Data Science for Biological, Medical and Health Research: Notes for 431

Thomas E. Love, Ph.D.

Version: 2017-08-13

Contents

Introduction

These Notes provide a series of examples using R to work through issues that are likely to come up in PQHS/CRSP/MPHP 431.

While these Notes share some of the features of a textbook, they are neither comprehensive nor completely original. The main purpose is to give 431 students a set of common materials on which to draw during the course. In class, we will sometimes:

- reiterate points made in this document,
- amplify what is here,
- simplify the presentation of things done here,
- use new examples to show some of the same techniques,
- refer to issues not mentioned in this document

but what we don't do is follow these notes very precisely. We assume instead that you will read the materials and try to learn from them, just as you will attend classes and try to learn from them. We welcome feedback of all kinds on this document or anything else. Just email us at 431-help at case dot edu, or submit a pull request.

What you will mostly find are brief explanations of a key idea or summary, accompanied (most of the time) by R code and a demonstration of the results of applying that code.

Everything you see here is available to you as HTML or PDF. You will also have access to the R Markdown files, which contain the code which generates everything in the document, including all of the R results. We will demonstrate the use of R Markdown (this document is generated with the additional help of an R package called `bookdown`) and R Studio (the "program" which we use to interface with the R language) in class.

Structure

The Notes, like the 431 course, are split into three main parts.

Part A is about **visualizing data and exploratory data analyses**. These Notes focus on using R to work through issues that arise in the process of exploring data, managing (cleaning and manipulating) data into a tidy format to facilitate useful work downstream, and describing those data effectively with visualizations, numerical summaries, and some simple models.

Part B is about **making comparisons** with data. The Notes discuss the use of R to address comparisons of means and of rates/proportions, primarily. The main ideas include confidence intervals, the bootstrap and parametric and non-parametric tests of hypotheses. Key ideas from Part A that have an impact here include visualizations to check the assumptions behind our inferences, and cleaning/manipulating data to facilitate our comparisons.

Part C is about **building models** with data. The Notes are primarily concerned (in 431) with linear regression models for continuous quantitative outcomes, using one or more predictors. We'll see how to use models to accomplish many of the comparisons discussed in Part B, and make heavy use of visualization and data management tools developed in Part A to assess our models.

Course Philosophy

In developing this course, we adopt a modern approach that places data at the center of our work. Our goal is to teach you how to do truly reproducible research with modern tools. We want you to be able to answer real questions using data and equip you with the tools you need in order to answer those questions well (Cetinkaya-Rundel (2017) has more on a related teaching philosophy.)

The curriculum includes more on several topics than you might expect from a standard graduate introduction to statistics.

- data gathering
- data wrangling
- exploratory data analysis and visualization
- multivariate modeling
- communication

It also nearly completely avoids formalism and is extremely applied - this is most definitely **not** a course in theoretical or mathematical statistics.

The 431 course is about **getting things done**. It's not a statistics course, nor is it a computer science course. It is instead a course in **data science**.

Chapter 1

Data Science

The definition of **data science** can be a little slippery. One current view of data science, is exemplified by Steven Geringer's 2014 Venn diagram.

- The field encompasses ideas from mathematics and statistics and from computer science, but with a heavy reliance on subject-matter knowledge. In our case, this includes clinical, health-related, medical or biological knowledge.
- As Gelman and Nolan (2017) suggest, the experience and intuition necessary for good statistical practice are hard to obtain, and teaching data science provides an excellent opportunity to reinforce statistical thinking skills across the full cycle of a data analysis project.
- The principal form in which computer science (coding/programming) play a role in this course is to provide a form of communication. You'll need to learn how to express your ideas not just orally and in writing, but also through your code.

1.1 Why a unicorn?

Data Science is a **team** activity. Everyone working in data science brings some part of the necessary skillset, but no one person can cover all three areas alone for excellent projects.

> [The individual who is truly expert in all three key areas (mathematics/statistics, computer science and subject-matter knowledge) is] a mythical beast with magical powers who's rumored to exist but is never actually seen in the wild.
>
> http://www.kdnuggets.com/2016/10/battle-data-science-venn-diagrams.html

1.2 Data Science Project Cycle

A typical data science project can be modeled as follows, which comes from the introduction to the amazing book **R for Data Science**, by Garrett Grolemund and Hadley Wickham, which is a key text for this course (Grolemund and Wickham 2017).

This diagram is sometimes referred to as the Krebs Cycle of Data Science. For more on the steps of a data science project, we encourage you to read the Introduction of Grolemund and Wickham (2017).

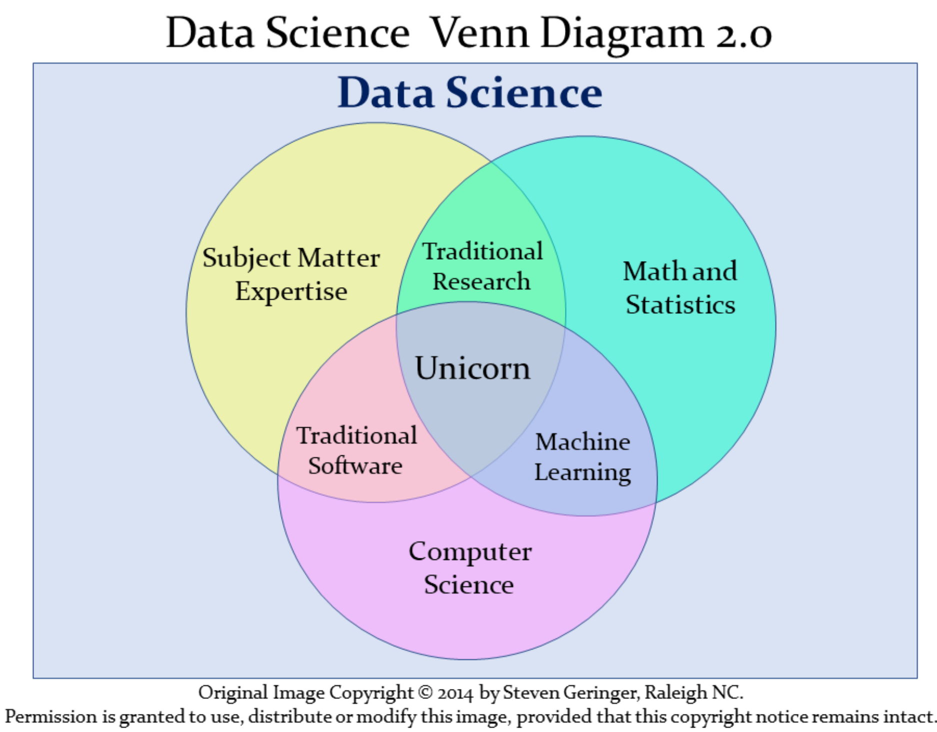


Figure 1.1: Data Science Venn Diagram from Steven Geringer

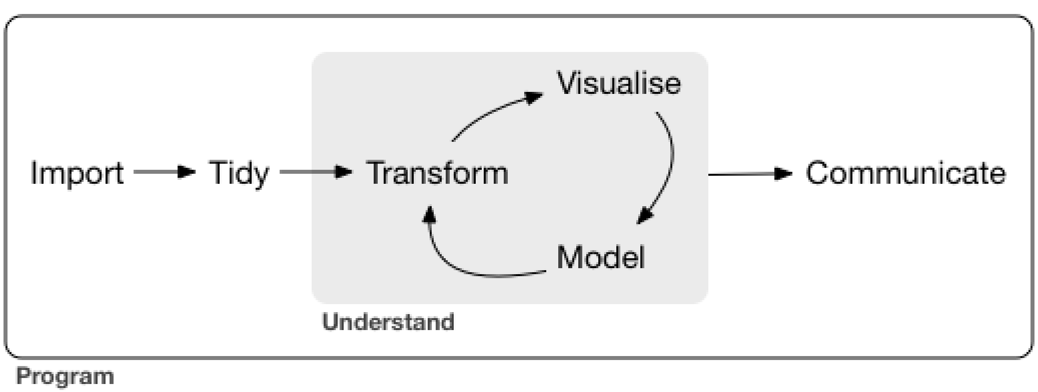


Figure 1.2: Source: R for Data Science: Introduction

1.3 What Will We Discuss in 431?

We'll discuss each of these elements in the 431 course, focusing at the start on understanding our data through transformation, modeling and (especially in the early stages) visualization. In 431, we learn how to get things done.

- We get people working with R and R Studio and R Markdown, even if they are completely new to coding. A gentle introduction is provided at Ismay and Kim (2017)
- We learn how to use the `tidyverse` (http://www.tidyverse.org/), an array of tools in R (mostly developed by Hadley Wickham and his colleagues at R Studio) which share an underlying philosophy to make data science faster, easier, more reproducible and more fun. A critical text for understanding the tidyverse is Grolemund and Wickham (2017). Tidyverse tools facilitate:
  - **importing** data into R, which can be the source of intense pain for some things, but is really quite easy 95% of the time with the right tool.
  - **tidying** data, that is, storing it in a format that includes one row per observation and one column per variable. This is harder, and more important, than you might think.
  - **transforming** data, perhaps by identifying specific subgroups of interest, creating new variables based on existing ones, or calculating summaries.
  - **visualizing** data to generate actual knowledge and identify questions about the data - this is an area where R really shines, and we'll start with it in class.
  - **modeling** data, taking the approach that modeling is complementary to visualization, and allows us to answer questions that visualization helps us identify.
  - and last, but definitely not least, **communicating** results, models and visualizations to others, in a way that is reproducible and effective.
- Some programming/coding is an inevitable requirement to accomplish all of these aims. If you are leery of coding, you'll need to get past that, with the help of this course and our stellar teaching assistants. Getting started is always the most challenging part, but our experience is that most of the pain of developing these new skills evaporates by early October.
- Having completed some fundamental work in Part A of the course, we then learn how to use a variety of R packages and statistical methods to accomplish specific inferential tasks (in Part B, mostly) and modeling tasks (in Part C, mostly.)

Chapter 2

Setting Up R

These Notes make extensive use of

- the statistical software language R, and
- the development environment R Studio

both of which are free, and you'll need to install them on your machine. Instructions for doing so are in found in the course syllabus.

If you need an even gentler introduction, or if you're just new to R and RStudio and need to learn about them, we encourage you to take a look at http://moderndive.com/, which provides an introduction to statistical and data sciences via R at Ismay and Kim (2017).

2.1 R Markdown

These notes were written using R Markdown. R Markdown, like R and R Studio, is free and open source.

R Markdown is described as an *authoring framework* for data science, which lets you

- save and execute R code
- generate high-quality reports that can be shared with an audience

This description comes from http://rmarkdown.rstudio.com/lesson-1.html which you can visit to get an overview and quick tour of what's possible with R Markdown.

Another excellent resource to learn more about R Markdown tools is the Communicate section (especially the R Markdown chapter) of Grolemund and Wickham (2017).

2.2 R Packages

To start, I'll present a series of commands I run at the beginning of these Notes. These particular commands set up the output so it will look nice as either an HTML or PDF file, and also set up R to use several packages (libraries) of functions that expand its capabilities. A chunk of code like this will occur near the top of any R Markdown work.

```
knitr::opts_chunk$set(comment = NA)

library(boot); library(devtools); library(forcats)
library(grid); library(knitr); library(pander)
```

```
library(pwr); library(viridis); library(NHANES)
library(tidyverse)
```

I have deliberately set up this list of loaded packages/libraries to be relatively small, and will add some other packages later, as needed. You only need to install a package once, but you need to reload it every time you start a new session.

