 10 10 ...
 $ DEATH_EVENT             : int  1 1 1 1 1 1 1 1 1 1 ...
```

Inspect the dataframe structure (`skimr`)

Remember the `skimr` function `skim`?

```r
# some variables
heart_failure %>% skimr::skim( age, DEATH_EVENT )

# the whole dataframe
heart_failure %>% skimr::skim()
```

You try…

Run `skimr::skim()` on your own either on the whole dataset or on any specific variable

- notice there are no (missing values) `NAs` in any of the variables

Recode some variables for later ease of analysis

I may need some variables coded as `factor` (e.g. categorical variables for plotting), and, while I am at it, I can add clearer labels for the variables' levels. Here, we are:

- using tidyverse packages `dplyr` and `forcats`
- adding new (recoded) variables called "*`oldname_f`*"

```r
heart_failure <-heart_failure %>%
  dplyr::mutate(DEATH_EVENT_f = as.factor(DEATH_EVENT) %>%
                  forcats::fct_recode("died" = "1", "survived" = "0")) %>%
  dplyr::mutate(sex_f = as.factor(sex) %>%
                  forcats::fct_recode("male" = "1", "female" = "0"))

# check
table(heart_failure$DEATH_EVENT_f)
```

```
survived     died 
     203       96 
```

```r
table(heart_failure$sex_f)
```

```
female   male 
   105    194 
```

Some more dummy variables recoded as factor

[Mostly for illustration: it’s totally fine (if not preferable) to keep these as binary [0,1] variables]

- It’s worth learning the useful function `dplyr::across`[1], which allows to iteratively transform several columns at once!

```r
# Recode as factor with levels "yes" (= 1), "no" (= 0)
fct_cols = c("anaemia", "diabetes", "high_blood_pressure", "smoking" )

heart_failure <- heart_failure  %>%
  ## ---- 1st create new cols as "factor versions" of old cols
  dplyr::mutate(
    # let's introduce `across` function
    dplyr::across(
      # Columns to transform
      .cols = all_of(fct_cols),
      # Functions to apply to each col
      .fns =  ~as.factor (.x),
      # new name to apply where "{.col}" stands for the selected column
      .names = "{.col}_f")) %>%
  ## ---- 2nd create new cols as "factor versions" of old cols
  dplyr::mutate(
    dplyr::across(
      # Columns to transform 2 conditions
      .cols = ends_with("_f") & !matches(c( "DEATH_EVENT_f", "sex_f" )) ,
      # Functions to apply to each col(different syntax)
      .fns = ~forcats::fct_recode(.x,  yes = "1", no = "0" )))
```

[1] This is a bit more advanced, but it will save a lot of typing in some situations…

(Small digression on `dplyr::across`)

Notice how `dplyr::across(.cols = ..., .fns = ..., .names = ...)` has these arguments:

1. `.cols =` to select the columns which we want to transform (i.e. `fct_cols`)
    - with help from `tidyselect` functions: `all_of`, `ends_with`, and `matches`
2. `.fns = ~function(.x)` to specify the `function`
    - where `~function(.x)` uses the “anonymous function” syntax of the `tidyverse`
    - and `.x` inside the function is a “stand in” for *each of the columns selected*
3. [*optional*] `.names =` to name the new cols created using `{.col}` in place of each of the transformed columns

```r
## ---- 1st create new cols as "factor versions" of old cols
heart_failure <- heart_failure  %>%
  dplyr::mutate(
    dplyr::across(
      .cols = all_of(fct_cols),
      .fns = ~as.factor (.x),
      # (optional)
      .names = "{.col}_f")) %>%
  ## ---- 2nd create new cols as "factor versions" of old cols
  dplyr::mutate(
    dplyr::across(
      .cols = ends_with("_f") & !matches(c( "DEATH_EVENT_f", "sex_f" )) ,
      .fns =  ~forcats::fct_recode(.x,  yes = "1", no = "0" )))
```

VISUAL DATA EXPLORATION FOR THE “HEART FAILURE”

CONTINUOUS VARIABLES

Why is visual exploration important?

- Gaining insight on the variables (range, outliers, missing data)
- Preliminary check of assumptions for parametric hypothesis testing:
  - normally distributed outcome variables?
  - homogeneity of variance across groups?

Let’s explore the **Heart failure dataset** with some data visualization…

- Following the referenced articles (which were mostly interested in predict mortality based on patients’ characteristics), we will take the categorical, binary variable `DEATH_EVENT_f` as our main criterion to split the sample (into *survived* and *dead* patients) to explore any significant difference between groups in terms of means of known quantitative features.
- We will look at both:
  - **continuous variables** in the dataset (with the Probability Density Function (PDF))
  - **discrete variables** in the dataset (with the Probability Mass Function (PMF))

Age

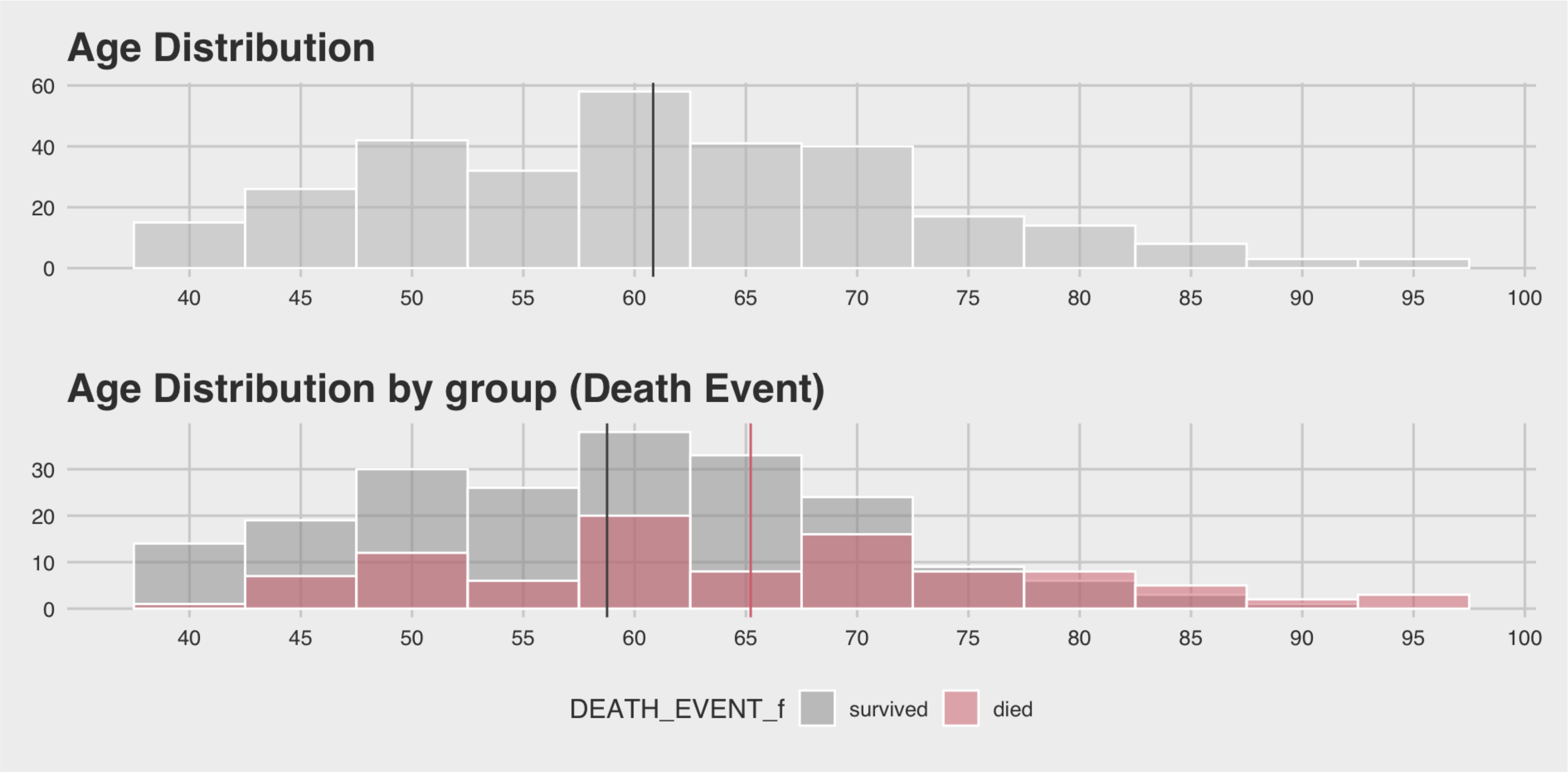
Introducing the handy R package `patchwork` which lets us compose different plots in a very simple and intuitive way

- (check it out with `??patchwork`)

```r
age <-ggplot(heart_failure,aes(x = age ))+
  geom_histogram(binwidth = 5, color = "white", fill = "grey",alpha = 0.5)+
  geom_vline(aes(xintercept = mean(age)), color = "#4c4c4c")+
  theme_fivethirtyeight()+
  labs(title = "Age Distribution" )+
  scale_x_continuous(breaks = seq(40,100,5))

age2 <-ggplot(heart_failure, aes(x = age, fill = DEATH_EVENT_f))+
  geom_histogram(binwidth = 5, position = "identity",alpha = 0.5,color = "white")+
  geom_vline(aes(xintercept = mean(age[DEATH_EVENT == 0])), color = "#4c4c4c")+
  geom_vline(aes(xintercept = mean(age[DEATH_EVENT==1])), color = "#d8717b")+
  theme_fivethirtyeight()+
  scale_fill_manual(values = c("#999999", "#d8717b"))+
  labs(title =  "Age Distribution by group (Death Event)")+
  scale_x_continuous(breaks = seq(40,100,5))

# patchwork
library(patchwork)
age + age2 + plot_layout(ncol = 1)
```

Age

As the age increases, the incidence of death event seems to increase

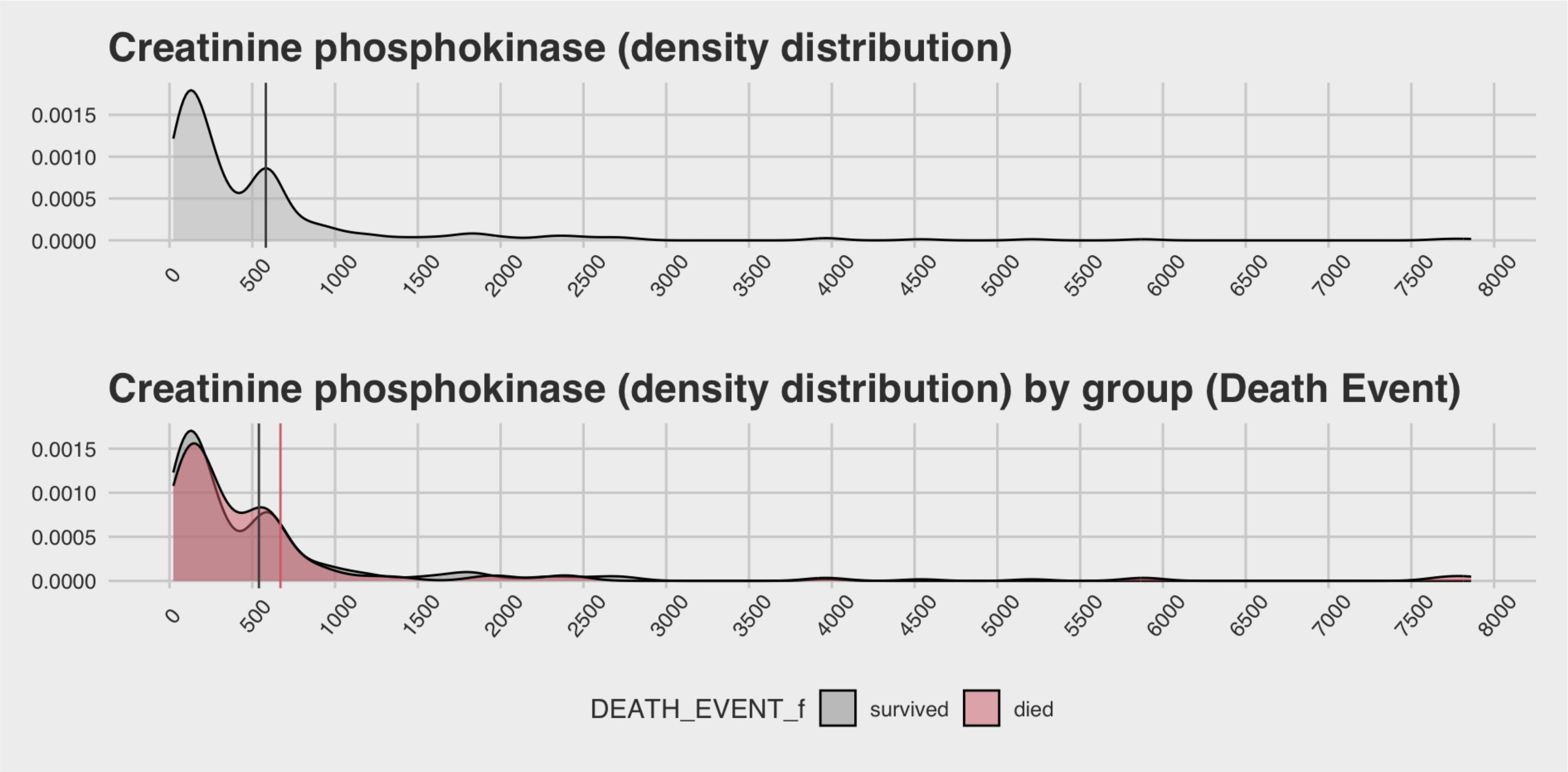


Creatinine Phosphokinase (CPK)

```r
cpk <- ggplot(heart_failure,aes(x = creatinine_phosphokinase))+
  geom_density(fill = "gray", alpha = 0.5)+
  scale_x_continuous(breaks = seq(0,8000, 500))+
  geom_vline(aes(xintercept = mean(creatinine_phosphokinase)), color = "#4c4c4c")+
  theme_fivethirtyeight()+
  theme(axis.text.x = element_text(angle=50, vjust=0.75))+
  labs(title = "Creatinine phosphokinase (density distribution)" )+
  theme(plot.caption = element_text(hjust = 0.5, face = "italic"))

cpk2 <- ggplot(heart_failure,aes(x = creatinine_phosphokinase,fill = DEATH_EVENT_f))+
  geom_density(alpha = 0.5)+theme_fivethirtyeight()+
  scale_fill_manual(values = c("#999999", "#d8717b"))+
  scale_x_continuous(breaks = seq(0,8000, 500))+
  geom_vline(aes(xintercept = mean(creatinine_phosphokinase[DEATH_EVENT == 0])),
             color = "#4c4c4c")+
  geom_vline(aes(xintercept = mean(creatinine_phosphokinase[DEATH_EVENT==1])),
             color = "#d8717b")+
  theme_fivethirtyeight()+
  theme(axis.text.x = element_text(angle=50, vjust=0.75))+
  labs(title =  "Creatinine phosphokinase (density distribution) by group (Death Event)")

cpk + cpk2 + plot_layout(ncol = 1)
```

Creatinine Phosphokinase (CPK)

This definitely doesn’t look like a normal distribution!

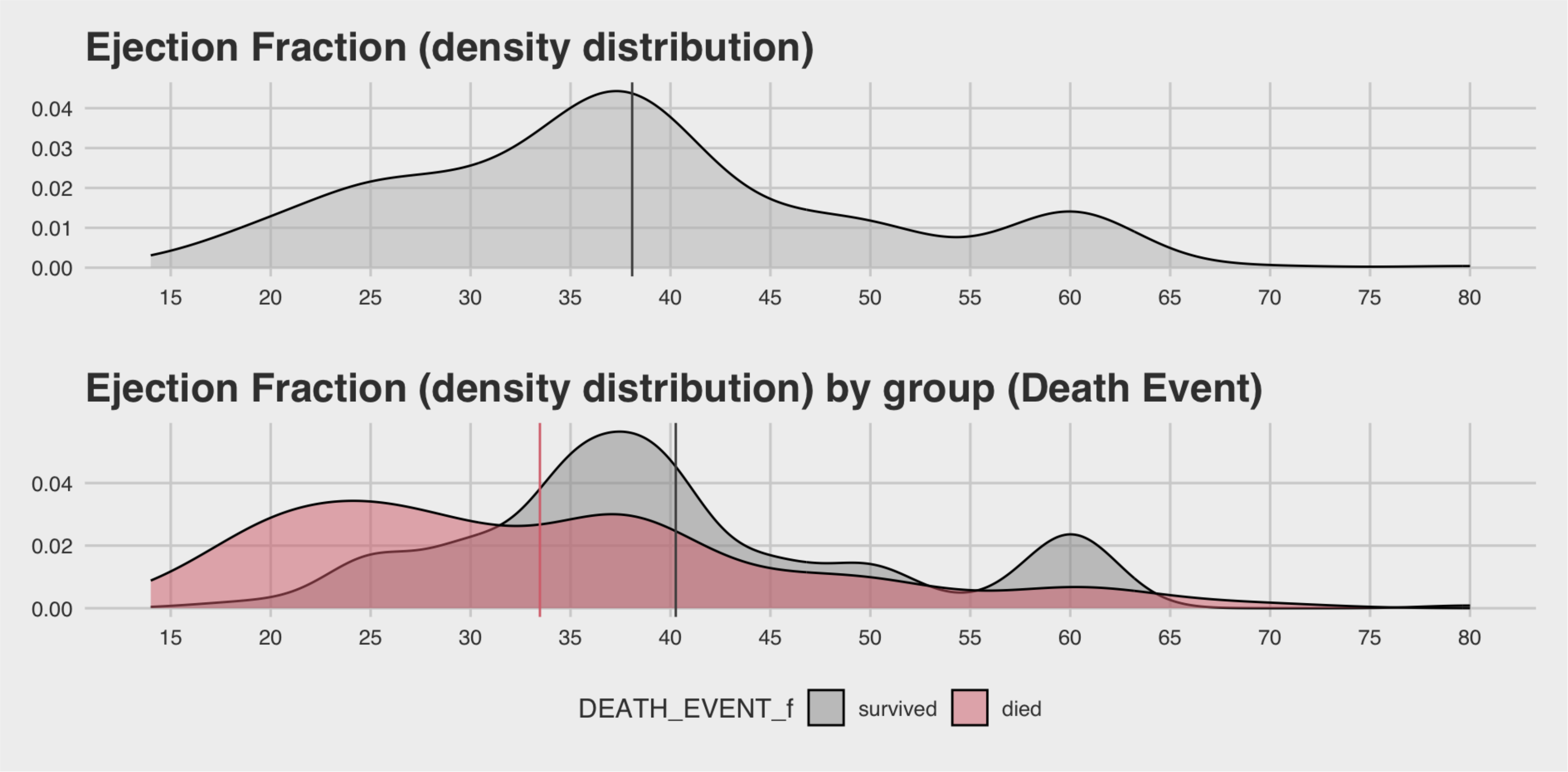


Ejection Fraction

```r
ejf <- ggplot(heart_failure,aes(x = ejection_fraction))+
  geom_density(fill = "gray", alpha = 0.5)+
  scale_x_continuous(breaks = seq(0,100, 5))+
  geom_vline(aes(xintercept = mean(ejection_fraction)), color = "#4c4c4c")+
  theme_fivethirtyeight()+
  labs(title = "Ejection Fraction (density distribution)" )+
  theme(plot.caption = element_text(hjust = 0.5, face = "italic"))

ejf2 <- ggplot(heart_failure,aes(x = ejection_fraction,fill = DEATH_EVENT_f))+
  geom_density(alpha = 0.5)+theme_fivethirtyeight()+
  scale_x_continuous(breaks = seq(0,100, 5))+
  scale_fill_manual(values = c("#999999", "#d8717b"))+
  geom_vline(aes(xintercept = mean(ejection_fraction[DEATH_EVENT == 0])),
             color = "#4c4c4c")+
  geom_vline(aes(xintercept = mean(ejection_fraction[DEATH_EVENT==1])),
             color = "#d8717b")+
  labs(title =  "Ejection Fraction (density distribution) by group (Death Event)")+
  theme_fivethirtyeight()

ejf + ejf2 + plot_layout(ncol = 1)
```

Ejection Fraction

This also doesn't look like a normal distribution… and there is a remarkable change in the *probability density function* (PDF) shape when we introduce the grouping variable

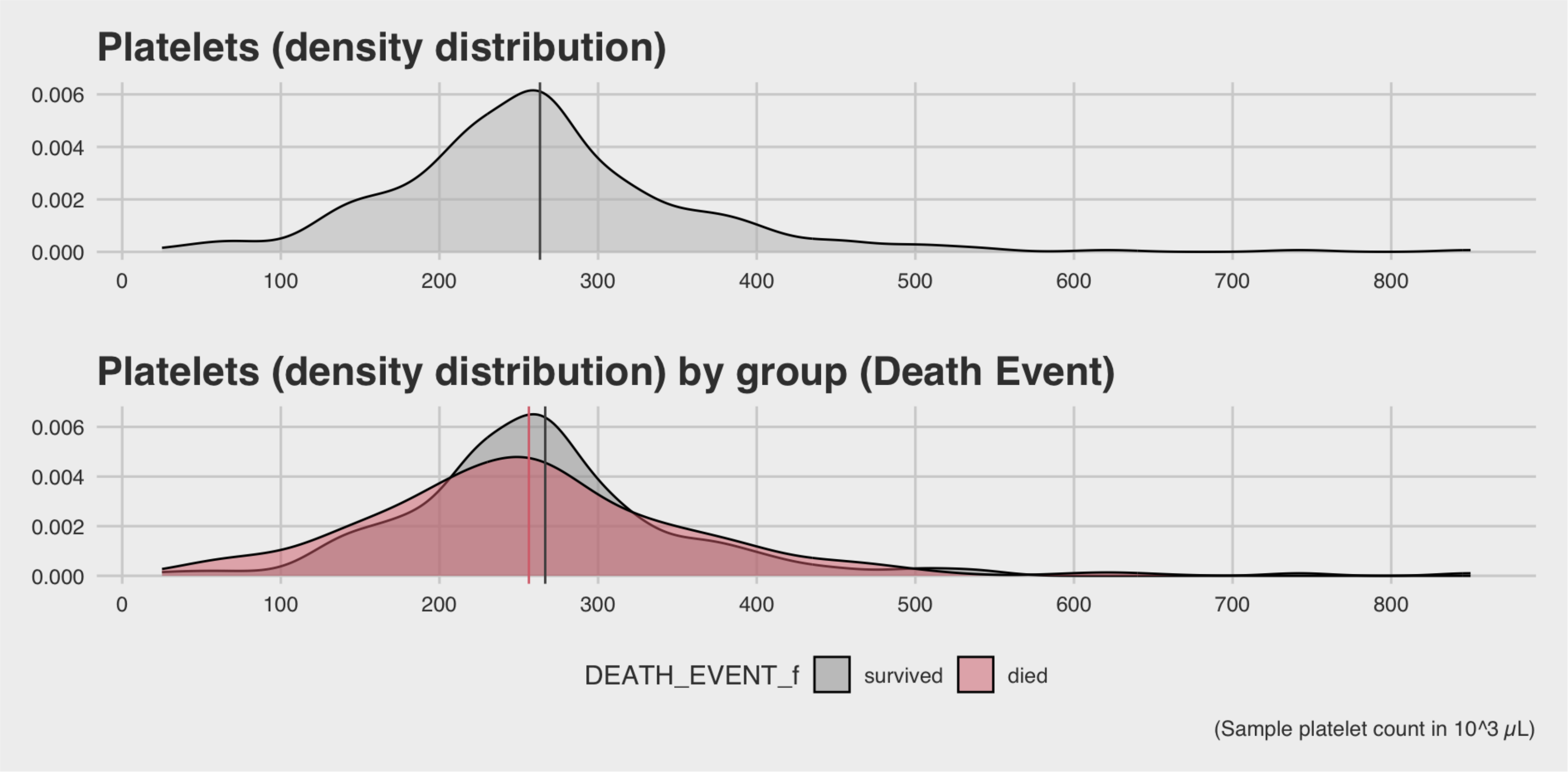


Platelets

```r
# normalize the var for readability
heart_failure  <-  heart_failure %>%  dplyr::mutate(plat_norm = platelets/1000)

plat <- ggplot(heart_failure,aes(x = plat_norm))+
  geom_density(fill = "gray", alpha = 0.5)+
  scale_x_continuous(breaks = seq(0,800, 100))+
  geom_vline(aes(xintercept = mean(plat_norm)), color = "#4c4c4c")+
  theme_fivethirtyeight()   +
  labs(title =  "Platelets (density distribution)",
       y = "Density", x = "Sample platelet count (in 10^3 μL)")

plat2 <- ggplot(heart_failure,aes(x = plat_norm,fill = DEATH_EVENT_f))+
  geom_density(alpha = 0.5)+theme_fivethirtyeight()+
  scale_x_continuous(breaks = seq(0,800, 100))+
  scale_fill_manual(values = c("#999999", "#d8717b"))+
  geom_vline(aes(xintercept = mean(plat_norm[DEATH_EVENT == 0])),
             color = "#4c4c4c")+
  geom_vline(aes(xintercept = mean(plat_norm[DEATH_EVENT==1])),
             color = "#d8717b")+
  theme_fivethirtyeight()   +
  labs(title =  "Platelets (density distribution) by group (Death Event)",
       caption = "(Sample platelet count in 10^3 μL)")

plat + plat2 + plot_layout(ncol = 1)
```

Platelets

Here the probability distributions resemble a Normal one and we observe more uniformity in the mean/variance across the 2 groups

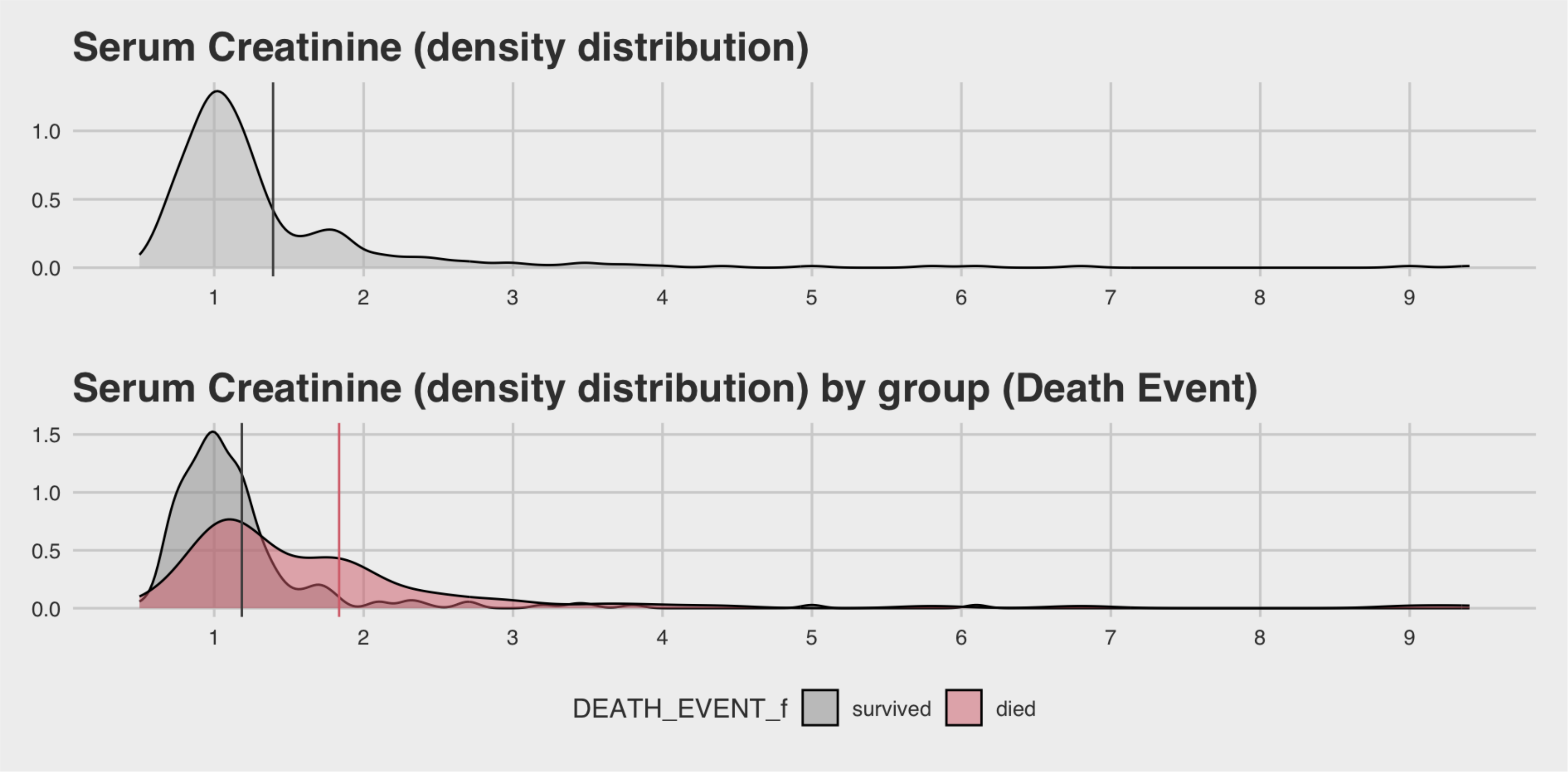


Serum Creatinine

```r
ser_cr <- ggplot(heart_failure,aes(x = serum_creatinine))+
  geom_density(fill = "gray", alpha = 0.5)+
  scale_x_continuous(breaks = seq(0,10, 1))+
  geom_vline(aes(xintercept = mean(serum_creatinine)), color = "#4c4c4c")+
  theme_fivethirtyeight()+
  labs(title = "Serum Creatinine (density distribution)" )+
  theme(plot.caption = element_text(hjust = 0.5, face = "italic"))

ser_cr2 <- ggplot(heart_failure,aes(x = serum_creatinine,fill = DEATH_EVENT_f))+
  geom_density(alpha = 0.5)+theme_fivethirtyeight()+
  scale_x_continuous(breaks = seq(0,10, 1))+
  scale_fill_manual(values = c("#999999", "#d8717b"))+
  geom_vline(aes(xintercept = mean(serum_creatinine[DEATH_EVENT == 0])),
             color = "#4c4c4c")+
  geom_vline(aes(xintercept = mean(serum_creatinine[DEATH_EVENT==1])),
             color = "#d8717b")+
  labs(title =  "Serum Creatinine (density distribution) by group (Death Event)")+
  theme_fivethirtyeight()

ser_cr + ser_cr2 + plot_layout(ncol = 1)
```

Serum Creatinine

Another continuous random variable with a non-normal distribution (long right tails) and a seemingly important difference in variance between the groups.

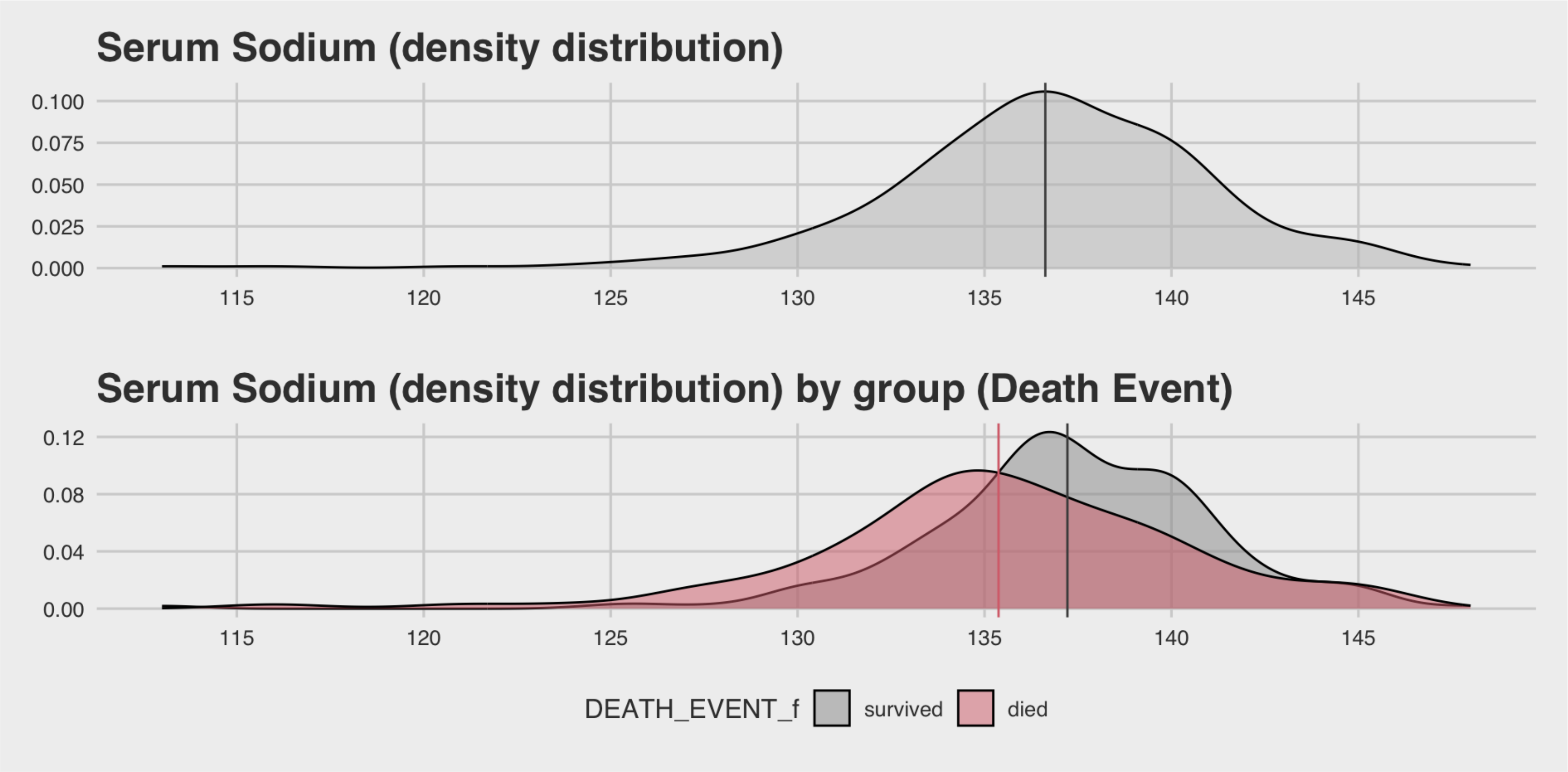


Serum Sodium