2.3 Other Packages

I will also make use of functions in the following packages/libraries, but when I do so, I will explicitly specify the package name, using a command like `Hmisc::describe(x)`, rather than just `describe(x)`, so as to specify that I want the Hmisc package's version of `describe` applied to whatever `x` is. Those packages are:

- `aplpack` which provides `stem.leaf` and `stem.leaf.backback` for building fancier stem-and-leaf displays
- `arm` which provides a set of functions for model building and checking that are used in Gelman and Hill (2007)
- `car` which provides some tools for building scatterplot matrices, but also many other functions described in Fox and Weisberg (2011)
- `Epi` for 2x2 table analyses and materials for classical epidemiology: http://BendixCarstensen.com/Epi/
- `GGally` for scatterplot and correlation matrix visualizations: http://ggobi.github.io/ggally/
- `gridExtra` which includes a variety of functions for manipulating graphs: https://github.com/baptiste/gridextra
- `Hmisc` from Frank Harrell at Vanderbilt U., for its version of `describe` and for many regression modeling functions we'll use in 432. Details on Hmisc are at http://biostat.mc.vanderbilt.edu/wiki/Main/Hmisc. Frank has written several books - the most useful of which for 431 students is probably Harrell and Slaughter (2017)
- `mice`, which we'll use (a little) in 431 for multiple imputation to deal with missing data: http://www.stefvanbuuren.nl/mi/
- `mosaic`, mostly for its `favstats` summary, but Project MOSAIC is a community of educators you might be interested in: http://mosaic-web.org/
- `psych` for its own version of `describe`, but other features are described at http://personality-project.org/r/psych/

We also will use a package called `xda` for two functions called `numSummary` and `charSummary`, but that package gets loaded via `devtools` and GitHub by the code in these Notes.

When compiling the Notes from the original code files, these packages will need to be installed (but not loaded) in R, or an error will be thrown when compiling this document. To install all of the packages used within these Notes, type in (or copy and paste) the following commands and run them in the R Console. Again, you only need to install a package once, but you need to reload it every time you start a new session.

```
pkgs <- c("aplpack", "arm", "boot", "car", "devtools", "Epi", "forcats", "GGally",
          "gridExtra", "Hmisc", "knitr", "mice", "mosaic", "NHANES", "pander",
          "psych", "pwr", "tidyverse", "viridis")
install.packages(pkgs)
```

Part A. Exploring Data

Chapter 3

Visualizing Data

Part A of these Notes is designed to ease your transition into working effectively with data, so that you can better understand it. We'll start by visualizing some data from the US National Health and Nutrition Examination Survey, or NHANES. We'll display R code as we go, but we'll return to all of the key coding ideas involved later in the Notes.

3.1 The NHANES data: Collecting a Sample

To begin, we'll gather a random sample of 1,000 subjects participating in NHANES, and then identify several variables of interest about those subjects[1]. The motivation for this example came from a Figure in Baumer, Kaplan, and Horton (2017).

```
library(NHANES) # load the NHANES package/library of functions, data

set.seed(431001)
# use set.seed to ensure that we all get the same random sample
# of 1,000 NHANES subjects in our nh_data collection

nh_data <- sample_n(NHANES, size = 1000) %>%
    select(ID, Gender, Age, Height, Weight, BMI, Pulse, Race1, HealthGen, Diabetes)

nh_data
```

```
# A tibble: 1,000 x 10
      ID Gender   Age Height Weight   BMI Pulse    Race1 HealthGen
   <int> <fctr> <int>  <dbl>  <dbl> <dbl> <int>   <fctr>    <fctr>
 1 59640   male    54  175.7  129.0 41.79    74    White      Good
 2 59826 female    67  156.5   50.2 20.50    66    White     Vgood
 3 56340   male     9  128.3   23.3 14.15    86    Black        NA
 4 56747   male    33  194.2  105.1 27.87    68    White     Vgood
 5 51754 female    58  167.2  106.0 37.92    70    White        NA
 6 52712   male     6  108.6   16.9 14.33    NA    White        NA
 7 63908   male    55  168.6   90.6 31.90    62  Mexican     Vgood
 8 60865 female    25  155.5   55.0 22.75    58    Other     Vgood
 9 66642   male    41  177.9   89.3 28.20    72    White     Vgood
10 59880 female    45  163.2   98.3 36.91    80 Hispanic      Good
```

[1]For more on the NHANES data available in the NHANES package, type ?NHANES in the Console in R Studio.

```
# ... with 990 more rows, and 1 more variables: Diabetes <fctr>
```

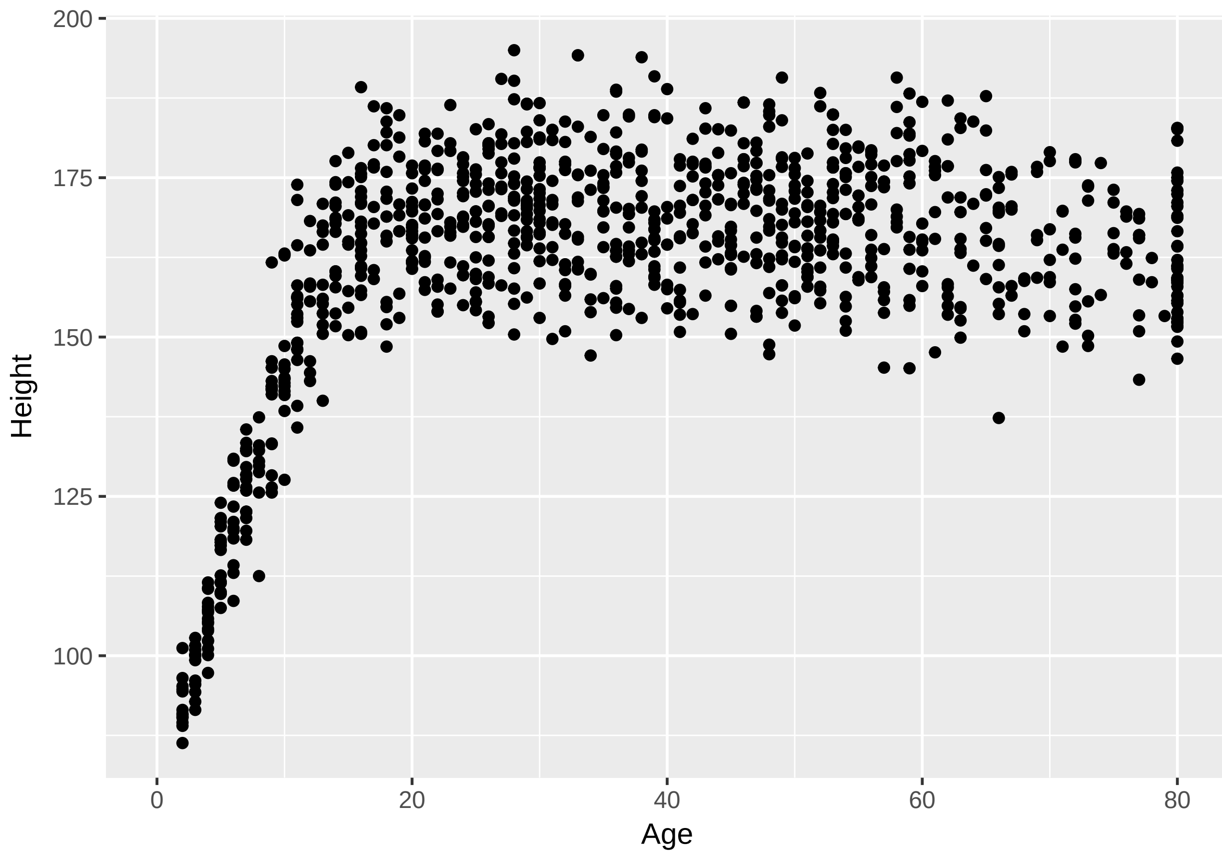
We have 1000 rows (observations) and 10 columns (variables) that describe the subjects listed in the rows.

3.2 Age and Height

Suppose we want to visualize the relationship of Height and Age in our 1,000 NHANES observations. The best choice is likely to be a scatterplot.

```
ggplot(data = nh_data, aes(x = Age, y = Height)) +
    geom_point()
```

```
Warning: Removed 25 rows containing missing values (geom_point).
```



We note several interesting results here.

1. As a warning, R tells us that it has "Removed 25 rows containing missing values (geom_point)." Only 975 subjects plotted here, because the remaining 25 people have missing (NA) values for either Height, Age or both.
2. Unsurprisingly, the measured Heights of subjects grow from Age 0 to Age 20 or so, and we see that a typical Height increases rapidly across these Ages. The middle of the distribution at later Ages is pretty consistentat at a Height somewhere between 150 and 175. The units aren't specified, but we expect they must be centimeters. The Ages are clearly reported in Years.
3. No Age is reported over 80, and it appears that there is a large cluster of Ages at 80. This may be due to a requirement that Ages 80 and above be reported at 80 so as to help mask the identity of those

individuals.[2]

As in this case, we're going to build most of our visualizations using tools from the `ggplot2` package, which is part of the `tidyverse` series of packages. You'll see similar coding structures throughout this Chapter, most of which are covered as well in Chapter 3 of Grolemund and Wickham (2017).

3.3 Subset of Subjects with Known Age and Height

Before we move on, let's manipulate the data set a bit, to focus on only those subjects who have complete data on both Age and Height. This will help us avoid that warning message.

```
nh_dat2 <- nh_data %>%
    filter(complete.cases(Age, Height))

summary(nh_dat2)
```

```
       ID           Gender          Age            Height     
 Min.   :51654   female:498   Min.   : 2.00   Min.   : 86.3  
 1st Qu.:56753   male  :477   1st Qu.:20.00   1st Qu.:156.4  
 Median :61453                Median :36.00   Median :165.8  
 Mean   :61602                Mean   :37.27   Mean   :161.7  
 3rd Qu.:66484                3rd Qu.:53.00   3rd Qu.:174.1  
 Max.   :71826                Max.   :80.00   Max.   :195.0  
                                                             
     Weight            BMI            Pulse             Race1    
 Min.   : 12.50   Min.   :13.17   Min.   : 42.00   Black   :112  
 1st Qu.: 57.60   1st Qu.:21.60   1st Qu.: 66.00   Hispanic: 69  
 Median : 73.40   Median :26.10   Median : 72.00   Mexican :104  
 Mean   : 73.41   Mean   :26.96   Mean   : 73.75   White   :607  
 3rd Qu.: 90.20   3rd Qu.:31.10   3rd Qu.: 82.00   Other   : 83  
 Max.   :198.70   Max.   :80.60   Max.   :124.00                 
 NA's   :2        NA's   :2       NA's   :120                    
     HealthGen    Diabetes  
 Excellent: 87   No  :910  
 Vgood    :276   Yes : 64  
 Good     :276   NA's:  1  
 Fair     :103             
 Poor     : 15             
 NA's     :218             
```

Note that the units and explanations for these variables are contained in the NHANES help file, available via ?NHANES in the Console of R Studio.

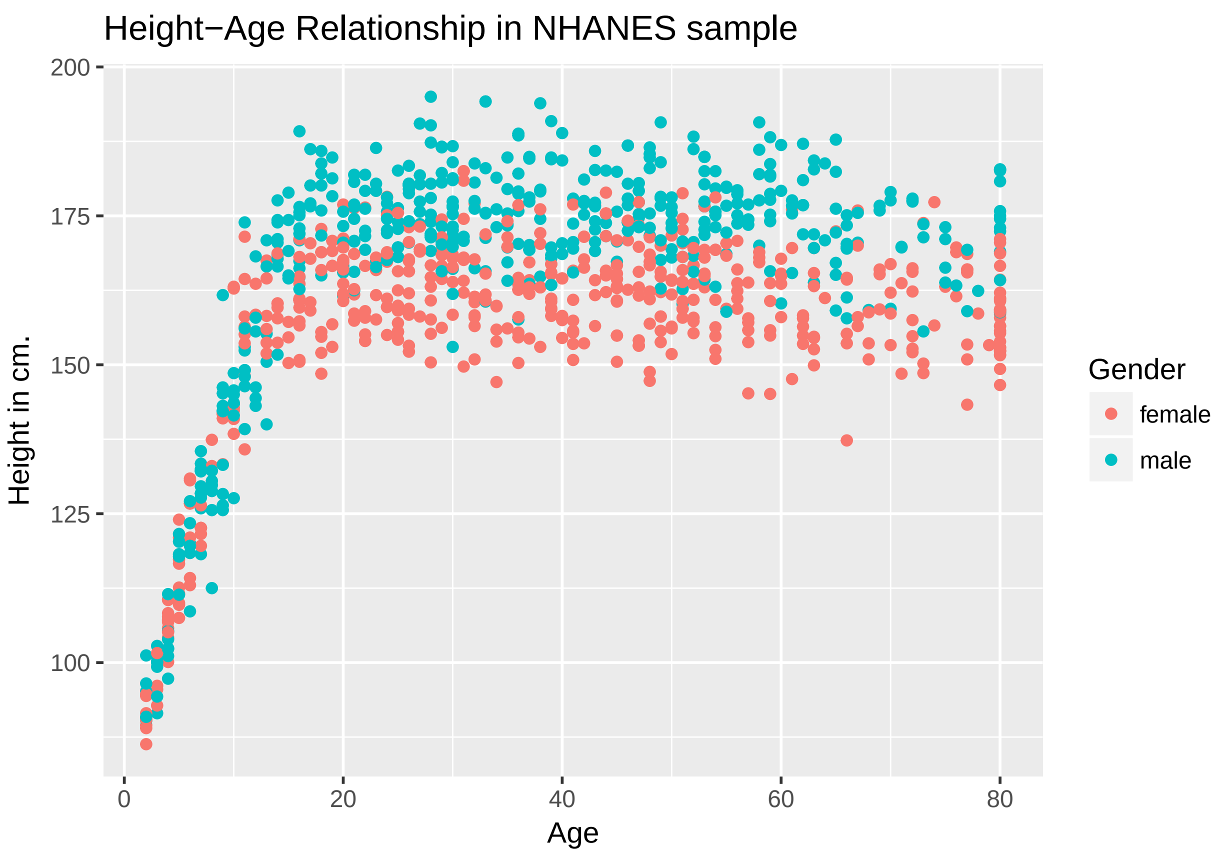
3.4 Age-Height and Gender?

Let's add Gender to the plot using color, and also adjust the y axis label to incorporate the units of measurement.

```
ggplot(data = nh_dat2, aes(x = Age, y = Height, color = Gender)) +
    geom_point() +
```

[2] If you visit the NHANES help file with ?NHANES, you will see that subjects 80 years or older were indeed recorded as 80.

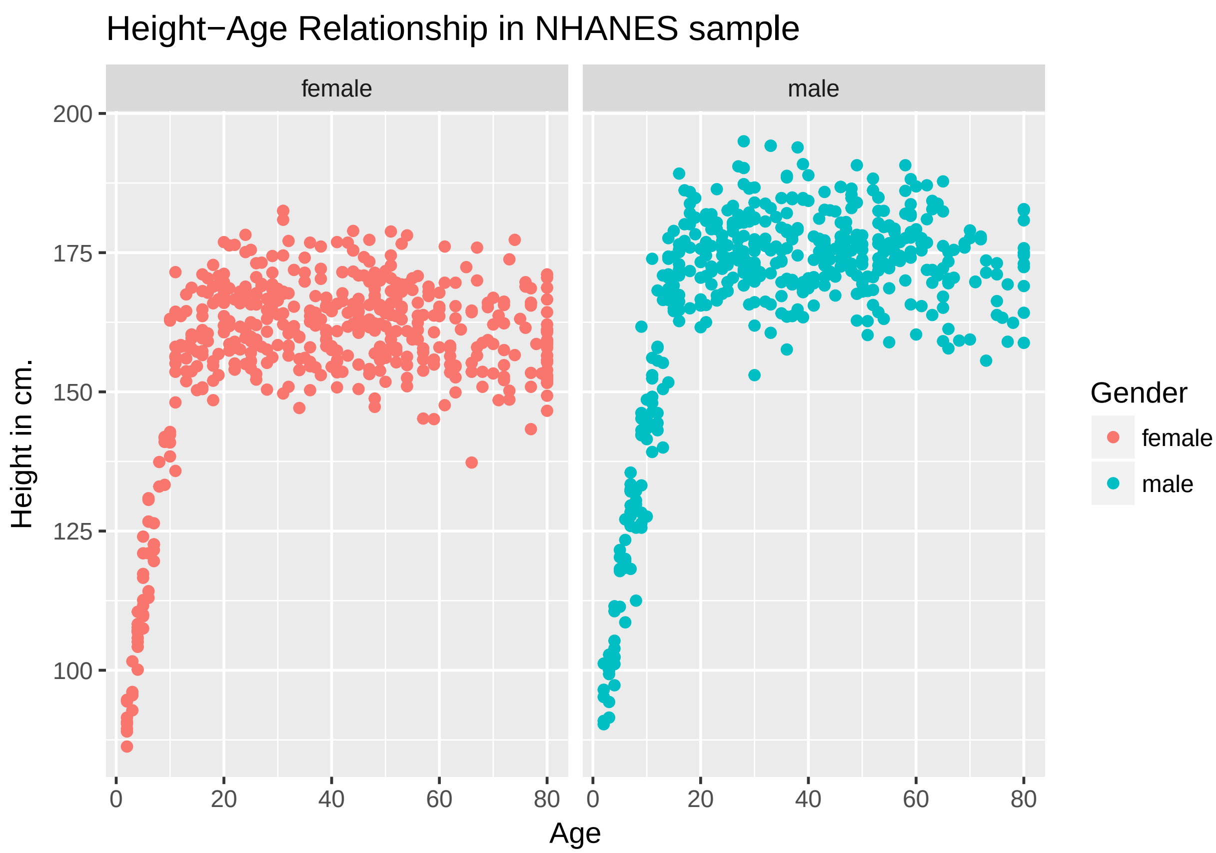
```
    labs(title = "Height-Age Relationship in NHANES sample",
         y = "Height in cm.")
```



3.4.1 Can we show the Female and Male relationships in separate panels?

Sure.

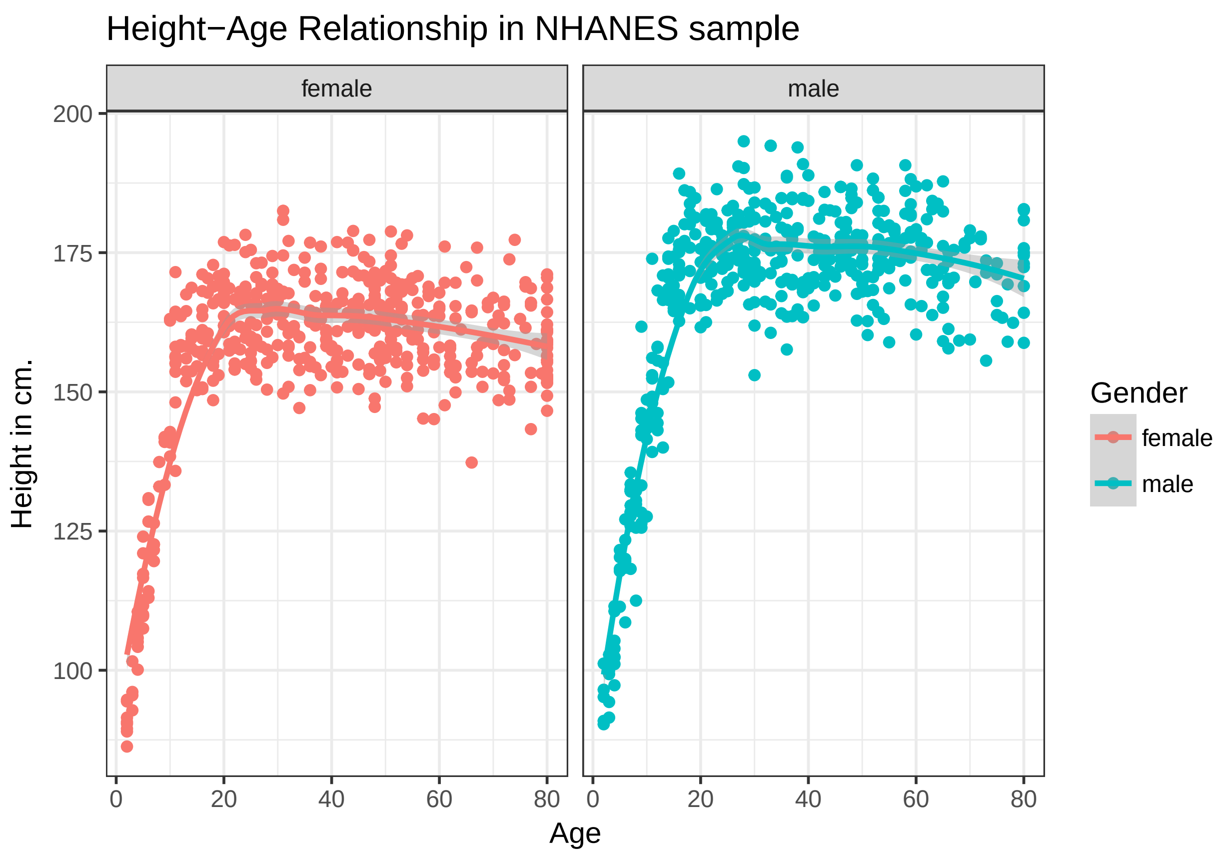
```
ggplot(data = nh_dat2, aes(x = Age, y = Height, color = Gender)) +
    geom_point() +
    labs(title = "Height-Age Relationship in NHANES sample",
         y = "Height in cm.") +
    facet_wrap(~ Gender)
```



3.4.2 Can we add a smooth curve to show the relationship in each plot?

Yep, and let's change the theme of the graph to remove the gray background, too.

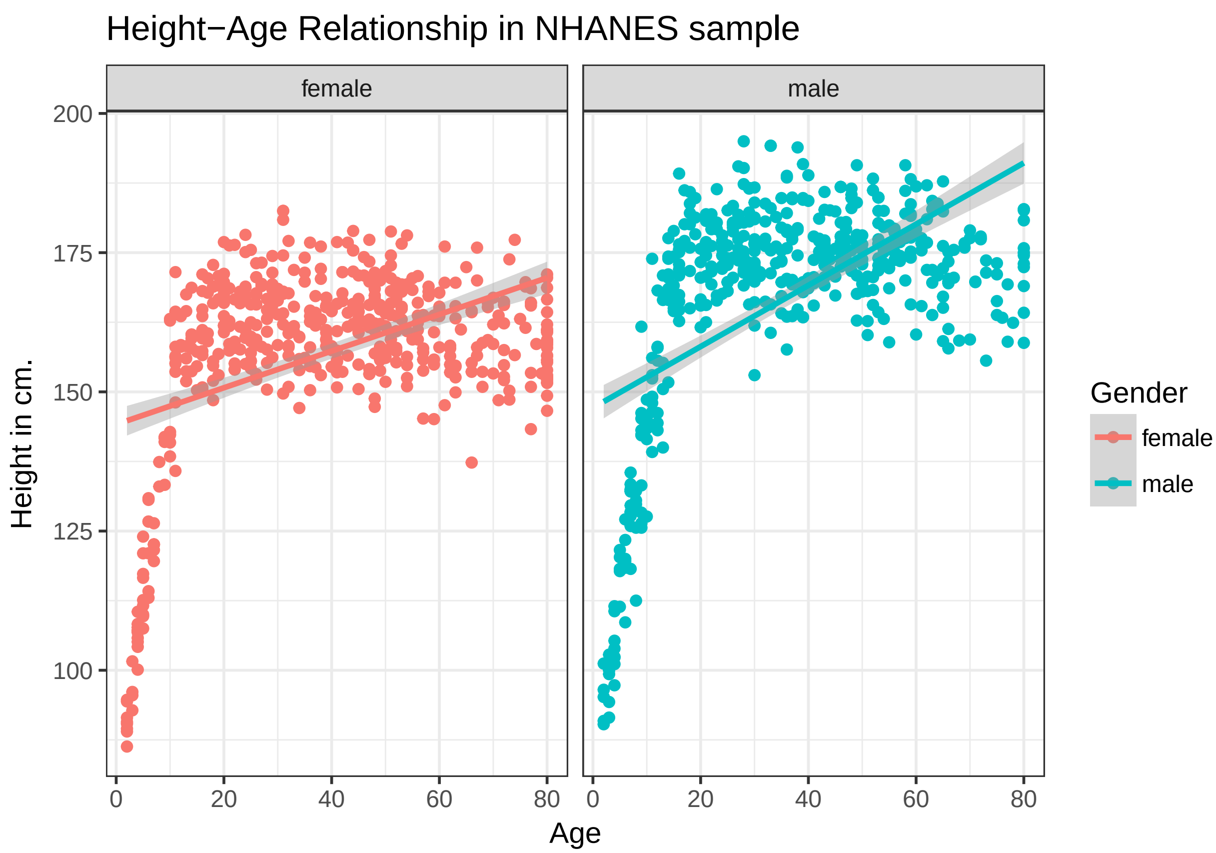
```
ggplot(data = nh_dat2, aes(x = Age, y = Height, color = Gender)) +
    geom_point() +
    geom_smooth(method = "loess") +
    labs(title = "Height-Age Relationship in NHANES sample",
         y = "Height in cm.") +
    theme_bw() +
    facet_wrap(~ Gender)
```