```r
ser_sod <- ggplot(heart_failure,aes(x = serum_sodium))+
  geom_density(fill = "gray", alpha = 0.5)+
  scale_x_continuous(breaks = seq(0,150, 5))+
  geom_vline(aes(xintercept = mean(serum_sodium)), color = "#4c4c4c")+
  theme_fivethirtyeight()+
  labs(title = "Serum Sodium (density distribution)" )

ser_sod2 <- ggplot(heart_failure,aes(x = serum_sodium,fill = DEATH_EVENT_f))+
  geom_density(alpha = 0.5)+
  scale_x_continuous(breaks = seq(0,150, 5))+
  scale_fill_manual(values = c("#999999", "#d8717b"))+
  geom_vline(aes(xintercept = mean(serum_sodium[DEATH_EVENT == 0])),
             color = "#4c4c4c")+
  geom_vline(aes(xintercept = mean(serum_sodium[DEATH_EVENT==1])),
             color = "#d8717b")+
  theme_fivethirtyeight()+
  labs(title =  "Serum Sodium (density distribution) by group (Death Event)")+
  theme_fivethirtyeight()

ser_sod + ser_sod2 + plot_layout(ncol = 1)
```

Serum Sodium

Same as above, except for the long left tails…



VISUAL DATA EXPLORATION FOR THE “HEART FAILURE”

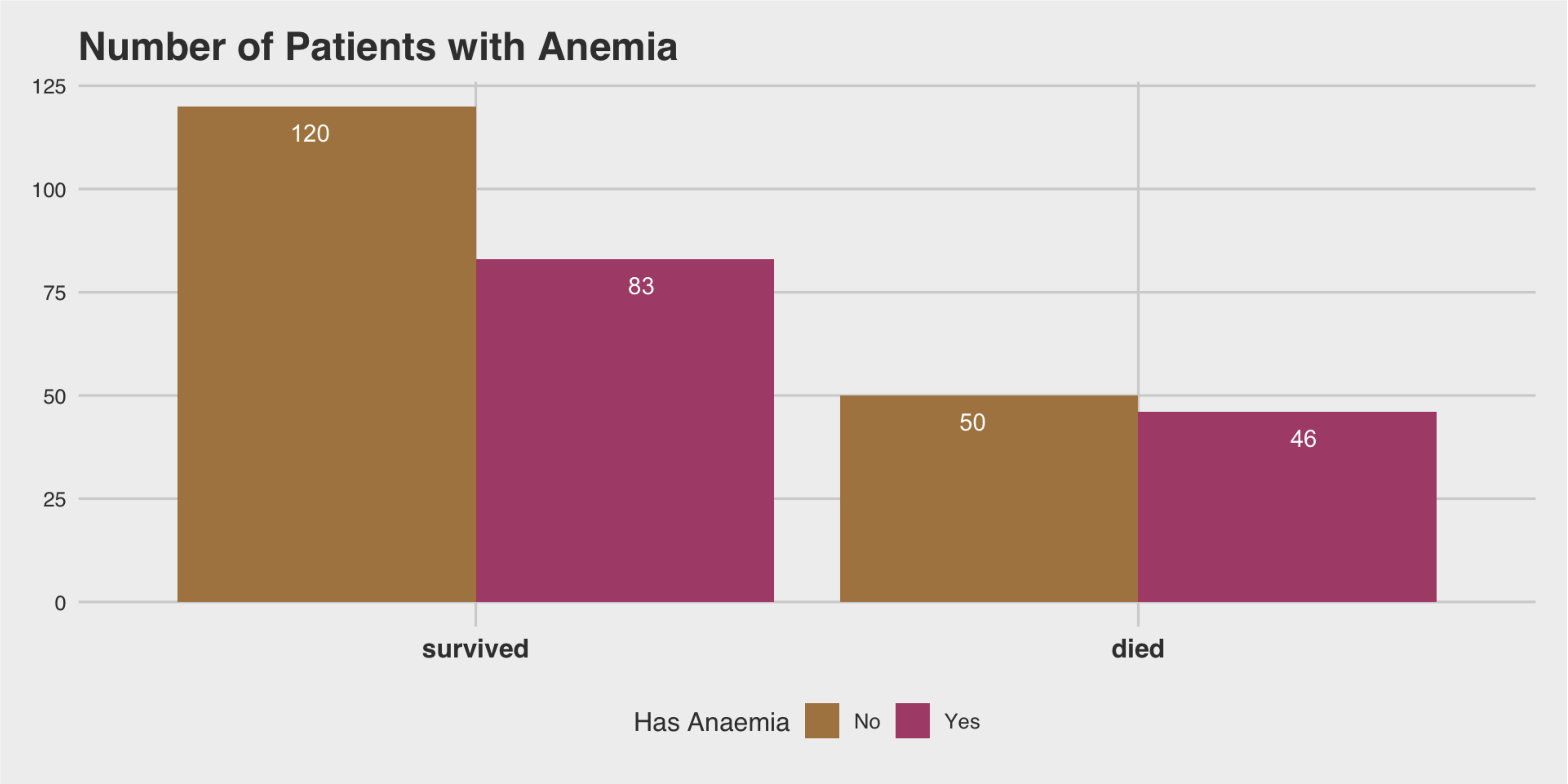
DISCRETE VARIABLES

Anaemia

```r
anem <- ggplot(heart_failure, aes(x = forcats::fct_infreq(DEATH_EVENT_f ),
                                  fill = anaemia_f ))+
  geom_bar(position = "dodge")+
  ## add count labels
  geom_text(stat = "count", aes(label = ..count..),
            ## make labels suit the dodged bars
            position=position_dodge(width = 1 ),
            hjust=0.5, vjust=2,color = "white") +
  theme_fivethirtyeight() +
  #scale_x_discrete(labels  = c("Death Event:No","Death Event:Yes"))+
  scale_fill_manual(values = c("#af854f", "#af4f78"),
                    name = "Has Anaemia",
                    labels = c("No","Yes"))+
  labs(title = "Number of Patients with Anemia") +
  theme(#axis.text.x = element_text(angle=50, vjust=0.75),
    axis.text.x = element_text(size=12,face="bold"))

anem
```

Anaemia

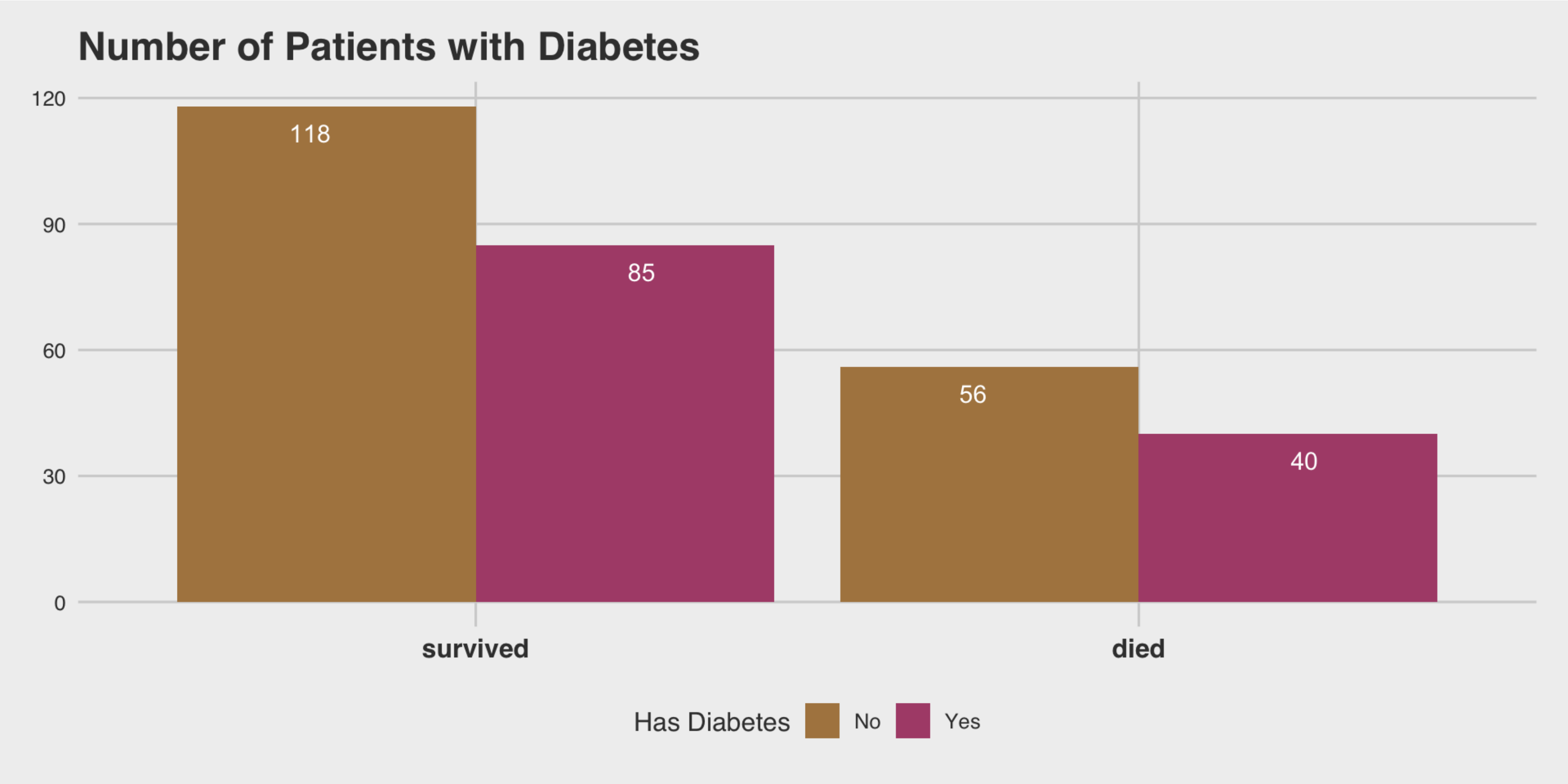
There seems to be a greater incidence of anaemia in group ‘died’



Diabetes

```r
diab <- ggplot(heart_failure,
               aes(x = forcats::fct_infreq(DEATH_EVENT_f ), fill = diabetes_f ))+
  geom_bar(position = "dodge")+
  ## add count labels
  geom_text(stat = "count", aes(label = ..count..),
            ## make labels suit the dodged bars
            position=position_dodge(width = 1 ),
            hjust=0.5, vjust=2,color = "white", size =4) +
  theme_fivethirtyeight() +
  #scale_x_discrete(labels  = c("Death Event:No","Death Event:Yes"))+
  scale_fill_manual(values = c("#af854f", "#af4f78"),
                    name = "Has Diabetes",
                    labels = c("No","Yes"))+
  labs(title = "Number of Patients with Diabetes") +
  theme(#axis.text.x = element_text(angle=50, vjust=0.75),
    axis.text.x = element_text(size=12,face="bold"))

diab
```

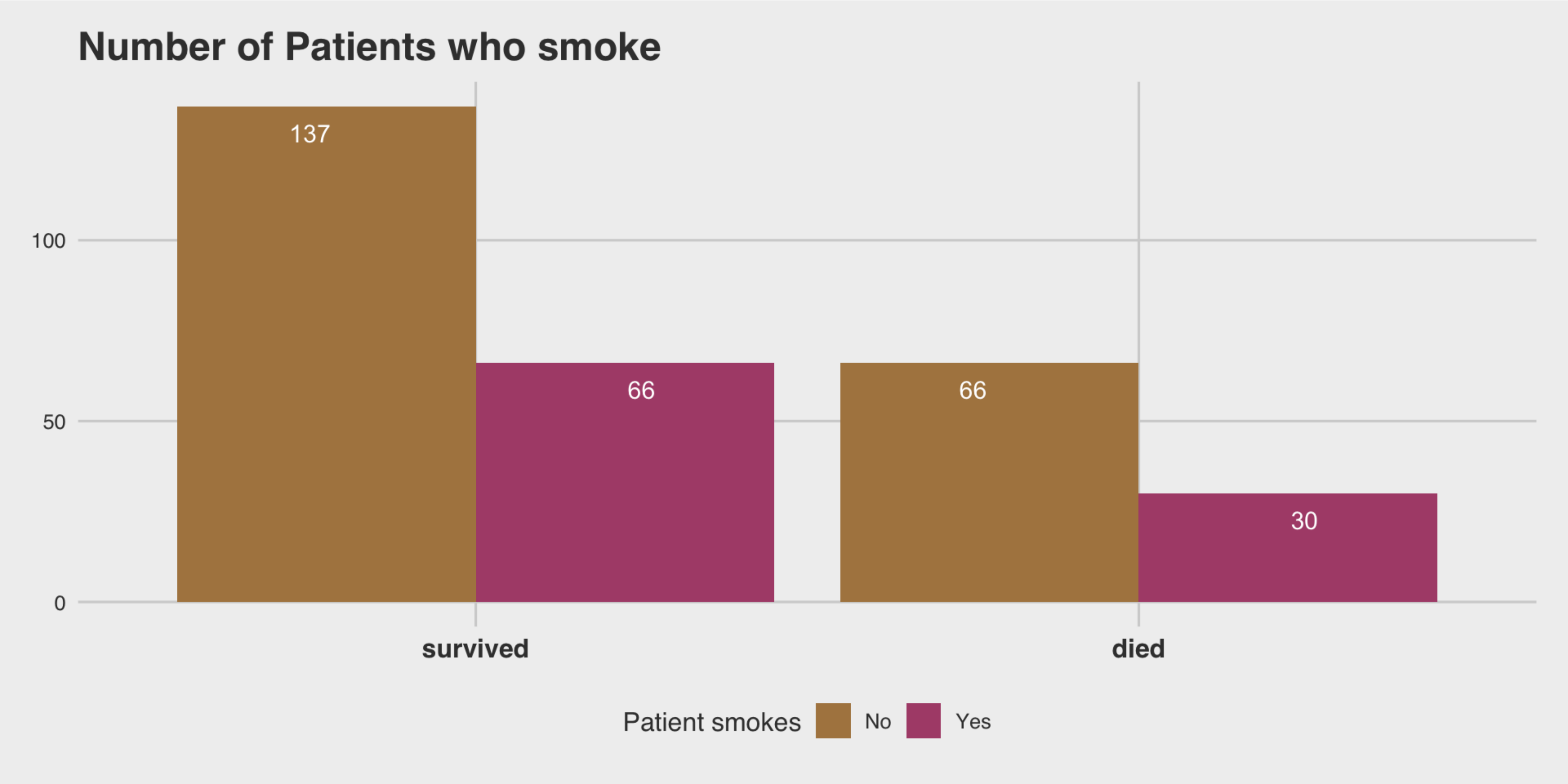
Diabetes


Number of Patients with Diabetes
120
90
60
30
0
118
85
56
40
survived
died
Has Diabetes
No
Yes

Smoking