3.4.3 What if we want to assume straight line relationships?

We could look at a linear model in the plot. Does this make sense here?

```
ggplot(data = nh_dat2, aes(x = Age, y = Height, color = Gender)) +
    geom_point() +
    geom_smooth(method = "lm") +
    labs(title = "Height-Age Relationship in NHANES sample",
         y = "Height in cm.") +
    theme_bw() +
    facet_wrap(~ Gender)
```



3.5 A Subset: Ages 21-79

Suppose we wanted to look at a subset of our sample - those observations (subjects) whose Age is at least 21 and at most 79. We'll create that sample below, and also subset the variables to include nine of particular interest, and remove any observations with any missingness on *any* of the nine variables we're including here.

```
nh_data_2179 <- nh_data %>%
    filter(Age > 20 & Age < 80) %>%
    select(ID, Gender, Age, Height, Weight, BMI, Pulse, Race1, HealthGen, Diabetes) %>%
    na.omit

nh_data_2179
```

```
# A tibble: 594 x 10
      ID Gender   Age Height Weight   BMI Pulse    Race1 HealthGen
   <int> <fctr> <int>  <dbl>  <dbl> <dbl> <int>   <fctr>    <fctr>
 1 59640   male    54  175.7  129.0 41.79    74    White      Good
 2 59826 female    67  156.5   50.2 20.50    66    White     Vgood
 3 56747   male    33  194.2  105.1 27.87    68    White     Vgood
 4 63908   male    55  168.6   90.6 31.90    62  Mexican     Vgood
 5 60865 female    25  155.5   55.0 22.75    58    Other     Vgood
 6 66642   male    41  177.9   89.3 28.20    72    White     Vgood
 7 59880 female    45  163.2   98.3 36.91    80 Hispanic      Good
 8 71784 female    24  161.1   50.2 19.30    72    White     Vgood
 9 67616   male    63  184.3   70.0 20.60    82    White     Vgood
```

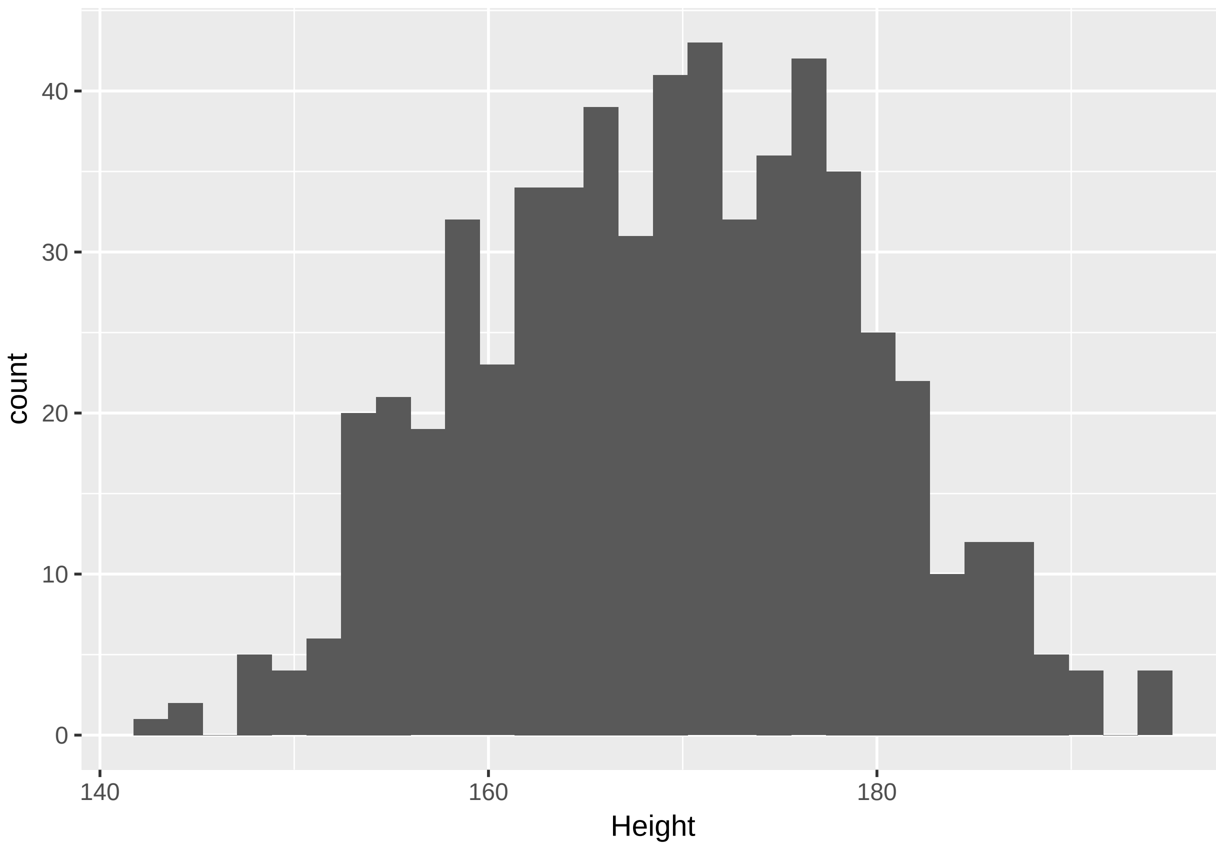
```
10 55391 female    32  161.4   69.2 26.56   114    Other      Good
# ... with 584 more rows, and 1 more variables: Diabetes <fctr>
```

3.6 Distribution of Heights

What is the distribution of height in this new sample?

```r
ggplot(data = nh_data_2179, aes(x = Height)) +
    geom_histogram()
```

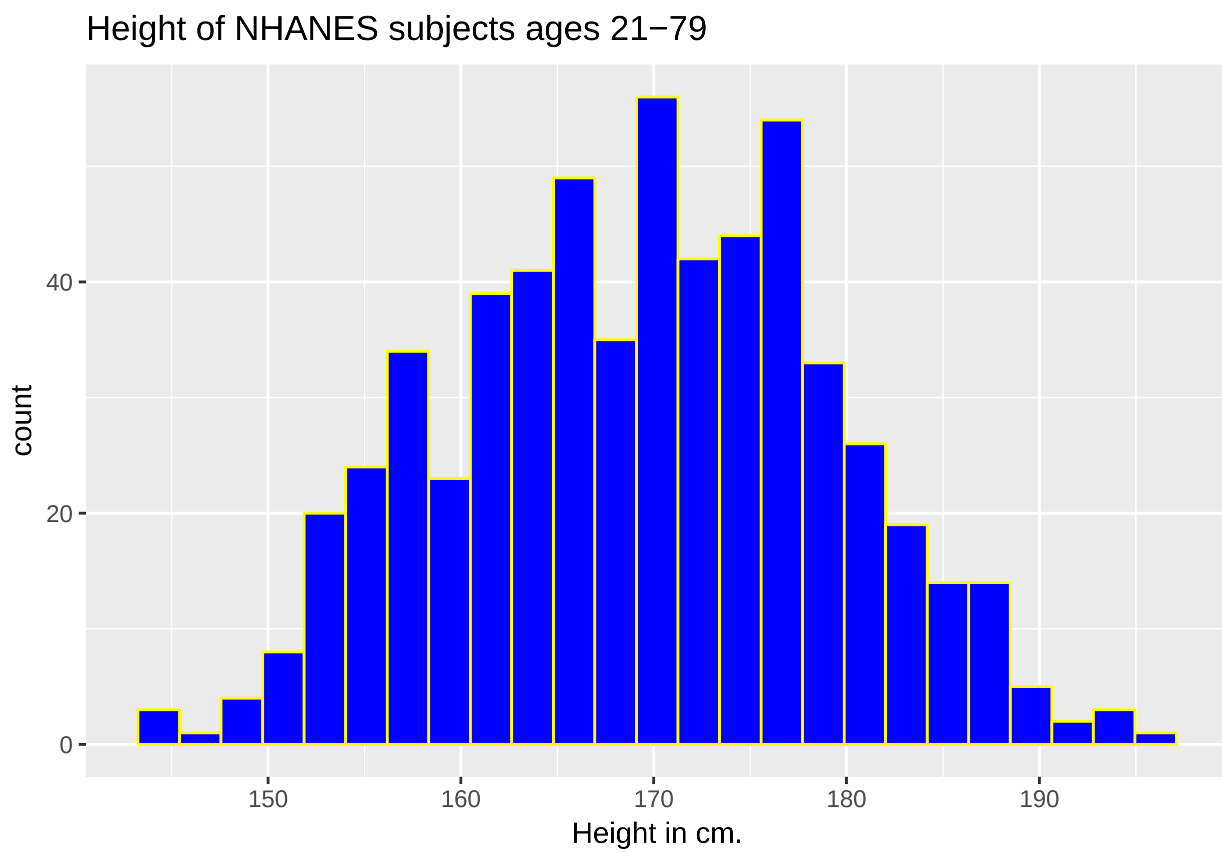
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



We can do several things to clean this up.

1. We'll change the color of the lines for each bar of the histogram.
2. We'll change the fill inside each bar to make them stand out a bit more.
3. We'll add a title and relabel the horizontal (x) axis to include the units of measurement.
4. We'll avoid the warning by selecting a number of bins (we'll use 25 here) into which we'll group the heights before drawing the histogram.

```r
ggplot(data = nh_data_2179, aes(x = Height)) +
    geom_histogram(bins = 25, col = "yellow", fill = "blue") +
    labs(title = "Height of NHANES subjects ages 21-79",
         x = "Height in cm.")
```

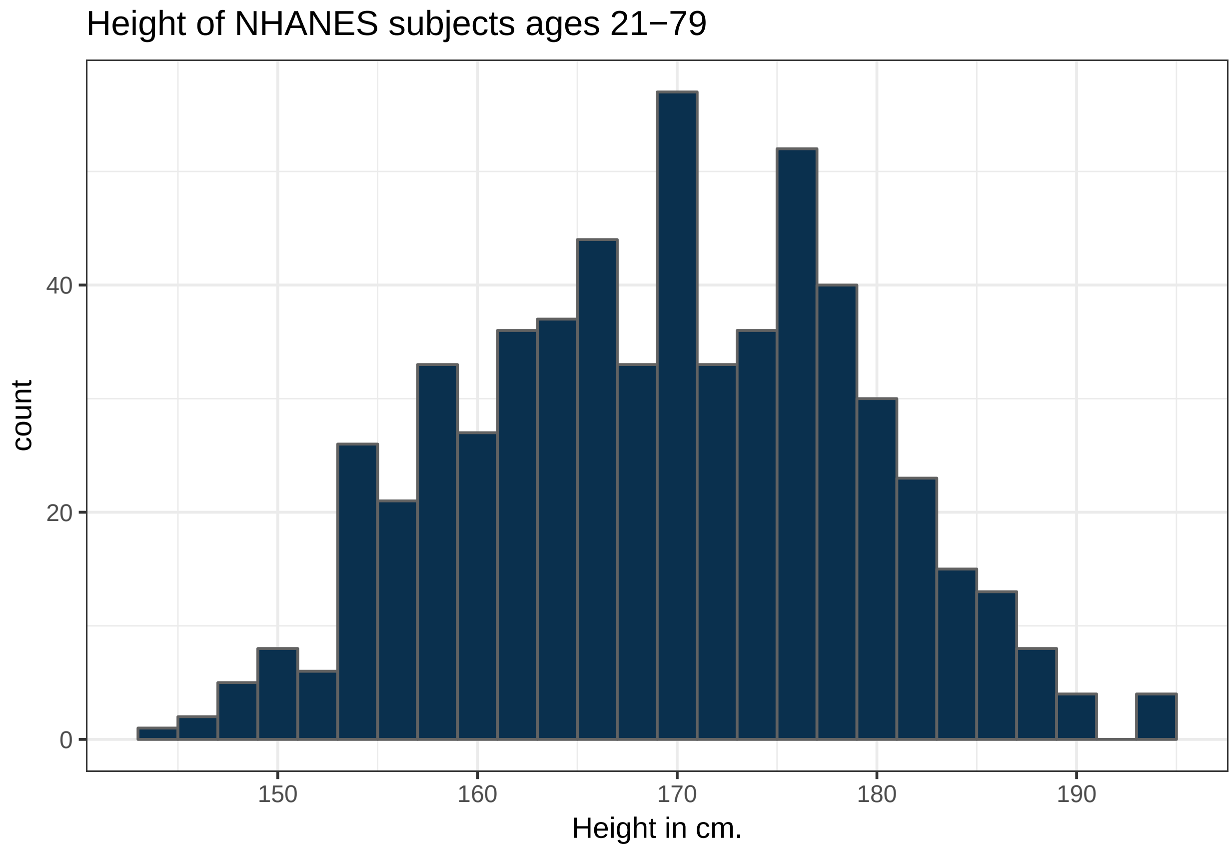


3.6.1 Changing a Histogram's Fill and Color

The CWRU color guide (https://case.edu/umc/our-brand/visual-guidelines/) lists the HTML color schemes for CWRU blue and CWRU gray. Let's match that color scheme.

```
cwru.blue <- '#0a304e'
cwru.gray <- '#626262'

ggplot(data = nh_data_2179, aes(x = Height)) +
    geom_histogram(binwidth = 2, col = cwru.gray, fill = cwru.blue) +
    labs(title = "Height of NHANES subjects ages 21-79",
         x = "Height in cm.") +
    theme_bw()
```

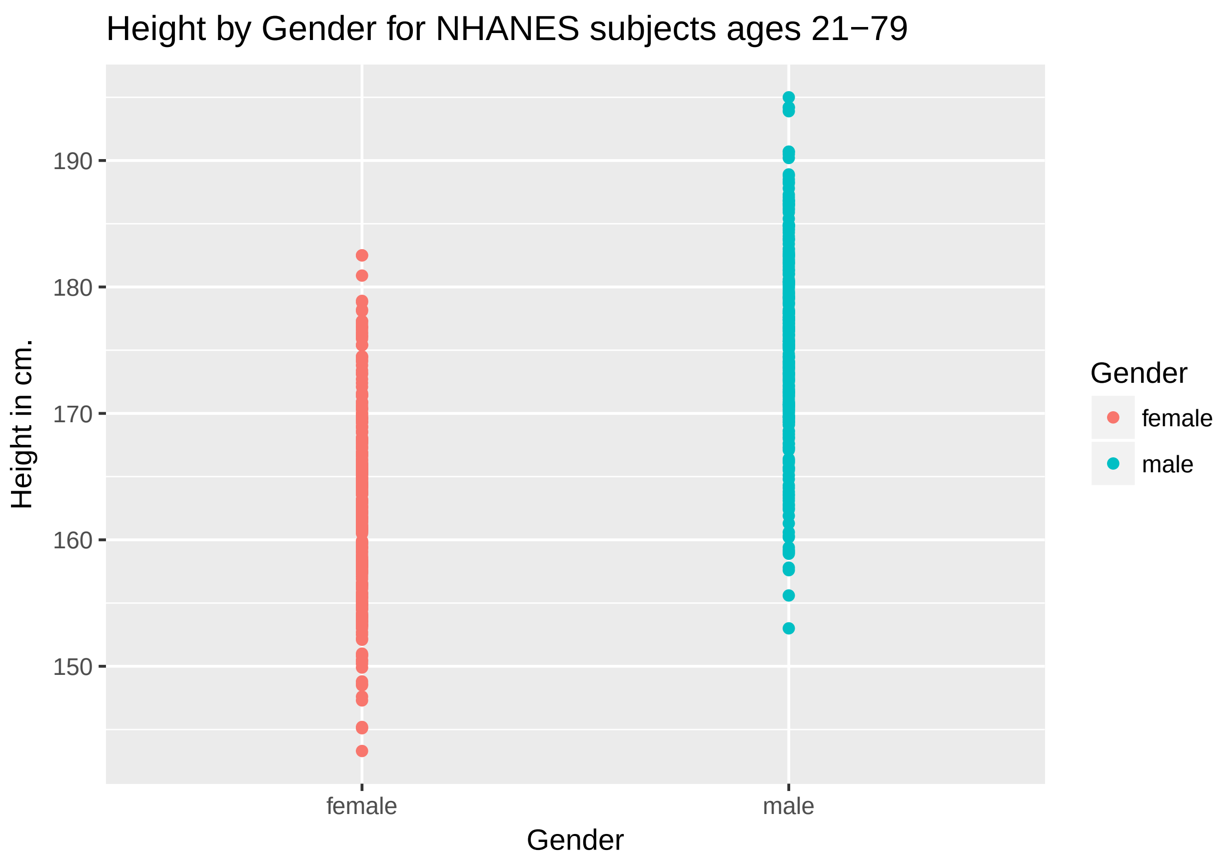


Note the other changes to the graph above.

1. We changed the theme to replace the gray background.
2. We changed the bins for the histogram, to gather observations into groups of 2 cm. each.

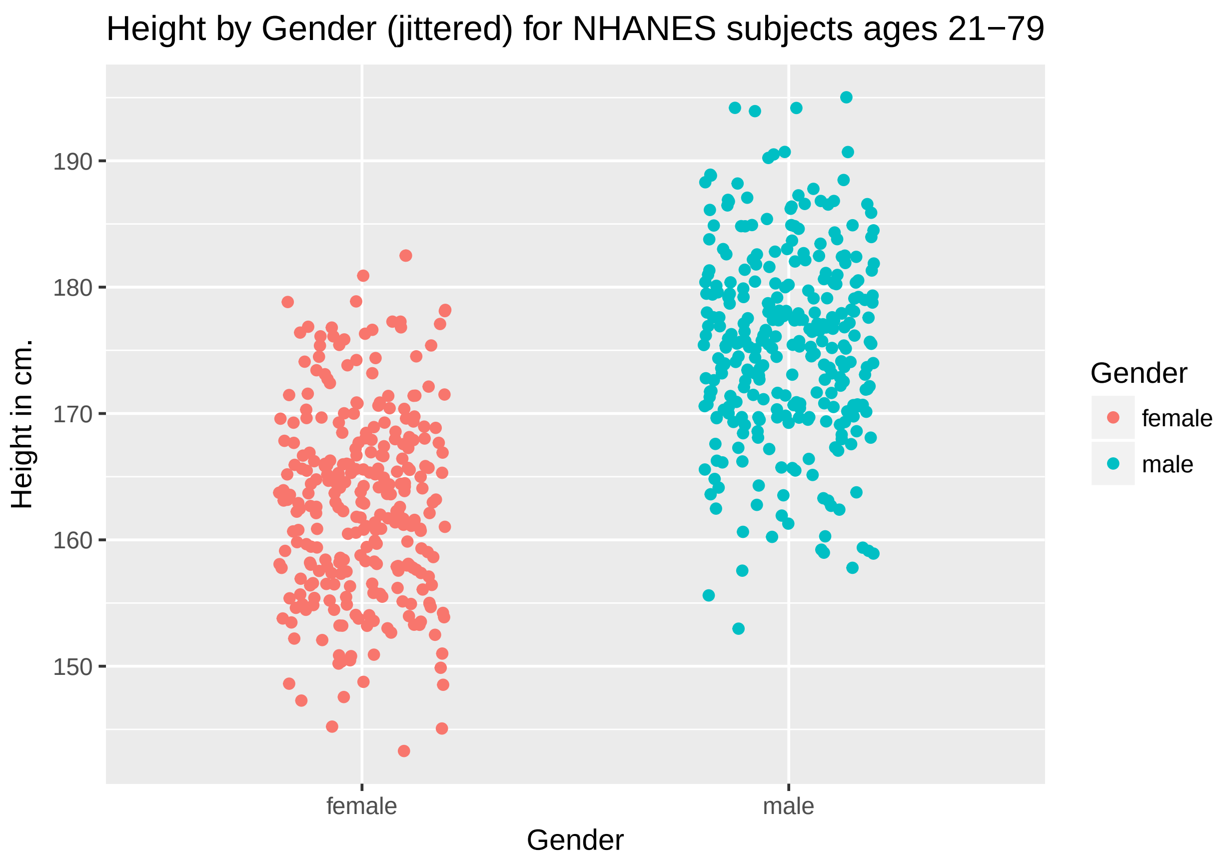
3.7 Height and Gender

```
ggplot(data = nh_data_2179, aes(x = Gender, y = Height, color = Gender)) +
    geom_point() +
    labs(title = "Height by Gender for NHANES subjects ages 21-79",
         y = "Height in cm.")
```



This plot isn't so useful. We can improve things a little by jittering the points horizontally, so that the overlap is reduced.

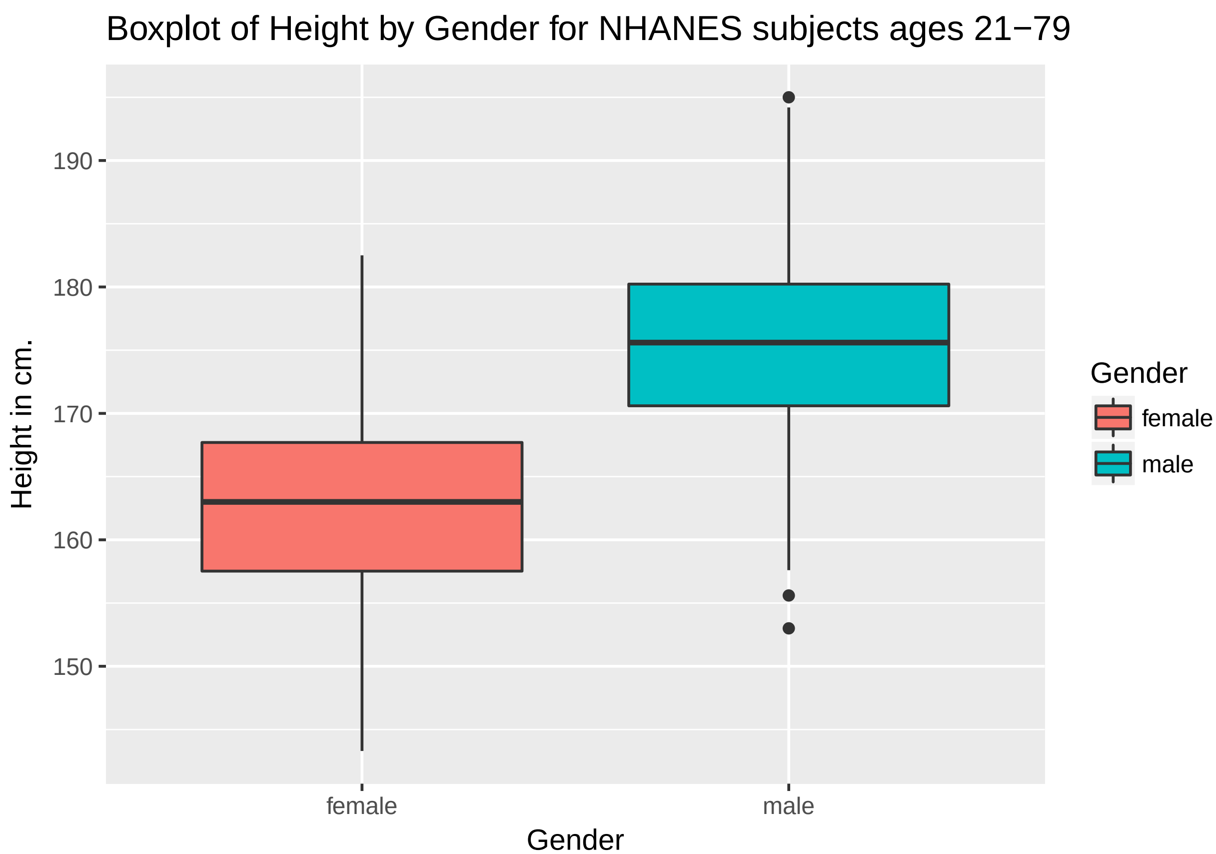
```
ggplot(data = nh_data_2179, aes(x = Gender, y = Height, color = Gender)) +
    geom_jitter(width = 0.2) +
    labs(title = "Height by Gender (jittered) for NHANES subjects ages 21-79",
         y = "Height in cm.")
```



Perhaps it might be better to summarize the distribution in a different way. We might consider a boxplot of the data.