```r
smok <- ggplot(heart_failure, aes(x = forcats::fct_infreq(DEATH_EVENT_f ),
                                  fill = smoking_f ))+
  geom_bar(position = "dodge")+
  ## add count labels
  geom_text(stat = "count", aes(label = ..count..),
            ## make labels suit the dodged bars
            position=position_dodge(width = 1 ),
            hjust=0.5, vjust=2,color = "white", size =4) +
  theme_fivethirtyeight() +
  #scale_x_discrete(labels  = c("Death Event:No","Death Event:Yes"))+
  scale_fill_manual(values = c("#af854f", "#af4f78"),
                    name = "Patient smokes",
                    labels = c("No","Yes"))+
  labs(title = "Number of Patients who smoke") +
  theme(#axis.text.x = element_text(angle=50, vjust=0.75),
    axis.text.x = element_text(size=12,face="bold"))

smok
```

Smoking

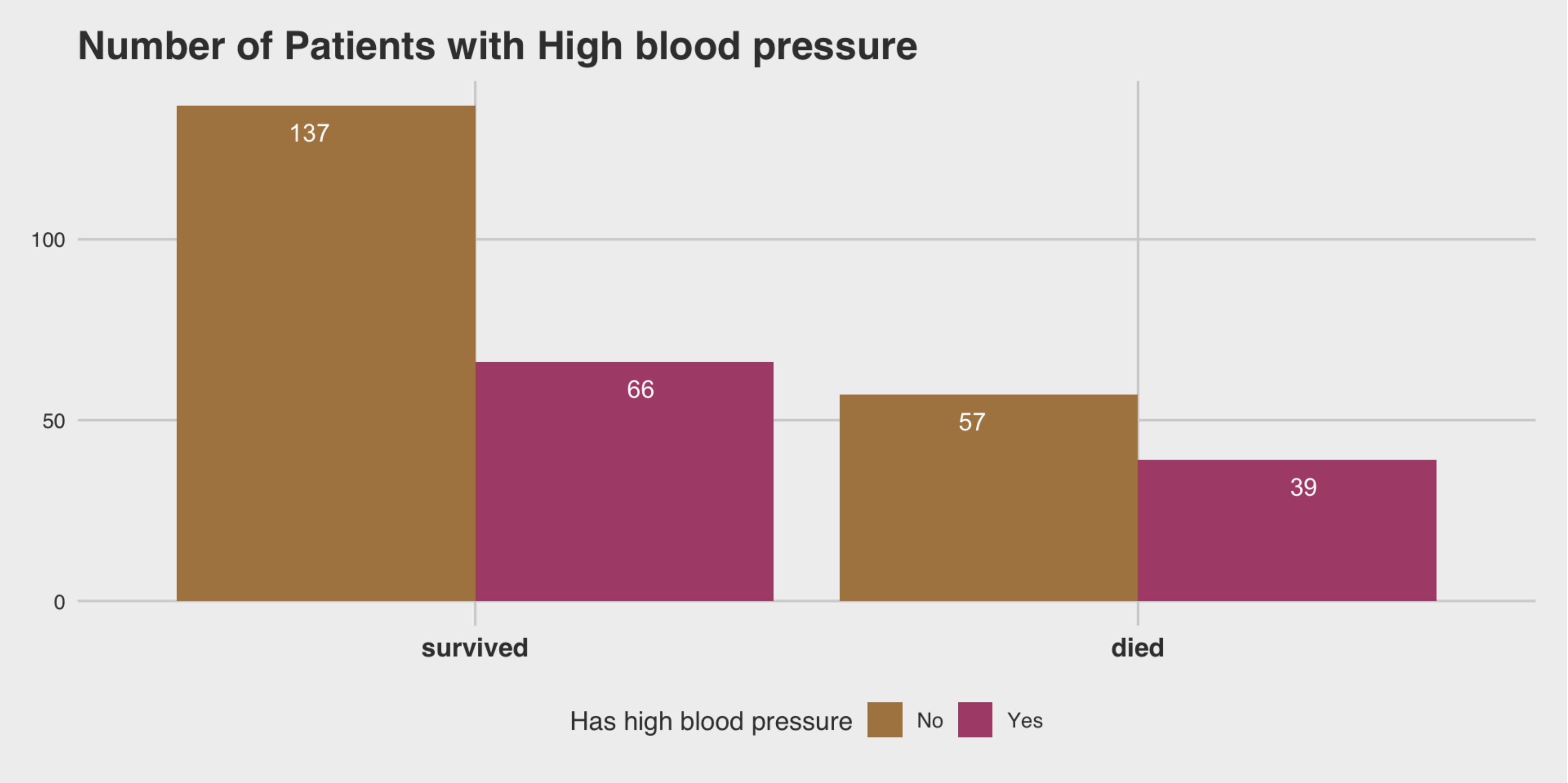

Number of Patients who smoke
137
66
66
30
100
50
0
survived
died
Patient smokes
No
Yes

High blood pressure

```r
hbp <- ggplot(heart_failure, aes(x = forcats::fct_infreq(DEATH_EVENT_f ),
                                  fill = high_blood_pressure_f ))+
  geom_bar(position = "dodge")+
    ## add count labels
  geom_text(stat = "count", aes(label = ..count..),
            ## make labels suit the dodged bars
            position=position_dodge(width = 1 ),
            hjust=0.5, vjust=2,color = "white", size =4) +
  theme_fivethirtyeight() +
  #scale_x_discrete(labels  = c("Death Event:No","Death Event:Yes"))+
  scale_fill_manual(values = c("#af854f", "#af4f78"),
                    name = "Has high blood pressure",
                    labels = c("No","Yes"))+
  labs(title = "Number of Patients with High blood pressure") +
  theme(#axis.text.x = element_text(angle=50, vjust=0.75),
    axis.text.x = element_text(size=12,face="bold"))

hbp
```

High blood pressure

There is also a greater incidence of high blood pressure in group ‘died’



HYPOTHESIS TESTNG - some examples -

Let’s continue to explore data from the **heart failure patients’ dataset**, but this time using **hypothesis testing** as we learned in Lecture 2. We will do two types of test:

1. Comparing a sample **against a *hypothetical* general population**
2. Testing if mean variables’ **differences between the two groups of patients** (those who survived after heart failure event and those who didn’t) is statistically significant

— EXAMPLE A —

(1 sample | n > 30 | Z test)

Comparing sample mean to a hypothesized population mean (with Z test)

Stating the above hypotheses more formally:

What is the population Total Platelet Count (TPC) mean for all people who suffered of heart failure (μ_{HF})?

- H_0 : there is no difference in mean TPC between patients who suffered heart failure and the general population
  - $\mu_{HF} = 236$ -> hypothesis of no effect or (“no difference”)
- H_a : there is a difference in mean TPC between patients who have suffered heart failure and the general population (“some effect”). This can be formalized as either:
  - $\mu_{HF} < 236$ (one-sided test), or
  - $\mu_{HF} > 236$ (one-sided test), or
  - $\mu_{HF} \neq 236$ (two-sided test)

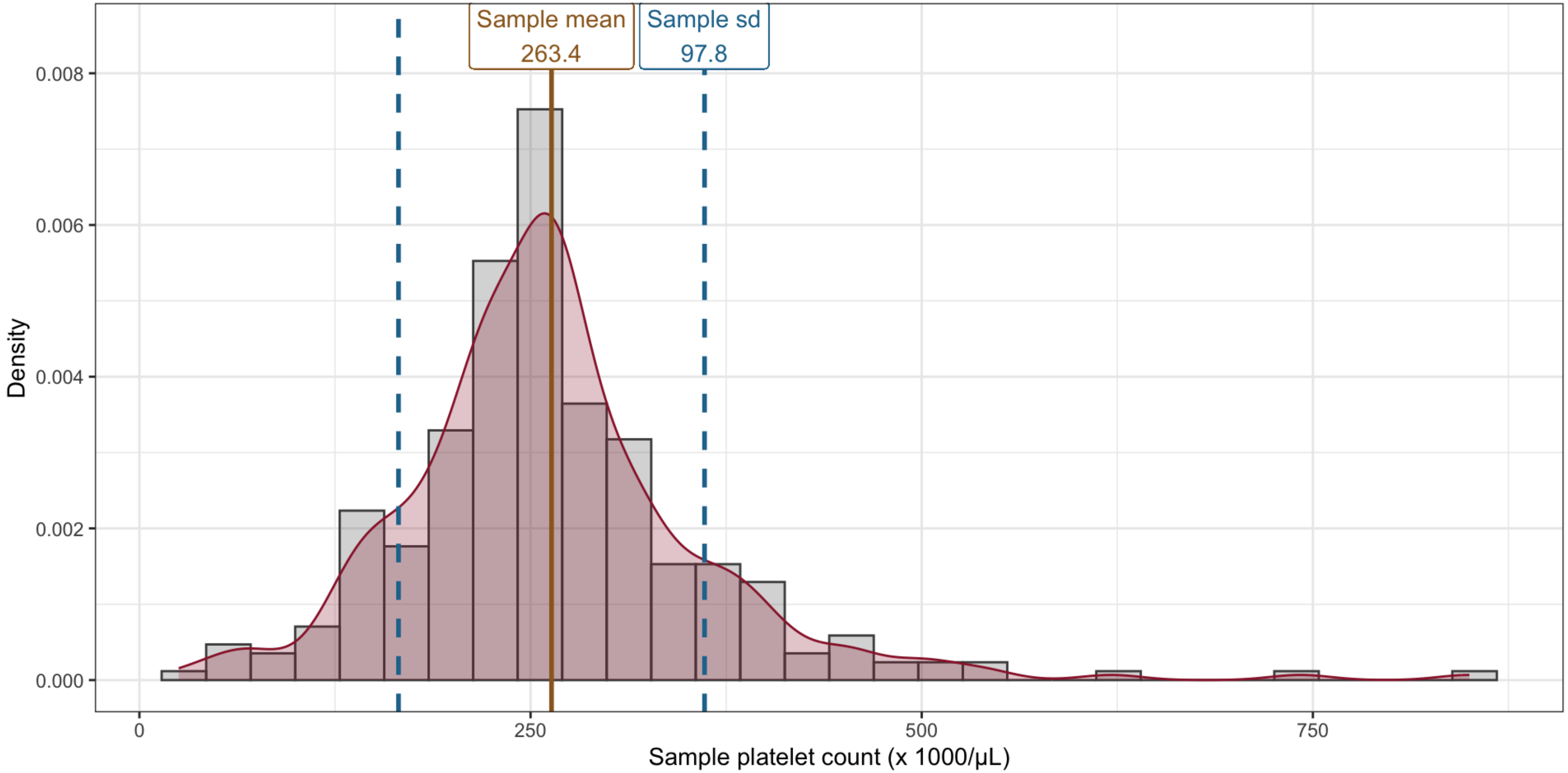
1. Question: How does the mean platelets count in the patients’ sample compare against a r

```r
# compute mean & sd for plot
mean_plat_p <- round(mean(heart_failure$plat_norm), digits = 1)
sd_plat_p <- round(sd(heart_failure$plat_norm), digits = 1)

heart_failure %>%
  ggplot(aes(x = plat_norm))+
  geom_histogram(aes(y = ..density..), bins=30, alpha=0.25, colour = "#4c4c4c") +
  geom_density(colour ="#9b2339", alpha=0.25, fill = "#9b2339") +
  # add mean vertical line
  geom_vline(xintercept = mean_plat_p, na.rm = FALSE,size = 1,color= "#9b6723") +
  # add also +/- 1sd
  geom_vline(aes(xintercept = mean_plat_p + sd_plat_p),
             color = "#23749b", size = 1, linetype = "dashed") +
  geom_vline(aes(xintercept = mean_plat_p - sd_plat_p),
             color = "#23749b", size = 1, linetype = "dashed") +
  # add annotations with the mean value
  geom_label(aes(x=mean_plat_p,  y=0.0085, label=paste0("Sample mean\n",mean_plat_p)),
             color = "#9b6723") +
  geom_label(aes(x=361,  y=0.0085, label=paste0("Sample sd\n",sd_plat_p)),
             color = "#23749b") +
  theme_bw() +  labs(y = "Density", x = "Sample platelet count (x 1000/µL)")
```

General population data taken from the literature (See Wongsaengsak, Dennis, Arevalo, Ball, & Nugent, 2019).

1. Question: How does the mean platelets count in the patients’ sample compare against a r

For a general population, the Total Platelet Count (TPL) has **μ**=236 (1000 /µL) and **σ**= 59 (1000 /µL). Below is the sample distribution:



2.a Computation of the test statistic

In this case, we have:

- a large sample $(n > 100)$
- a known $\boldsymbol{\sigma}^2$ (of the reference population)
- the observed sample mean $\bar{x}$ and sample sd s.

So we can compute:

$$Z_{calc} = \frac{\bar{x} - \mu}{\frac{\sigma}{\sqrt{n}}}$$

- ✍🏻 Let’s do it “by hand” first to see the steps

```r
# General Population of reference
mu <- 236
sigma  <- 59
# Sample of HF patients
n <- 299
x_HF <- mean(heart_failure$plat_norm)         #    263.358
s_HF <- sd(heart_failure$plat_norm)           #    97.80424
# IF large sample & KNOWN pop variance
std_err_HF <- sigma /sqrt(n)                  # 3.412058
z_calc_HF <-  (x_HF - mu) / std_err_HF        # 8.018043
```

2.b Computation of the p-value associated to the test statistic

To find the **p-value** associated with a z-score in R, we can use the `pnorm()` function, which uses the following syntax:

- `q`: The z-score
- `mean`: The mean of the normal distribution. Default is `0`.
- `sd`: The standard deviation of the normal distribution. Default is 1.
- `lower.tail`:
  - If TRUE, the probability to the left of q in the normal distribution is returned
  - If FALSE, the probability to the right is returned. Default is TRUE.

```r
# Left-tailed test
p_value_l <- stats::pnorm(z_calc_HF, mean = 0, sd = 1, lower.tail = TRUE)
# Right-tailed test
p_value_r <- stats::pnorm(z_calc_HF, mean = 0, sd = 1,lower.tail = FALSE)
# Two-tailed test  (our case)
p_value_two <- 2*stats::pnorm(z_calc_HF, mean = 0, sd = 1, lower.tail = FALSE)
```

2.c Computation of the p-value associated to the test statistic

- 👩🏻‍💻 Let’s see how this could be done using an R function `BSDA::z.test`

```r
z_test_summary <- BSDA::z.test(x = heart_failure$plat_norm,
             alternative='two.sided',
             mu=236,
             sigma.x=59,
             conf.level=.95)
z_test_summary
```

```

    One-sample z-Test

data:  heart_failure$plat_norm
z = 8.018, p-value = 1.074e-15
alternative hypothesis: true mean is not equal to 236
95 percent confidence interval:
 256.6705 270.0455
sample estimates:
mean of x
  263.358
```

Same results!

3. Results and interpretation

1. Based on the critical region, the calculated test statistic `z_calc_HF = 8.0180` falls in the CRITICAL REGION (well beyond the critical point)

```r
# given
z_critical  <- c(-1.96, +1.96) # (Z score corresponding to α  = 0.05)
# Check
z_calc_HF > z_critical
```

```
[1] TRUE TRUE
```

2. Based on the p-value, `p_value_two = 1.07443e-15` is much much smaller than α

```r
# Check
p_value_two <  0.05
```

```
[1] TRUE
```

DECISION: we reject the Null Hypothesis (basically we conclude that it is extremely unlikely that the sample we drew could have occurred just by chance). So the test indicates that, indeed, there is a difference between heart failure patients and the general population in terms of average platelets count.

— EXAMPLE B —

(1 sample | n < 30 | t test)

Comparing sample mean to a hypothesized population mean (with t test)

Same question, but with a *smaller sample* to work on (this varies, but generally it means $n < 30$). Imagine the patients were only observed over a **follow-up period of 21 days**, and also let's assume we don't know the population's variance

Stating the hypothesis more formally:

What is the population Total Platelet Count (TPC) mean for all people who suffered of heart failure (μ_{HF21d}) in the past 21 days or less?

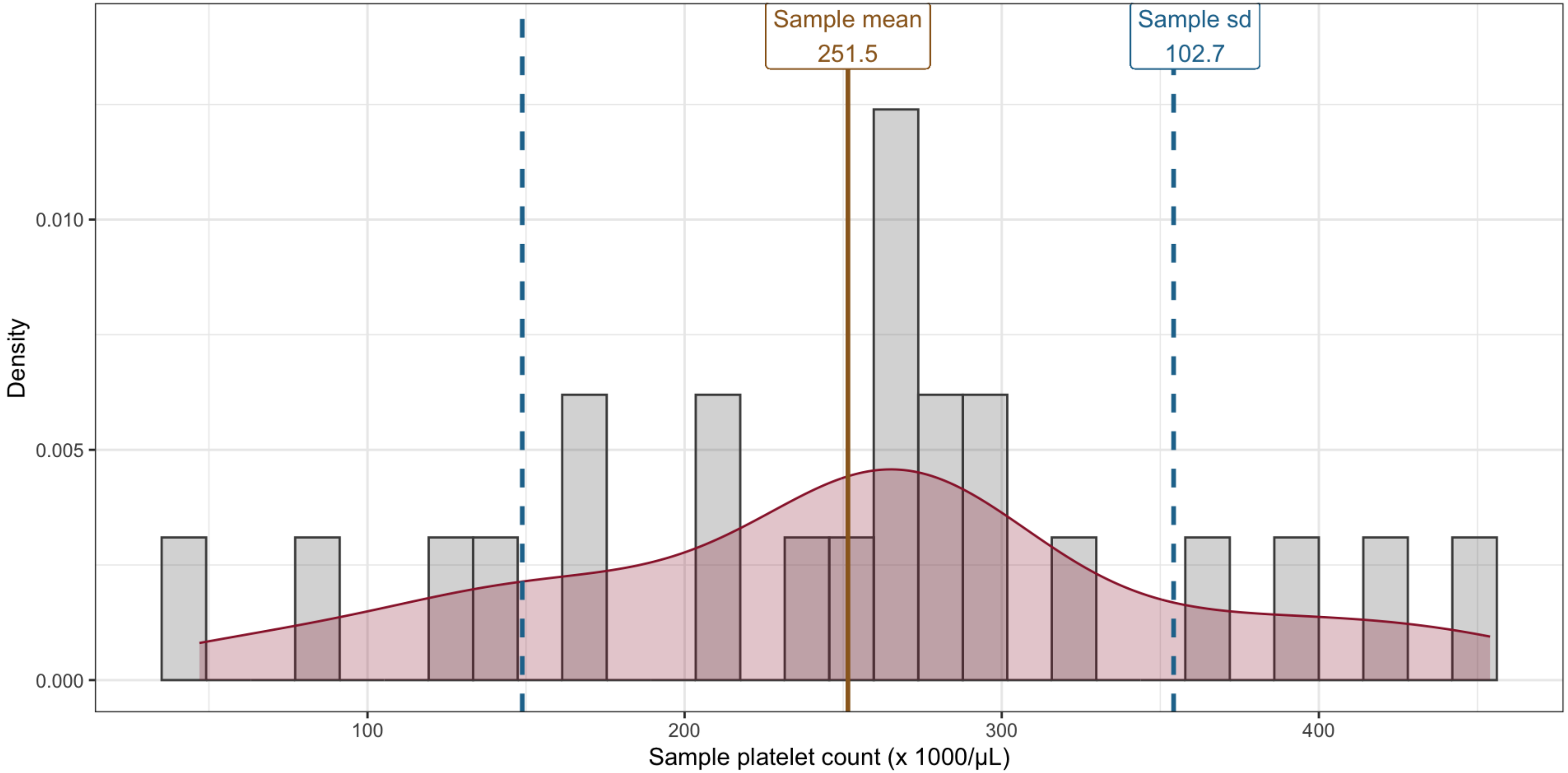
- H_0 : there is no difference in mean TPC between patients who suffered heart failure (visited in 21 days) and the general population
  - μ_{HF21d} = 236 -> hypothesis of no effect or ("no difference")
- H_a : there is a difference in mean TPC between patients who have suffered heart failure and the general population ("some effect"). This can be formalized as:
  - $\mu_{HF21d} \neq 236$ (two-sided test)

1. Question: How does the mean platelets count in the patients' sample compare against a r

```r
# normalize the var for readability
heart_21d  <-  heart_failure %>%  dplyr::mutate(plat_norm = platelets/1000) %>%
  filter(time <= 21)                                # 23 obs
# compute mean & sd for plot
mean_plat_p <- round(mean(heart_21d$plat_norm), digits = 1)
sd_plat_p <- round(sd(heart_21d$plat_norm), digits = 1)

heart_21d %>%
  ggplot(aes(x = plat_norm))+
  geom_histogram(aes(y = ..density..), bins=30, alpha=0.25, colour = "#4c4c4c") +
  geom_density(colour ="#9b2339", alpha=0.25, fill = "#9b2339") +
  # add mean vertical line
  geom_vline(xintercept = mean_plat_p, na.rm = FALSE,size = 1,color= "#9b6723") +
  # add also +/- 1sd
  geom_vline(aes(xintercept = mean_plat_p + sd_plat_p),
             color = "#23749b", size = 1, linetype = "dashed") +
  geom_vline(aes(xintercept = mean_plat_p - sd_plat_p),
             color = "#23749b", size = 1, linetype = "dashed") +
  # add annotations with the mean value
  geom_label(aes(x=mean_plat_p,  y=0.014, label=paste0("Sample mean\n",mean_plat_p)),
             color = "#9b6723") +
  geom_label(aes(x=361,  y=0.014, label=paste0("Sample sd\n",sd_plat_p)),
             color = "#23749b") +
  theme_bw() +  labs(y = "Density", x = "Sample platelet count (x 1000/µL)")
```

1. Question: How does the mean platelets count in the patients’ sample compare against a r

For a general population, the Total Platelet Count (TPL) has μ=236 (1000 /µL) and σ= 59 (1000 /µL). Below is the smaller sample distribution:



2.a Picking the suitable test

In this case, we have:

- a “small” sample $n = 23$
- an unknown $\boldsymbol{\sigma}^{\mathbf{2}}$ (of the reference population)
- We obtained the sample mean $\bar{x}$ and sample sd s.

So we can compute:

$$t_{calc} = \frac{\bar{x} - \mu}{\frac{s_{\bar{x}}}{\sqrt{n-1}}}$$

2.b Computation of the test statistic

- Option 1: Let’s compute the t test “by hand” ✍🏻

```r
# General Population of reference
mu_pop <- 236

# SAMPLE HF patients follow up less 21 days
heart_21d <- heart_failure %>% filter(time <= 21)

n_21d <- nrow(heart_21d)                            # 23
x_HF_21d <- mean(heart_21d$plat_norm)               # 251.5094
s_HF_21d <- sd(heart_21d$plat_norm)                 # 102.7341
df_HF_21d <- n_21d-1                                # 22

# IF SMALL sample UNKNOWN sigma
std_err_HF_21d <- s_HF_21d /sqrt(n_21d -1)        # 21.90298
t_calc <-  (x_HF_21d - mu_pop) / std_err_HF_21d   # 0.7080951
```

- Option 2: Let’s compute the t test with `stats::t.test` 👩🏻‍💻

```r
t_stat_HF_21d_v2 <- stats::t.test(x = heart_21d$plat_norm,
                                  mu = mu_pop,
                                  alternative = "two.sided")
# extract t_calc from results df
t_calc_v2  <- t_stat_HF_21d_v2[["statistic"]][["t"]] # 0.7240093
```

There is a small difference in the `t_calc ≠ t_calc_v2` due to the fact that I use the Bessel’s correction `n−1` in the

2.c Computation of the p-value associated to the test statistic

- Option 1: “by hand” ✍🏻

To find the **p-value** associated with a t-score in R, we can use the `pt(q, df, lower.tail = TRUE)` function, which uses the following syntax:

- `q`: The t-score
- `df`: The degrees of freedom
- `lower.tail`:
  - TRUE to calculate the probability to the left of q which is called as left-tailed test
  - FALSE as right-tailed test.

```r
# ---- Option 1
# -- Left-tailed test
#pt(t_stat_HF_21d, df_HF_21d, lower.tail = TRUE)

# -- Right-tailed test
#pt(t_stat_HF_21d, df_HF_21d, lower.tail = FALSE)

# -- Two-tailed test  (our case)
p_value_t_test <- 2*pt(t_calc, df_HF_21d, lower.tail = FALSE) # 0.4863214
```

- Option 2: from results of `stats::t.test` 👩🏻‍💻

```r
# ---- Option 2
# extract  p_value from results df
p_value_v2  <- t_stat_HF_21d_v2[["p.value"]] # 0.4766892
```

3. Results and interpretation

1. Based on the critical region, `t_calc ≃ 0.71` is smaller than the t critical value, i.e. it falls within the region of acceptance, so he null hypothesis is not rejected

```r
#find two-tailed t critical values

t_crit_two <- qt(p=.05/2, df=22, lower.tail=FALSE)    # 2.073873
# Compare t score against t critical
t_calc > t_crit_two  # FALSE
```

```
[1] FALSE
```

2. Based on the p-value, `p_value ≃ 0.48` is larger than α, i.e. the probability of observing a test statistic (assuming H_0 is true) is quite large

```r
# Check
p_value_t_test <  0.05  # FALSE
```

```
[1] FALSE
```

DECISION: we FAIL to reject H_0. So the test indicates that there is not a statistically significant difference between heart failure patients visited within 21 days and the general population in terms of average platelets count.

> **Note**
>
> What changed testing a sample with smaller n, instead of a large one?

— EXAMPLE C —

(2 samples | t test)

Comparing two independent sample means (t test)

This time, we investigate if there might be an actual difference in the Platelet Count means **between the patients who died and the patients who survived** heart failure.

Stating the above hypotheses more formally:

Is there a statistically significant difference between the mean values of two groups?

- H_0 : The two population means are equal
  - $\mu_1 = \mu_0 \iff \mu_1 - \mu_0 = 0$
- H_a : There is a mean difference between the two groups in the population. Possible directional difference formulation (two-tailed, left-tailed, right-tailed)
  - $\mu_1 \neq \mu_0 \iff \mu_1 - \mu_0 \neq 0$ (the two population means are not equal)
  - $\mu_1 < \mu_0 \iff \mu_1 - \mu_0 < 0$ (population 1 mean is less than population 0 mean)
  - $\mu_1 > \mu_0 \iff \mu_1 - \mu_0 > 0$ (population 1 mean is greater than population 0 mean)

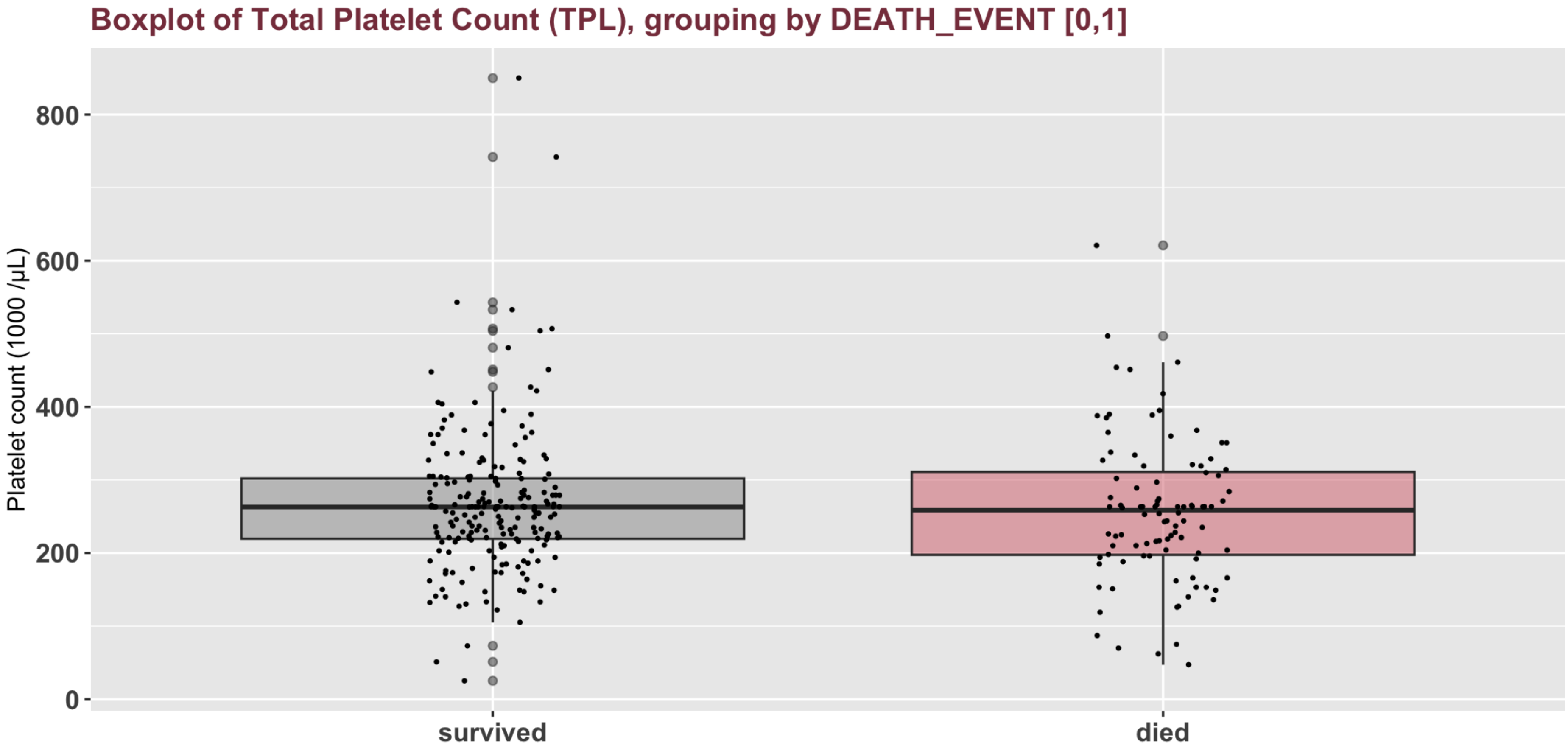
Comparing two independent sample means (t test) (cont.)

1. Question: Is there a statistically significant difference between the Platelet Counts in the patients

```r
# boxplot by group
heart_failure %>%
  ggplot(mapping = aes(y = plat_norm, x = DEATH_EVENT_f, fill = DEATH_EVENT_f)) +
  geom_boxplot(alpha=0.5)+
  #geom_violin(alpha=0.5) +
  geom_point(position = position_jitter(width = 0.1), size = 0.5)+
  scale_fill_manual(values = c("#999999", "#d8717b"))  +
  # drop legend and Y-axis title
  theme(plot.title = element_text(size = 14,face="bold", color = "#873c4a"),
        legend.position = "none",
        axis.text.x = element_text(size=12,face="bold"),
        axis.text.y = element_text(size=12,face="bold")) +
  labs(title = "Boxplot of Total Platelet Count (TPL), grouping by DEATH_EVENT [0,1]",
       x = "", y  = "Platelet count (1000 /µL)")
```

Comparing two independent sample means (t test) (cont.)

There seems to be no major difference in the two groups



2. Verify the assumptions for independent t-test

1. The 2 samples (“died” and “survived”) must be independent ✅
2. The dependent variable is scaled in intervals (Platelets Count in 10^3 “/µL”) ✅
3. The dependent variable is normally distributed (Platelets Count in 10^3 “/µL”) ✅

- (If not, use *non parametric* test)

4. The variance within the 2 groups should be similar ❓

- (If not, perform Welch’s t-test)

Preliminary Fisher’s F test to check for variance equality

- We can compute the Fisher test “by hand” ✍🏻

```r
## -- data by group
n_died <- nrow(heart_failure[heart_failure$DEATH_EVENT == 1 ,])
mean_died <- mean(heart_failure [ heart_failure$DEATH_EVENT == 1,  "plat_norm"])
sd_died <- sd(heart_failure [heart_failure$DEATH_EVENT == 1 ,  "plat_norm"])
var_died <- var(heart_failure [heart_failure$DEATH_EVENT == 1 ,  "plat_norm"])

n_survived <- nrow(heart_failure[heart_failure$DEATH_EVENT == 0, ])
mean_survived <- mean(heart_failure [ heart_failure$DEATH_EVENT == 0,  "plat_norm"])
sd_survived <- sd(heart_failure [heart_failure$DEATH_EVENT == 0 ,  "plat_norm"])
var_survived <- var(heart_failure [heart_failure$DEATH_EVENT == 0 ,  "plat_norm"])

## -- F TEST
F_ratio <- var_died / var_survived
F_ratio  # 1.020497
```

```
[1] 1.020497
```

Preliminary Fisher’s F test to check for variance equality (.cont)

```r
## -- Define the critical value of F distribution for a risk of alpha = 0.05
# qf(p=.05, df1 = n_died-1, df2 = n_survived-1, lower.tail = FALSE) # RIGHT-Tailed
# qf(0.95, df1 = n_died-1, df2 = n_survived-1, lower.tail = FALSE) # LEFT- Tailed
qf(c(0.025, 0.975), df1 = n_died-1, df2 = n_survived-1) # TWO-Tailed
```

```
[1] 0.6994659 1.3987233
```

```r
## --Compute the exact p-value (two-tailed )
p_value_f <- 2 * (1 - pf(F_ratio, df1 = (n_died-1), df2 = (n_survived-1)))
p_value_f
```

```
[1] 0.8914982
```

A test statistic (F) of 1.02 is obtained, with degrees of freedom 95 and 202.

The p-value is 0.89, greater than the p-value threshold of 0.05. This suggests **we can not reject the null hypothesis of equal variances**.

The variance within the 2 groups should be similar ✅ –> we can run a t-test.

3.a Computation of t test statistic

Since we verified the required assumptions, the test method is the independent (two-sample) t-test. In this case, we have:

- a large sample $(\mathbf{n}_1 + \mathbf{n}_2 > 100)$
- the population variance(s) are unknown, but we can assume = variances in 2 groups
- *standard error* of the means' difference is obtained as **pooled estimate standard deviation of the sampling distribution of the difference**

So we can compute: $t_{calc} = \frac{Difference\ Between\ Sample\ means}{Std.\ Err.\ of\ the\ difference} = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$