3.7.1 A Boxplot of Height by Gender

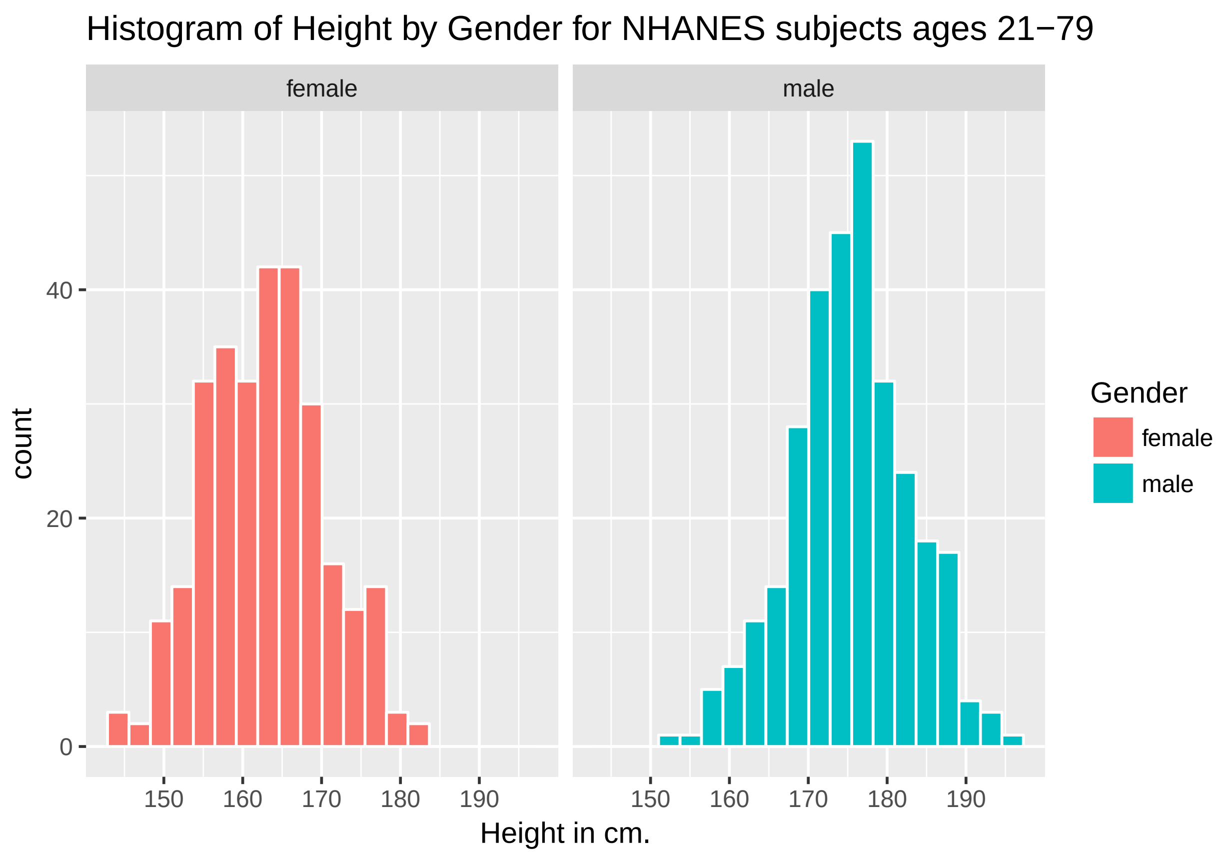
```
ggplot(data = nh_data_2179, aes(x = Gender, y = Height, fill = Gender)) +
    geom_boxplot() +
    labs(title = "Boxplot of Height by Gender for NHANES subjects ages 21-79",
         y = "Height in cm.")
```



Or perhaps we'd like to see a pair of histograms?

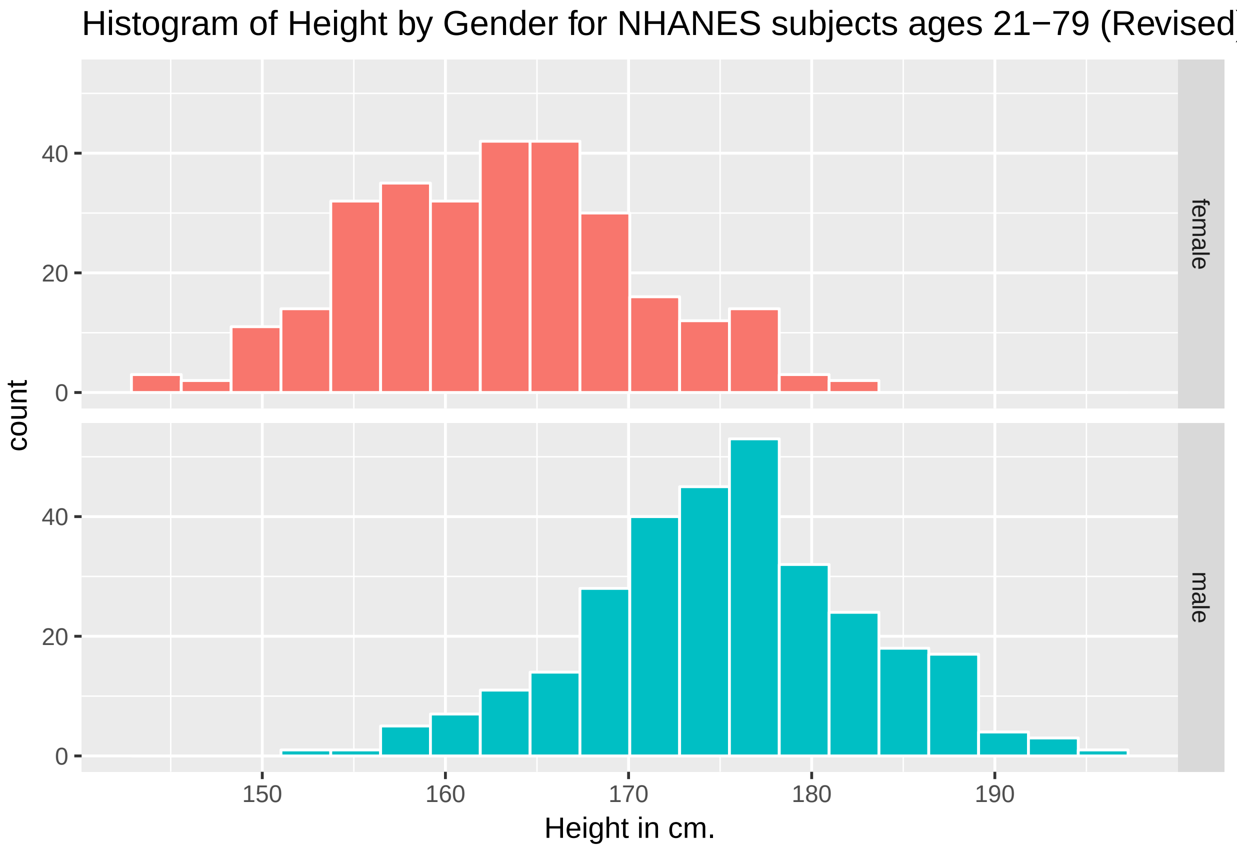
3.7.2 Histograms of Height by Gender

```
ggplot(data = nh_data_2179, aes(x = Height, fill = Gender)) +
    geom_histogram(color = "white", bins = 20) +
    labs(title = "Histogram of Height by Gender for NHANES subjects ages 21-79",
         x = "Height in cm.") +
    facet_wrap(~ Gender)
```



Can we redraw these histograms so that they are a little more comparable, and to get rid of the unnecessary legend?

```
ggplot(data = nh_data_2179, aes(x = Height, fill = Gender)) +
    geom_histogram(color = "white", bins = 20) +
    labs(title = "Histogram of Height by Gender for NHANES subjects ages 21-79 (Revised)",
         x = "Height in cm.") +
    guides(fill = FALSE) +
    facet_grid(Gender ~ .)
```



3.8 A Look at Body-Mass Index

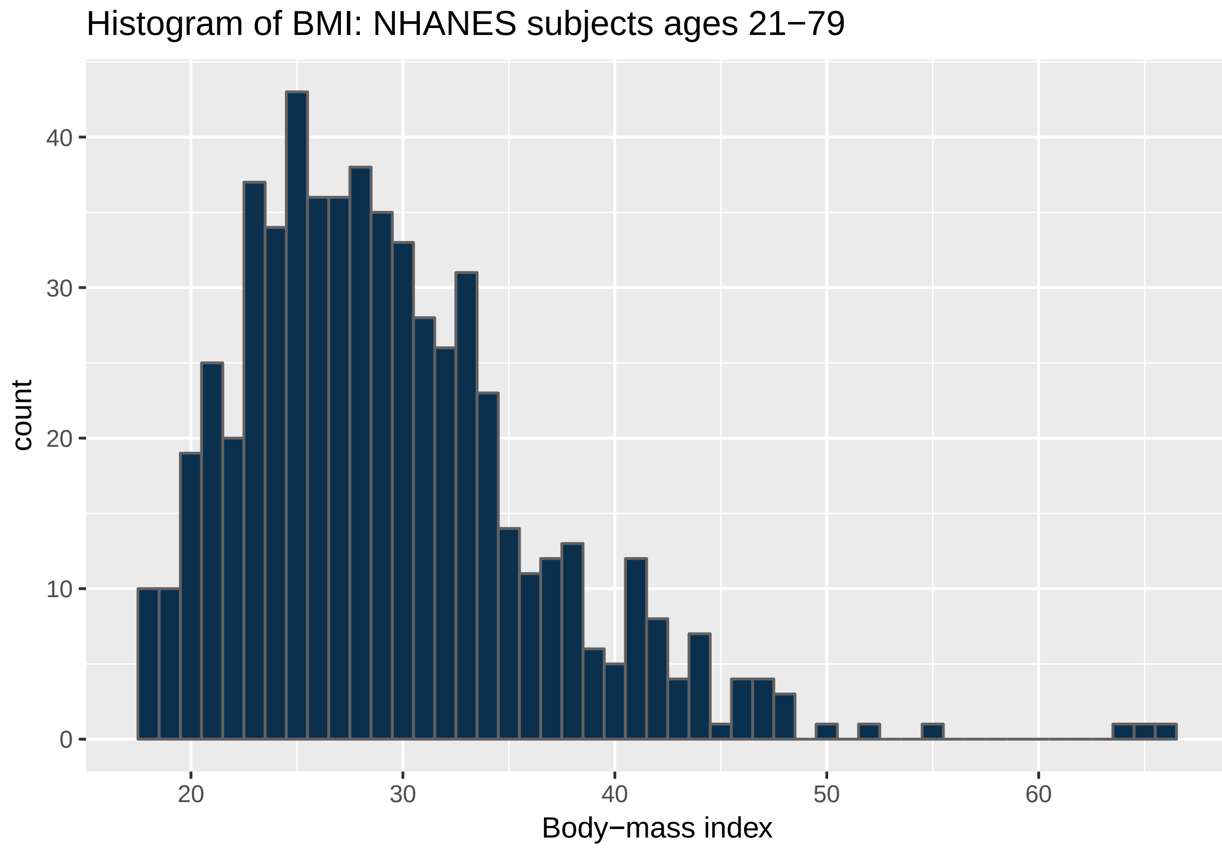
Let's look at a different outcome, the *body-mass index*, or BMI. The definition of BMI for adult subjects (which is expressed in units of kg/m^2) is:

$$\text{BMI} = \frac{\text{weight in kg}}{(\text{height in meters})^2} = 703 \times \frac{\text{weight in pounds}}{(\text{height in inches})^2}$$