```r
# Step 1 - compute difference of sample means
mean_diff <- (mean_died - mean_survived) # -10.27645

# Step 2 - Compute associated t-statistics
# pooled std error
pooled_stderror <- sqrt(sd_died^2/(n_died ) + sd_survived^2/(n_survived ))
# pooled std error corrected
pooled_stderror_corr <- sqrt(sd_died^2/(n_died-1) + sd_survived^2/(n_survived-1))

###  t statistic
t_calc <- (mean_died - mean_survived) / pooled_stderror_corr
```

3.b Computation of the p-value associated to the t statistic

```r
# Step 3 - degrees of freedom
# n1 + n2 - number of estimated parameters (2 means)
d_f <- n_died + n_survived - 1 - 1 # 297

# Step 4 - Deduced p-value
p_value <- 2 * pt(t_calc, df = d_f) # 0.4009635
p_value
```

```
[1] 0.4009635
```

4. Results and interpretation

1. Looking at the confidence interval of the difference, the `sample mean_diff` is well inside the 95% CI of = population mean

```r
mean_diff
```

```
[1] -10.27645
```

```r
# CI of the means difference
CI_lower <- mean_diff + qt(.025, sum(n_died + n_survived) - 2) * pooled_stderror_corr
CI_lower
```

```
[1] -34.32074
```

```r
CI_upper <- mean_diff + qt(.975, sum(n_died + n_survived) - 2) * pooled_stderror_corr
CI_upper
```

```
[1] 13.76785
```

2. As for the p-value, `p_value = 0.40` is bigger than threshold probability α

```r
# Check
p_value
```

```
[1] 0.4009635
```

```r
p_value <  0.05  # FALSE
```

```
[1] FALSE
```

DECISION: So, we fail to reject the null hypothesis of equal populations means of TPC. So the test indicates that we do not have sufficient evidence to say that the mean counts of platelets in between these two populations is different.

— EXAMPLE D —

(3+ samples | ANOVA test)

Comparing sample means from 3 or more groups (ANOVA)

In this example, we adopt the ANOVA (“Analysis Of Variance”) test, i.e. an extension of the previous test, but examined how means of a variable differ across 3 or more groups. We will use **‘one- way’ ANOVA**, which serves when there is only one explanatory variable (“treatment”) with 3 or more levels, and only one level of treatment is applied for a given subject.

For this particular case, we use another realistic dataset showing the survival times of 33 laboratory mice with thymic leukemia who were randomly divided into 3 groups:

- 1st group received Treatment 1
- 2nd group received Treatment 2
- 3rd group as Control

```r
# load new dataset
mice <- readxl::read_excel(here::here("practice","data_input",
                                      "02_datasets","mice_exe_ANOVA.xlsx"))
```

1. Question: Is there a statistically significant difference between the mean values of the k

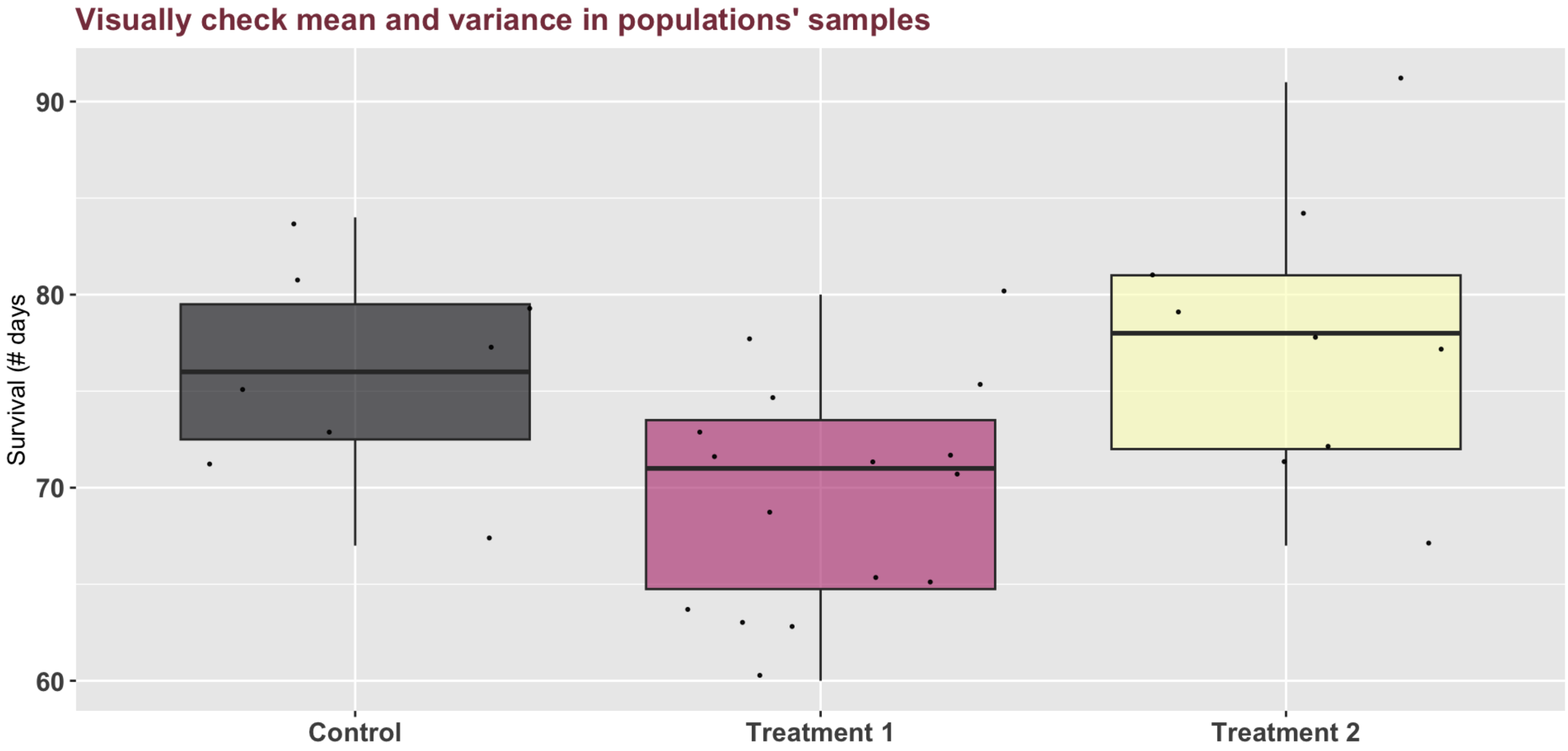
Defining the question formally:

- $H_0 : \mu_1 = \mu_2 = \mu_3$ all 3 population means are equal
- H_a : at least one of (μ_1, μ_2, μ_3) is not equal to the other means

```r
# boxplot by group
mice %>%
ggplot(., aes(x = group, y = surv_days, fill = group)) +
  geom_boxplot() +
  scale_fill_viridis(discrete = TRUE, alpha=0.6, option="A") +
  geom_jitter(color="black", size=0.4, alpha=0.9) +
  # theme_minimal() +
  # drop legend and Y-axis title
  theme(plot.title = element_text(size = 14,face="bold", color = "#873c4a"),
        axis.text.x = element_text(size=12,face="bold"),
        axis.text.y = element_text(size=12,face="bold"),
        legend.position = "none",
        ) +
  labs(title = "Visually check mean and variance in populations' samples" ) +
  ylab(label = "Survival (# days") + xlab(label = "")
```

1. Question: Is there a statistically significant difference between the mean values of the k p

The boxplot suggests that the 3 groups might have some fairly different distributions



2. Verify the assumptions for one-way ANOVA

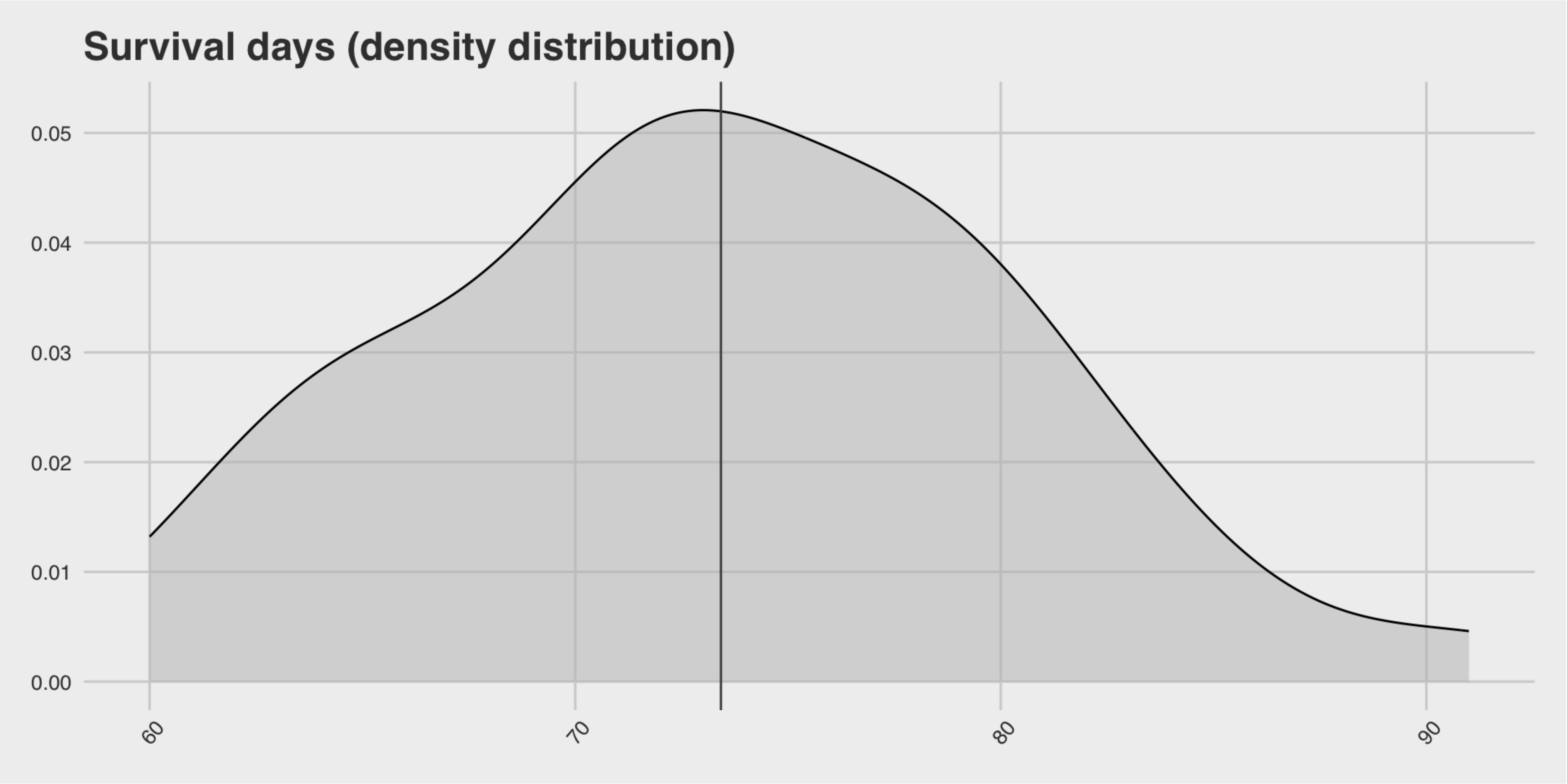
The dependent variable is on a metric scale. In the case of the analysis of variance, the independent variable (factor) has at least three levels.

Assumptions for the results of a one-way ANOVA to be valid:

1. **Independence of observations** – The observations in each group are independent of each other and the observations within groups were obtained by a random sample. ✅
2. **Normally-distributed response variable** – The values of the dependent variable follow a normal distribution. ❓
3. **Homogeneity of variance** – The variances of the populations that the samples come from are equal. ❓

Preliminary check for normality (visual)

2. **Normally-distributed response variable** ✅ • (confirmed by visual inspection)



Preliminary check for normality (test) with `stats::shapiro.test`

```r
# Shapiro-Wilk Normality Test to verify normality
# option 1
stats::shapiro.test(mice[mice$group == "Control", "surv_days", drop=TRUE])
```

```
	Shapiro-Wilk normality test

data:  mice[mice$group == "Control", "surv_days", drop = TRUE]
W = 0.99374, p-value = 0.9989
```

```r
stats::shapiro.test(mice[mice$group == "Treatment 1", "surv_days", drop=TRUE])
```

```
	Shapiro-Wilk normality test

data:  mice[mice$group == "Treatment 1", "surv_days", drop = TRUE]
W = 0.95716, p-value = 0.6106
```

```r
stats::shapiro.test(mice[mice$group == "Treatment 2", "surv_days", drop=TRUE])
```

```
	Shapiro-Wilk normality test

data:  mice[mice$group == "Treatment 2", "surv_days", drop = TRUE]
W = 0.97921, p-value = 0.9601
```

Preliminary check for normality (test) with `rstatix::shapiro_test`

(same thing, but using a different R function)

2. **Normally-distributed response variable** – ✅

- (confirmed by Shapiro-Wilk normality test)

[The null hypothesis of this test is H_0 **= "sample distribution is normal"**]

```r
# Shapiro-Wilk Normality Test to verify normality
# option 2 (all 3 groups at once)
mice %>%
  dplyr::group_by(group) %>%
  rstatix::shapiro_test(surv_days)
```

```
# A tibble: 3 × 4
  group       variable  statistic     p
  <chr>       <chr>         <dbl> <dbl>
1 Control     surv_days     0.994 0.999
2 Treatment 1 surv_days     0.957 0.611
3 Treatment 2 surv_days     0.979 0.960
```

Preliminary check variance equality

3. **Homogeneity of variance** – ✅

- (Besides visual inspection, confirmed by Levene test for variance equality)

[The null hypothesis H_0 **= several groups have the same variance** (possible variance differences occur only by chance, since there are small differences in each sampling)]

```r
# Levene test for variance equality
levene <- mice %>%                                # name of the data
  car::leveneTest(surv_days ~ as.factor(group),   # continuous DV ~   group IV
                  data = .,            # pipe the data from above
                  center = mean)       # default is median
levene
```

```
Levene's Test for Homogeneity of Variance (center = mean)
      Df F value Pr(>F)
group  2  0.1721 0.8427
      30
```

No evidence of violations of HOV were found, since the p-value for the Levene test (= 0.8427157) is greater than .05, then the variances are not significantly different from each other (i.e., the homogeneity assumption of the variance is met).

3 Computation of ANOVA F-ratio

ANOVA in R can be done in several ways.

Since it’s quite straightforward, let’s do all the steps by hand first. We need to obtain the needed “ingredients” to calculate the F-ratio:

$$\boldsymbol{F}_{calc} = \frac{Mean\ Square\ Between}{Mean,\ Square\ Within} = \frac{MSB}{MSW} = \frac{\frac{SSB}{df1}}{\frac{SSW}{df2}}$$

3.a Computation of ANOVA F-ratio (“by hand”)

- Option 1: Let’s compute the ANOVA test “by hand” ✍🏻

```r
# Summary statistics
mice_calc <- mice %>%
  dplyr::mutate(mean_all = mean(surv_days),
         sd_all = sd (surv_days),
         dfw = 33-3, # df1 = n-k
         dfb = 3-1, # df2 = K-1
         group_f = as.factor(group)
         ) %>%
  dplyr::group_by(group) %>%
  dplyr::mutate(n_group = n(),
         mean_group = mean(surv_days),
         sd_group = sd (surv_days)) %>%
  ungroup() %>%
  mutate (ST = (surv_days - mean_all)^2,
          SW = (surv_days - mean_group)^2,
          SB = (mean_group - mean_all)^2)

# Sum of Squares
SST <- sum(mice_calc$ST)
SSB <- sum(mice_calc$SB)
SSW <- sum(mice_calc$SW)
dfw <- 33-3  # df2
dfb <- 3-1 # df1

# calculated F statistic
F_calc <- (SSB/dfb)/(SSW/dfw) # 5.65
# F critical value
F_crit <- qf(p = 0.01, df1 = 2, df2 = 30, lower.tail = FALSE) # 5.390346
```

3.b Computation of ANOVA F-ratio (with R functions)

That was just to show how to build it step-by-step (🤓), but we don't have to! We have alternative R functions that can do ANOVA for us:

- Option 2: With the `stats::aov` followed by the command `summary` 👩🏻‍💻

```r
aov_1 <- stats::aov(surv_days ~ group_f,
                    data = mice_calc)
summary(aov_1)
```

```
            Df Sum Sq Mean Sq F value  Pr(>F)   
group_f      2  434.6  217.32   5.652 0.00826 **
Residuals   30 1153.4   38.45                   
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Option 3: With the `stats::oneway.test()` function 👩🏻‍💻

```r
aov_2 <- stats::oneway.test(surv_days ~ group_f,
            data = mice_calc,
            # assuming equal variances
            var.equal = TRUE)
aov_2
```

```

	One-way analysis of means

data:  surv_days and group_f
F = 5.6522, num df = 2, denom df = 30, p-value = 0.008258
```

4. Results and interpretation

All 3 options have given the same results, i.e., `F-ratio = 5.652` and a `p-value = 0.00826`

DECISION: Given that the p-value is smaller than 0.05, we reject the null hypothesis, so we reject the hypothesis that all means are equal. Therefore, we can conclude that *at least one* group is different than the others in mean number of survival days.