> [BMI is essentially] … a measure of a person's *thinness* or *thickness*… BMI was designed for use as a simple means of classifying average sedentary (physically inactive) populations, with an average body composition. For these individuals, the current value recommendations are as follow: a BMI from 18.5 up to 25 may indicate optimal weight, a BMI lower than 18.5 suggests the person is underweight, a number from 25 up to 30 may indicate the person is overweight, and a number from 30 upwards suggests the person is obese.
>
> Wikipedia, https://en.wikipedia.org/wiki/Body_mass_index

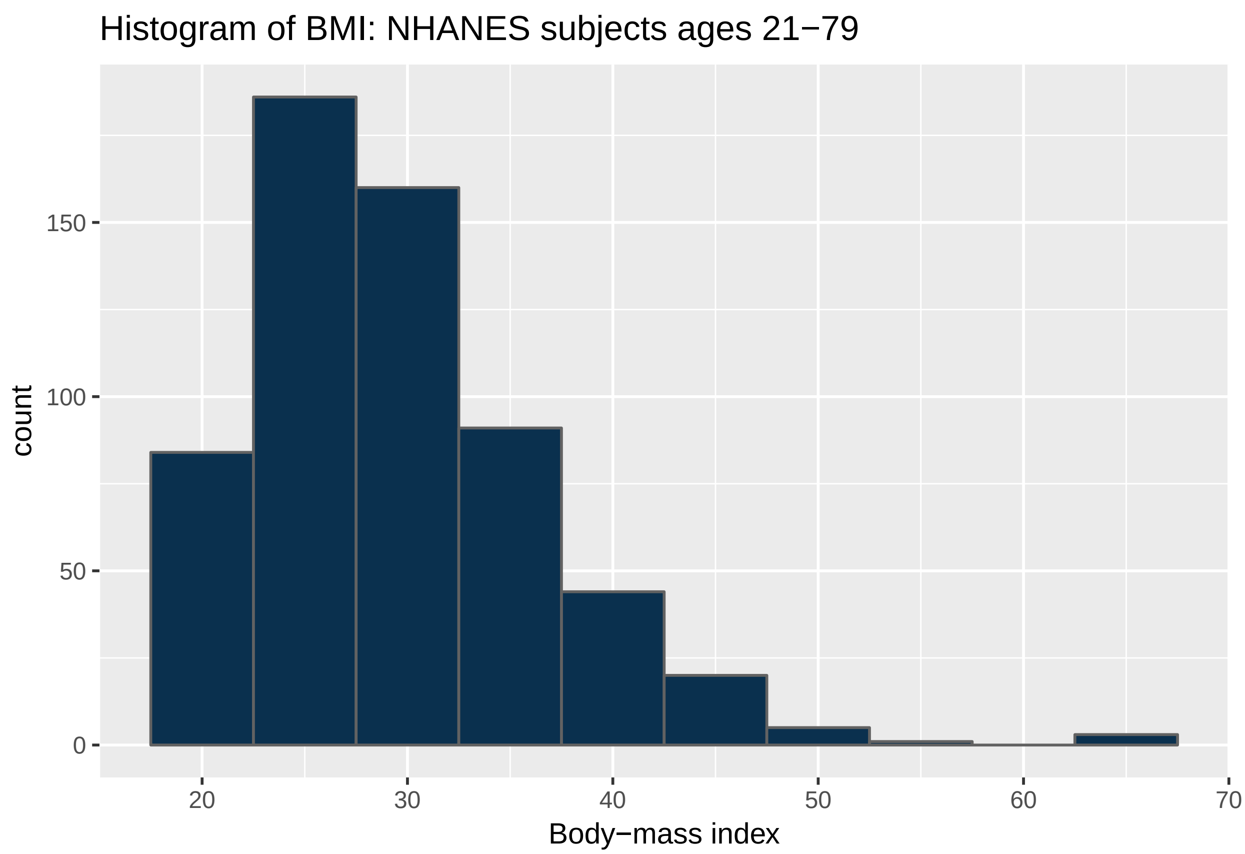
Here's a histogram, again with CWRU colors, for the BMI data.

```
ggplot(data = nh_data_2179, aes(x = BMI)) +
    geom_histogram(binwidth = 1, fill = cwru.blue, col = cwru.gray) +
    labs(title = "Histogram of BMI: NHANES subjects ages 21-79",
         x = "Body-mass index")
```



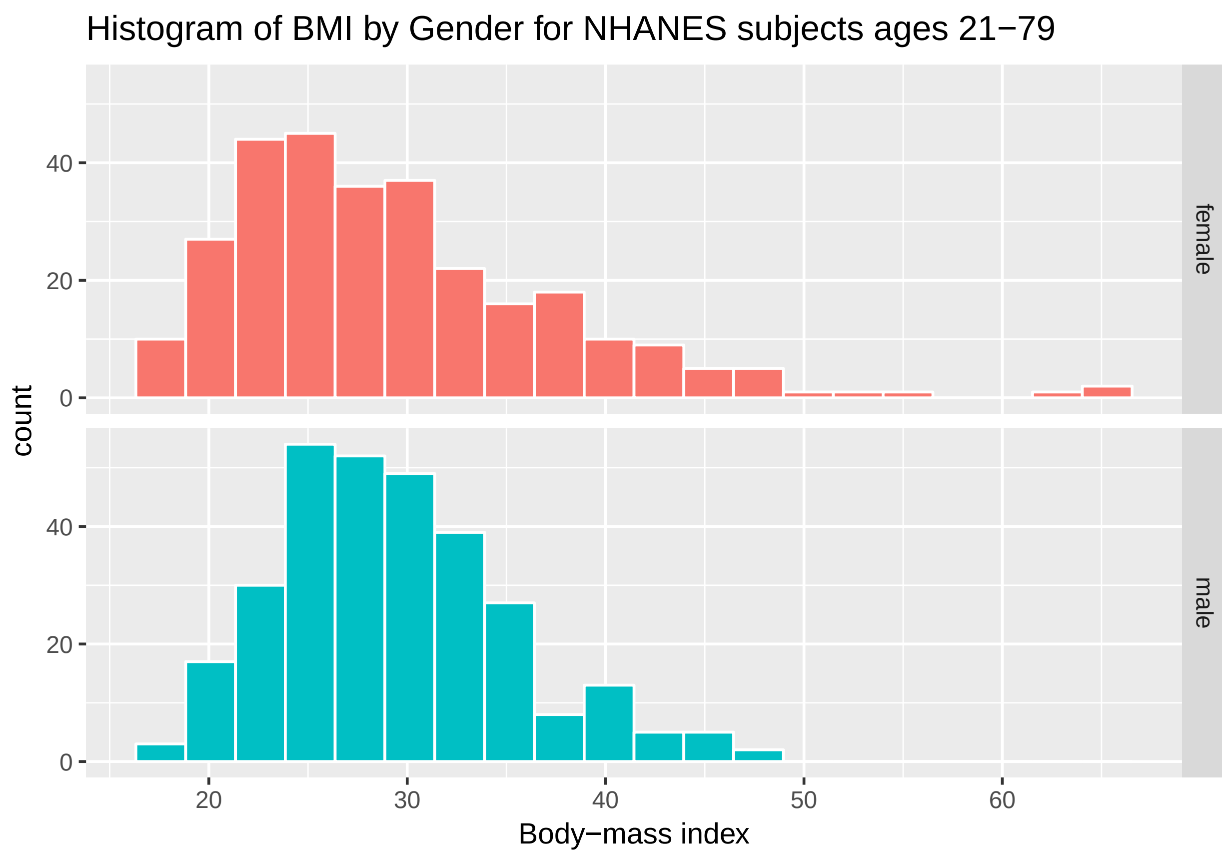
Note how different this picture looks if instead we bin up groups of 5 kg/m^2 at a time. Which is the more useful representation will depend a lot on what questions you're trying to answer.

```
ggplot(data = nh_data_2179, aes(x = BMI)) +
    geom_histogram(binwidth = 5, fill = cwru.blue, col = cwru.gray) +
    labs(title = "Histogram of BMI: NHANES subjects ages 21-79",
         x = "Body-mass index")
```



3.8.1 BMI by Gender

```
ggplot(data = nh_data_2179, aes(x = BMI, fill = Gender)) +
    geom_histogram(color = "white", bins = 20) +
    labs(title = "Histogram of BMI by Gender for NHANES subjects ages 21-79",
         x = "Body-mass index") +
    guides(fill = FALSE) +
    facet_grid(Gender ~ .)
```



As an accompanying numerical summary, we might ask how many people fall into each of these Gender categories, and what is their "average" BMI.

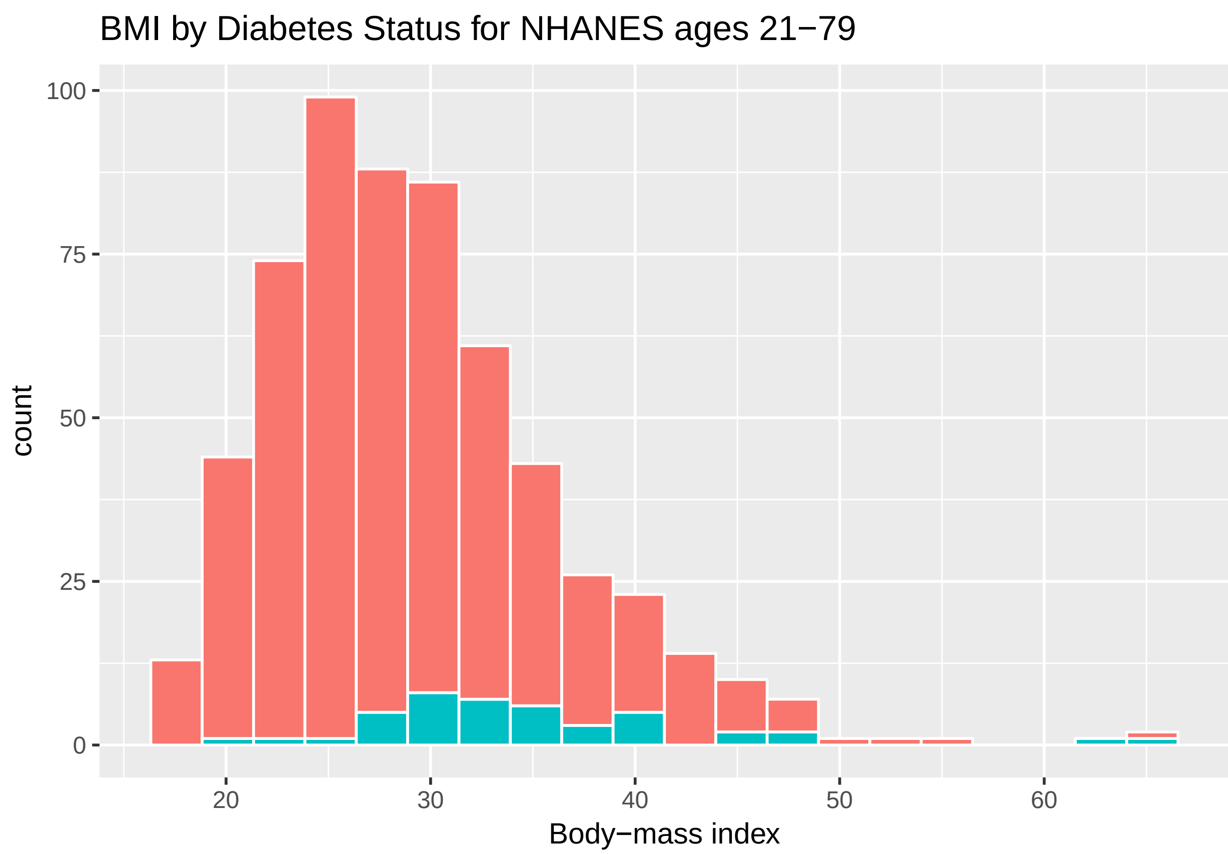
```
nh_data_2179 %>%
    group_by(Gender) %>%
    summarize(count = n(), mean(BMI), median(BMI)) %>%
    knitr::kable()
```

Gender	count	mean(BMI)	median(BMI)
female	290	29.35486	27.43
male	304	29.35773	28.69

3.8.2 BMI and Diabetes

We can split up our histogram into groups based on whether the subjects have been told they have diabetes.

```
ggplot(data = nh_data_2179, aes(x = BMI, fill = Diabetes)) +
    geom_histogram(color = "white", bins = 20) +
    labs(title = "BMI by Diabetes Status for NHANES ages 21-79",
         x = "Body-mass index") +
    guides(fill = FALSE)
```



How many people fall into each of these Diabetes categories, and what is their "average" BMI?