> **Note**
>
> Have you seen the kind of notation `Pr(>F) 0.00826 **` before (as in the output of the `stats::aov` function)?

A CLOSER LOOK AT TESTING ASSUMPTIONS

— EXAMPLE E —

Testing two groups that are *not* independent

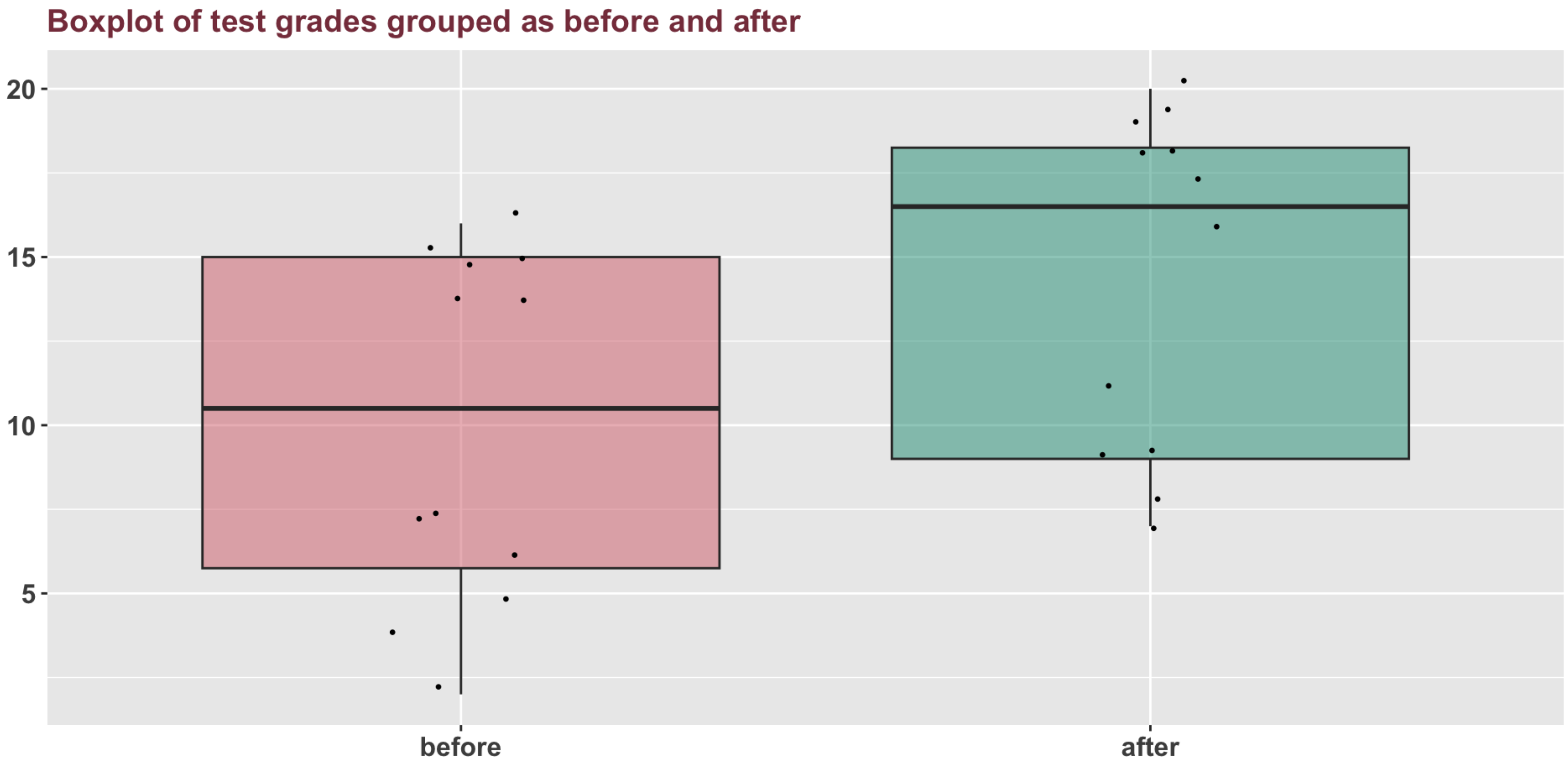
Let’s introduce another toy dataset just for demonstration purposes: imagine a statistics test is administered to the *same* group of 12 students **before and after** attending a workshop 😉.

```r
# toy dataset for paired groups
grades <- data.frame(
  before = c(16, 5, 15, 2, 14, 15, 4, 7, 15, 6, 7, 14),
  after = c(19, 18, 9, 17, 8, 7, 16, 19, 20, 9, 11, 18)
)
```

We may reshape the dataframe into the long form using `tidyr::pivot_longer` (for plotting)

```r
# reshape into long form
grades_long <- grades %>%
  dplyr::mutate(id = row_number()) %>%
  tidyr::pivot_longer(cols = before:after,
                      names_to = "time",
                      values_to = "grade") %>%
  dplyr::group_by(id) %>%
  # recode time as factor
  dplyr::mutate(time_f = as_factor(time ))  %>%
  # reorder time_  levels
  dplyr::mutate(time_f =  fct_relevel(time_f, "after", after =  1))
```

1. Question: Is the difference between two PAIRED samples statistically significant?



What a successful workshop! 😁

2 Hypotehsis for the PAIRED t-test for dependent samples

In this example, it is clear that the two samples are not independent since the same 12 students took the test before and after the workshop.

Given that the normality assumption is NOT violated (and given the small sample size), we use the **paired t-test**, with the following hypotheses:

- $\boldsymbol{H_0}$: mean grades before and after the workshop are equal
- $\boldsymbol{H_a}$: mean grades before and after the workshop are different

2 Computation of the PAIRED t-test for dependent samples

```r
t_stat_paired <- stats::t.test(x = grades$before,
                               y = grades$after,
                               mu = 0,
                               alternative = "two.sided",
                               paired = TRUE
)
t_stat_paired
```

```

	Paired t-test

data:  grades$before and grades$after
t = -1.8777, df = 11, p-value = 0.08718
alternative hypothesis: true mean difference is not equal to 0
95 percent confidence interval:
 -9.2317713  0.7317713
sample estimates:
mean difference 
          -4.25 
```

```r
# extract t_calc from results df
t_calc_pair   <- t_stat_paired[["statistic"]][["t"]] # -1.877683
p_value_pair   <- t_stat_paired[["p.value"]] # -1.877683
```

3. Results and interpretation

We obtain the test statistic, the p-value and a reminder of the hypothesis tested.

The calculated **t value** is -1.8776829 The **p-value** is 0.087177. Therefore, at the 5% significance level, **we do not reject the null hypothesis** that the statistics’ grades are similar before and after the workshop (😭).

Bonus function!

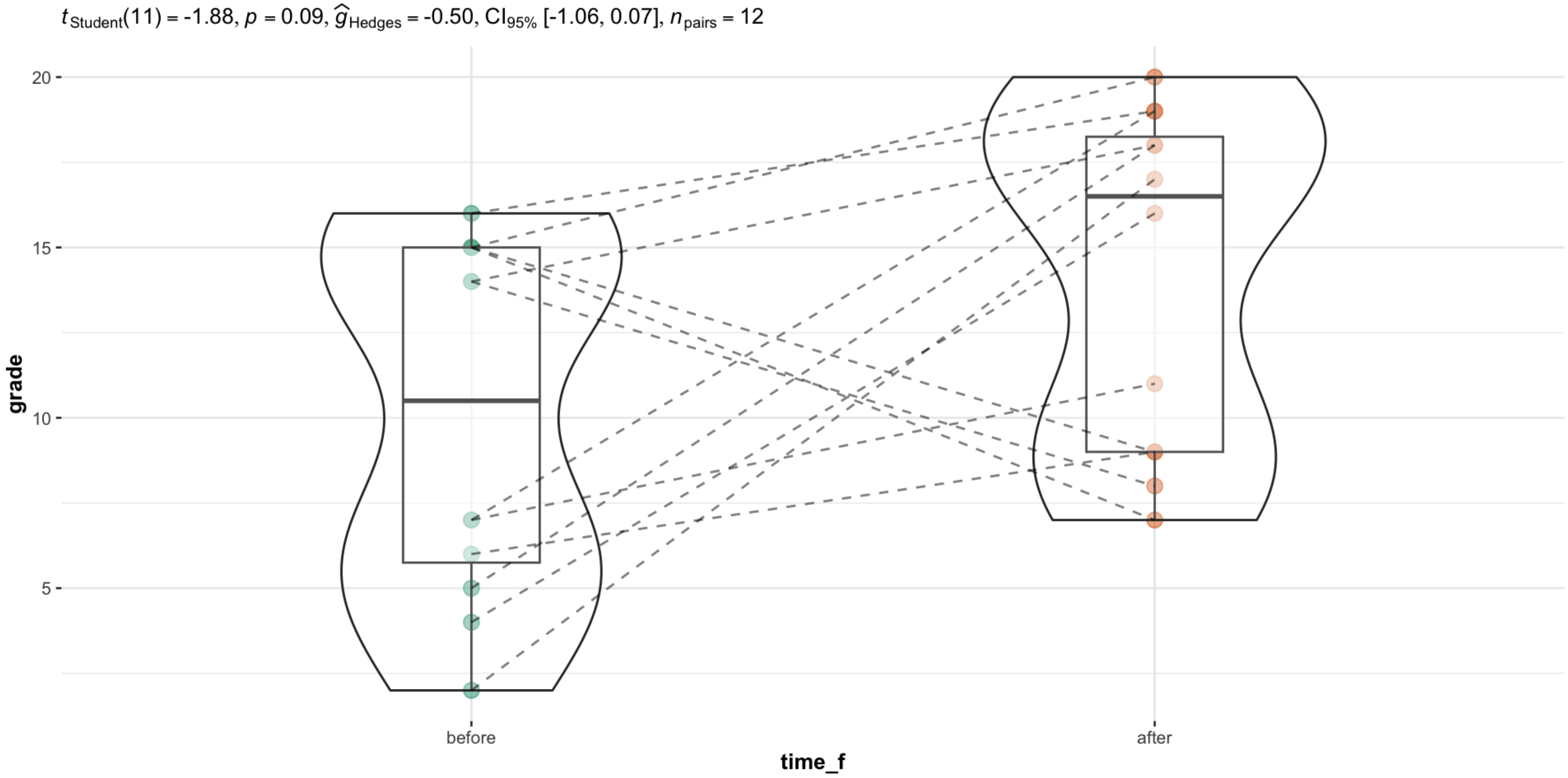
It is worth mentioning the `ggstatsplot` package, which combines plots representing the distribution for each group—and the results of the statistical test displayed in the subtitle of the plot.

Below we check out the `ggwithinstats()` function for *paired samples*.

```r
# load package
library(ggstatsplot)

# plot with statistical results
grades_long %>%
  # must ungroup the dataframe or it will give an error
  ungroup () %>%
  ggstatsplot::ggwithinstats(.,
                             x = time_f ,
                             y = grade ,
                             type = "parametric", # for wilcoxon
                             centrality.plotting = FALSE # remove median
  )
```

Bonus function!

The test results are rendered with the plot!



— EXAMPLE F —

(2 samples no normal | Wilcoxon Rank Sum Test)

Testing samples *without* normality assumption

Let's go back to the HEART FAILURE dataset but looking at the levels of **Creatinine Phosphokinase (CPK)** in the blood, an enzyme that might indicate a heart failure or injury

1. Question: Is there a statistically significant difference between CPK levels in the blood of

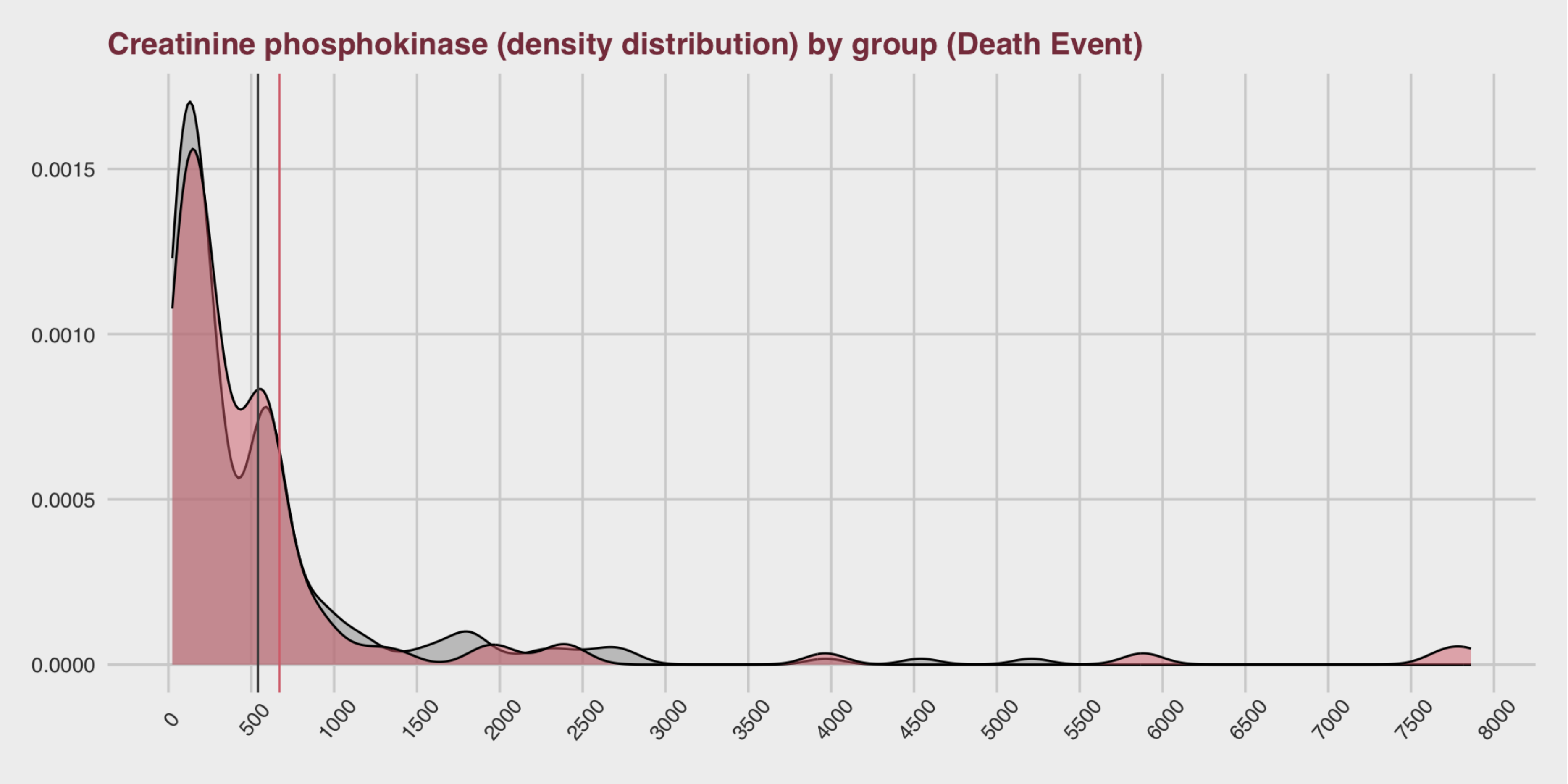
Defining the question formally:

- $H_0 : \mu_{CPK-died} = \mu_{CPK-surv}$ there is no difference in mean CPK between patients who suffered heart failure and died versus patients who survived after heart failure
- $H_a : \mu_{CPK-died} \neq \mu_{CPK-surv}$ there is a difference in mean CPK between patients who suffered heart failure and died versus patients who survived after heart failure (two-sided test)

```r
ggplot(heart_failure,aes(x = creatinine_phosphokinase,fill = DEATH_EVENT_f))+
  geom_density(alpha = 0.5)+theme_fivethirtyeight()+
  scale_fill_manual(values = c("#999999", "#d8717b"))+
  guides(fill = "none") +
  scale_x_continuous(breaks = seq(0,8000, 500))+
  geom_vline(aes(xintercept = mean(creatinine_phosphokinase[DEATH_EVENT == 0])),
             color = "#4c4c4c")+
  geom_vline(aes(xintercept = mean(creatinine_phosphokinase[DEATH_EVENT==1])),
             color = "#d8717b")+
  theme_fivethirtyeight()+
  theme(axis.text.x = element_text(angle=50, vjust=0.75))+
  labs(title =  "Creatinine phosphokinase (density distribution) by group (Death Event)") +
  theme(plot.title = element_text(size = 14,face="bold", color = "#873c4a"))
```

1. Question: Is there a statistically significant difference between CPK levels in the blood of

The density plot suggests non normality of the variable distribution



Preliminary check for normality (visual)

- **Normally-distributed response variable** - ❌

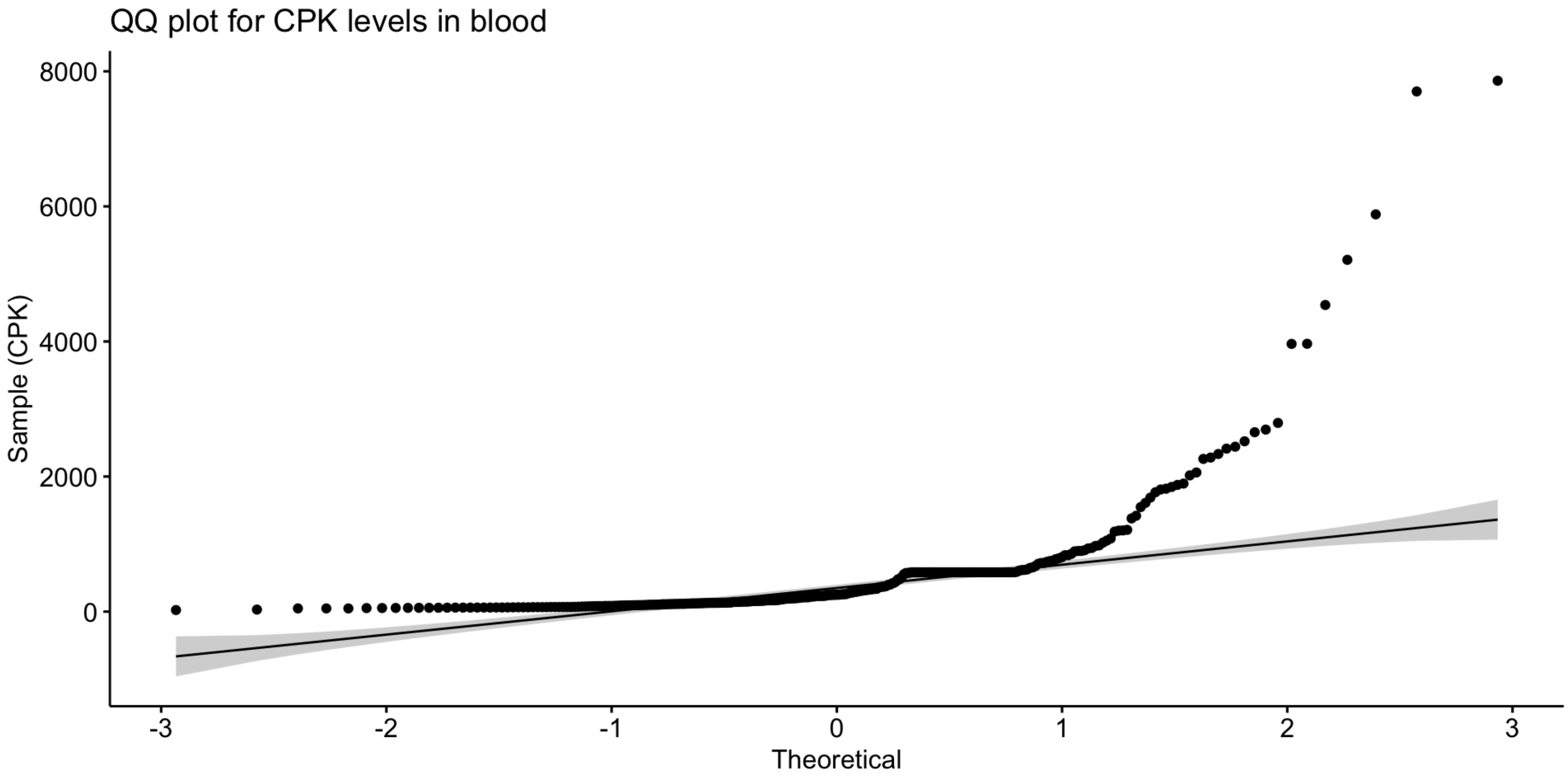
QQ plot (or quantile-quantile plot) draws the correlation between a given sample and the normal distribution. A 45-degree reference line is also plotted. In a QQ plot, each observation is plotted as a single dot.

- If the data are normal, the dots should form a straight line.

```r
# visual verification with QQ plot
ggpubr::ggqqplot(
  heart_failure$creatinine_phosphokinase,
  title = "QQ plot for CPK levels in blood",
  xlab ="Theoretical", ylab = "Sample (CPK)")
```

Preliminary check for normality (visual)

In a QQ plot, if the data are normal, the dots should follow a straight line.



Preliminary check for normality (test) with `rstatix::shapiro_test`

(same thing, but using a different R function)

- **Normally-distributed response variable** - ❌
  - (NOT normality confirmed by Shapiro-Wilk normality test)

[The null hypothesis of this test is H_0 = **"sample distribution(s) is/are normal"**]

Given the p-value we reject the null hypothesis

```r
# Shapiro-Wilk Normality Test to verify normality
heart_failure %>%
  dplyr::group_by(DEATH_EVENT_f) %>%
  rstatix::shapiro_test(creatinine_phosphokinase)
```

```
# A tibble: 2 × 4
  DEATH_EVENT_f variable                 statistic        p
  <fct>         <chr>                        <dbl>    <dbl>
1 survived      creatinine_phosphokinase     0.628 8.51e-21
2 died          creatinine_phosphokinase     0.439 1.99e-17
```

3. Computation of the Wilcoxon Rank Sum test statistic

The **Wilcoxon Rank Sum test** is considered to be the nonparametric equivalent to the **two-sample independent t-test**

Its ASSUMPTIONS are:

- Ordinal or Continuous dependent variable: e.g. CPK levels ✅
- Independence: All of the observations from both groups are independent of each other ✅
- Shape: The shapes of the distributions for the two groups are roughly the same ✅

```r
wrs_res <- wilcox.test(creatinine_phosphokinase ~ DEATH_EVENT, # immagino 0, 1
                   data = heart_failure ,
                   exact = FALSE,
                   alternative = "two.sided" )
wrs_res
```

```
	Wilcoxon rank sum test with continuity correction

data:  creatinine_phosphokinase by DEATH_EVENT
W = 9460, p-value = 0.684
alternative hypothesis: true location shift is not equal to 0
```

The **Wilcoxon Rank Sum test** is equivalent to the **Mann-Whitney U test** to compare two independent samples. Different

4. Results and interpretation

RESULTS: since the test statistic is `W = 9460` and the corresponding `p-value is 0.684 > 0.05`, we fail to reject the null hypothesis.

INTERPRETATION: We do not have sufficient evidence to say that CPK levels for dead patients is different than that of survived patients $\boldsymbol{\mu}_{CPK-died} \neq \boldsymbol{\mu}_{CPK-surv}$ at some statistically significant level)

— EXAMPLE G —

(2 samples no HOV | t test with the Welch correction)

Testing samples *without* homogeneous variance of observations assumption

1. Question: Is there a statistically significant difference between serum sodium levels in th

Defining the question formally:

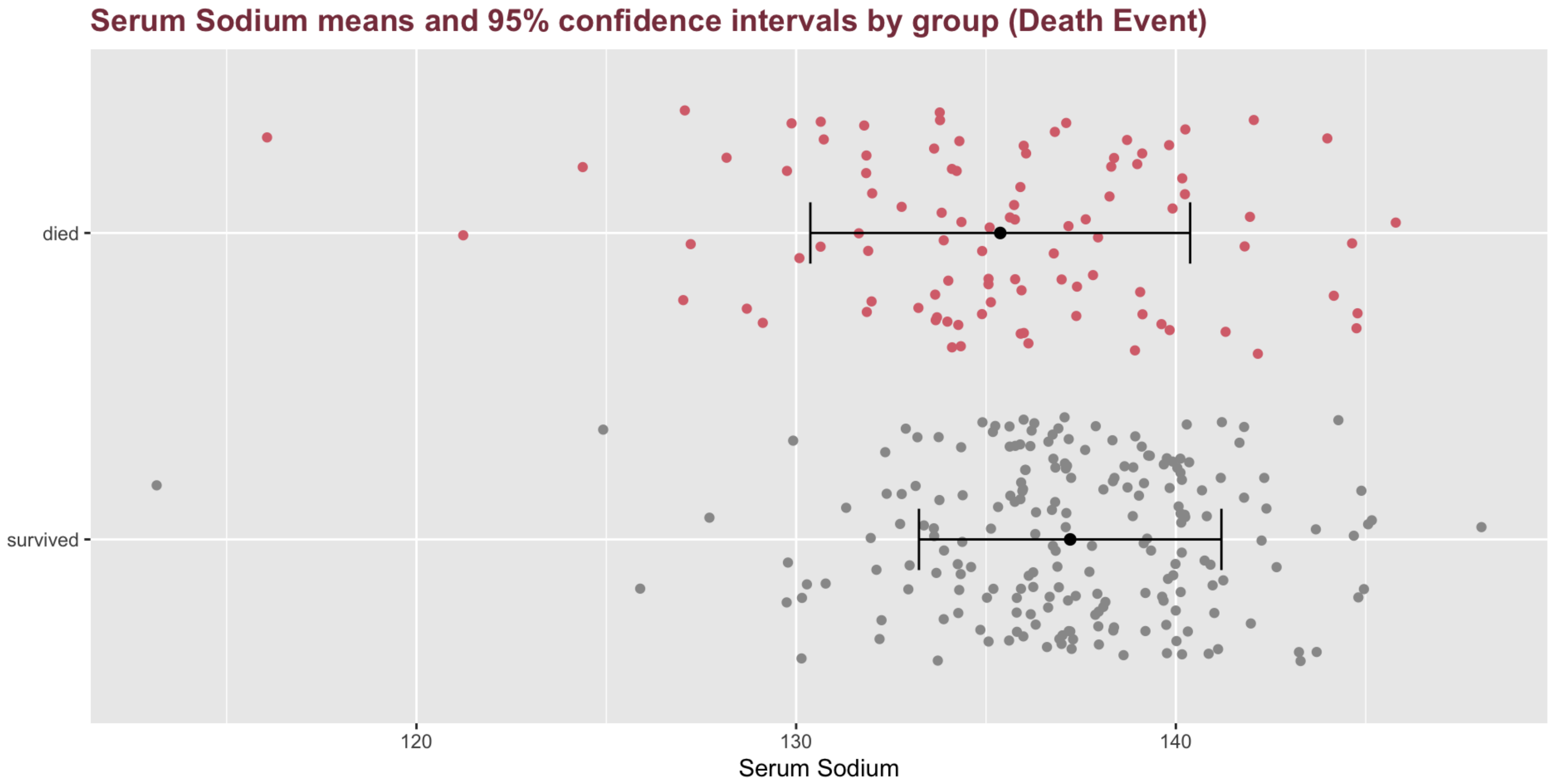
- $\boldsymbol{H_0} : \boldsymbol{\mu}_{sersod-died} = \boldsymbol{\mu}_{sersod-surv}$ there is no difference in mean serum sodium between patients who suffered heart failure and died versus patients who survived after heart failure
- $\boldsymbol{H_a} : \boldsymbol{\mu}_{sersod-died} \neq \boldsymbol{\mu}_{sersod-surv}$ there is a difference in mean serum sodium between patients who suffered heart failure and died versus patients who survived after heart failure (two-sided test)

Preliminary check “HOV” assumption (visual)

- **Homogeneity of Variance assumption** - ❌ Plotting the data offers some graphical intuition that the variance of observations in the two groups seem not homogenous

```r
#Compute means and 95% confidence intervals
swstats <- heart_failure %>%
  group_by(DEATH_EVENT_f) %>%
  summarise(count = n(),
    mean = mean(serum_sodium,na.rm=TRUE),
    stddev = sd(serum_sodium, na.rm=TRUE),
    meansd_l = mean - stddev,
    meansd_u = mean + stddev)

#The complete script with some styling added
ggplot(swstats, aes(x=DEATH_EVENT_f, y=mean)) +
  geom_point(colour = "black" , size = 2) +
  #Now plotting the individual data points before the mean values
  geom_point(data=heart_failure, aes(x=DEATH_EVENT_f, y=serum_sodium, colour = DEATH_EVENT_f),
             position = position_jitter() ) +
  scale_colour_manual(values = c("#999999","#d8717b") ) +
  #Add the error bars
  geom_errorbar(aes(ymin = meansd_l, ymax = meansd_u), width=0.2, color = "black") +
  labs(title = "Mean (-/+SD) serum sodium (mEq/L) by group", x = "", y = "Serum Sodium") +
  guides(fill = "none")  +
  coord_flip() +
  labs(title =  "Serum Sodium means and 95% confidence intervals by group (Death Event)") +
  theme(legend.position="none",plot.title = element_text(size = 14,face="bold", color = "#873c4a"))
```

Preliminary check “HOV” assumption (visual)



Preliminary check “HOV” assumption (test)

It is always best to use an actual test, so we use also the **Fisher’s F test** to verify equal variances of Serum Sodium concentration in the two groups. [In this test **H_0 = “the ratio of variances is equal to 1”**]

```r
f_test_res <- stats::var.test(heart_failure$serum_sodium[heart_failure$DEATH_EVENT == 1] ,
                              heart_failure$serum_sodium[heart_failure$DEATH_EVENT == 0])
f_test_res
```

```
	F test to compare two variances

data:  heart_failure$serum_sodium[heart_failure$DEATH_EVENT == 1] and
heart_failure$serum_sodium[heart_failure$DEATH_EVENT == 0]
F = 1.5769, num df = 95, denom df = 202, p-value = 0.007646
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 1.127401 2.254466
sample estimates:
ratio of variances
          1.576922
```

Given the `p−value = 0.007646` (smaller than α) we reject the null hypothesis, hence the HOV assumption for the t test does not hold.

2 Computation of the t test with the Welch correction

We can still run the **t test but with Welch correction**, i.e. the unequal variance condition is compensated by lowering the df. In fact the documentation (`?t.test`), reads:

- If `var.equal = TRUE`, then the pooled variance is used to estimate the variance
- Otherwise (`var.equal = FALSE`), the Welch approximation to the degrees of freedom is used.

```r
# With Welch correction (on by default) Unequal variance is compensated by lowering df
t_test_w <- t.test(heart_failure$serum_sodium[heart_failure$DEATH_EVENT == 1],
                   heart_failure$serum_sodium[heart_failure$DEATH_EVENT == 0],
                   # here we specify the situation
                   var.equal = FALSE,
                   paired = FALSE, alternative = "two.sided")

t_test_w
```

```
    Welch Two Sample t-test

data:  heart_failure$serum_sodium[heart_failure$DEATH_EVENT == 1] and
heart_failure$serum_sodium[heart_failure$DEATH_EVENT == 0]
t = -3.1645, df = 154.01, p-value = 0.001872
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -2.9914879 -0.6920096
sample estimates:
mean of x mean of y
 135.3750  137.2167
```

3. Results and interpretation

RESULTS: since the test statistic is `t = -3.1645 (with df = 154.01)` and the corresponding `p-value is 0.001872 < 0.05`, we reject the null hypothesis.

INTERPRETATION: We therefore have sufficient evidence to say that the level of serum sodium levels for dead patients is significantly different than that of survived patients $\boldsymbol{\mu}_{sersod-died} \neq \boldsymbol{\mu}_{sersod-surv}$

Final thoughts/recommendations

- There are often **many ways to do the same thing in R** (which is both a blessing and a curse in open source software). *Which should you choose?* It depends on the situation, but you may want to consider:
  - how recent/popular/well maintained is a `{package}` (this affects its stability)
  - the more a function abstracts away complexity, the easier it is to use interactively, but the harder it gets to handle inside your own custom functions
  - different function outputs may be more/less suitable for your analysis/publication requirements (check out your peers' choices!)
  - *(Always **read the documentation** to assess all of the above)*
- With easy equations, breaking them down "by hand" (at least once!) can really help you understand them
- It may seem a lot of work to write R code the first time 🥵 (e.g. for a publication-ready plot), but the good news is **once you wrote a script, you will be able to easily re-use it in many more instances** 🙌🏻 😃
- **Sample size n has a very powerful impact** on classical hypothesis testing results! More on this later...