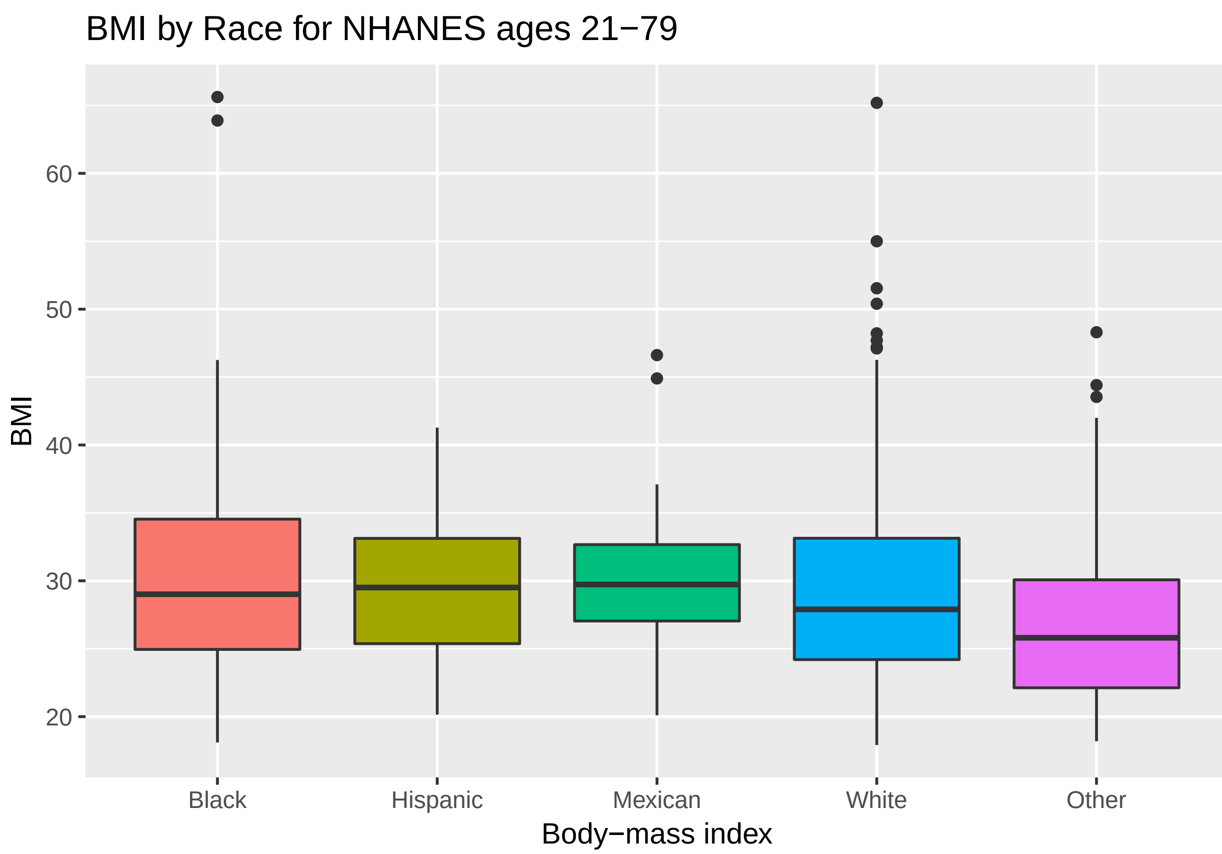
```
nh_data_2179 %>%
    group_by(Diabetes) %>%
    summarize(count = n(), mean(BMI), median(BMI)) %>%
    knitr::kable()
```

Diabetes	count	mean(BMI)	median(BMI)
No	551	28.89544	27.89
Yes	43	35.26209	33.43

3.8.3 BMI and Race

We can compare the distribution of BMI across Race groups, as well.

```
ggplot(data = nh_data_2179, aes(x = Race1, y = BMI, fill = Race1)) +
    geom_boxplot() +
    labs(title = "BMI by Race for NHANES ages 21-79",
         x = "Body-mass index") +
    guides(fill = FALSE)
```



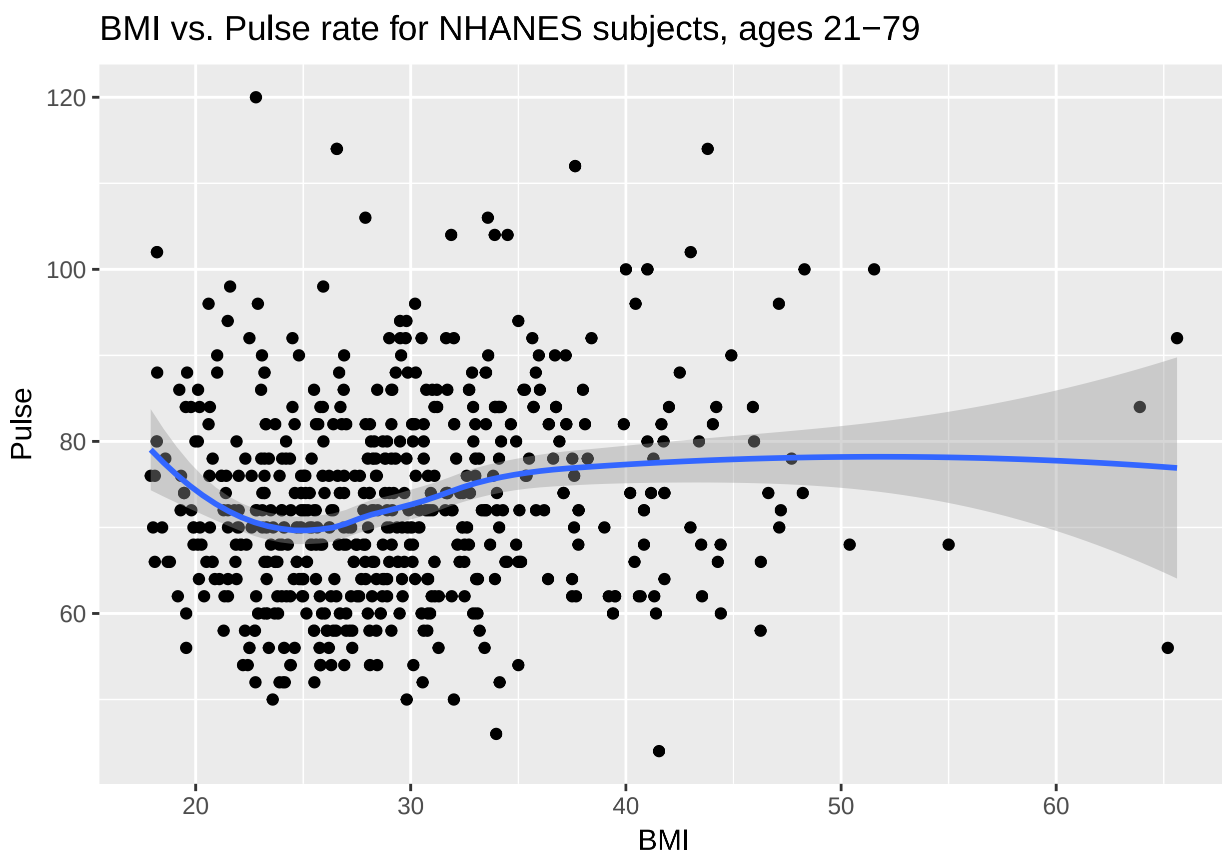
How many people fall into each of these Race1 categories, and what is their "average" BMI?

```
nh_data_2179 %>%
    group_by(Race1) %>%
    summarize(count = n(), mean(BMI), median(BMI)) %>%
    knitr::kable()
```

Race1	count	mean(BMI)	median(BMI)
Black	63	31.04444	29.010
Hispanic	44	29.36227	29.505
Mexican	50	29.97040	29.730
White	387	29.27326	27.900
Other	50	27.25300	25.805

3.8.4 BMI and Pulse Rate

```
ggplot(data = nh_data_2179, aes(x = BMI, y = Pulse)) +
    geom_point() +
    geom_smooth(method = "loess") +
    labs(title = "BMI vs. Pulse rate for NHANES subjects, ages 21-79")
```

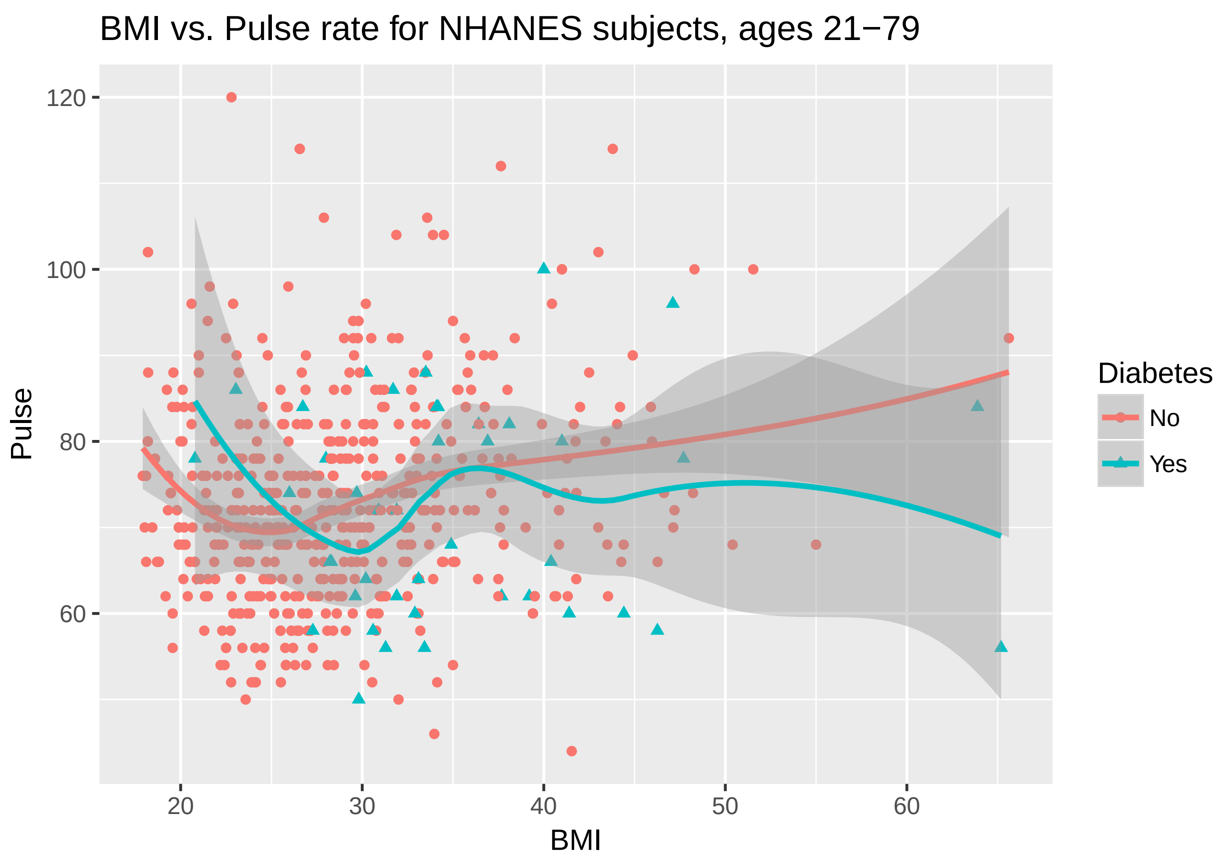


3.8.5 Diabetes vs. No Diabetes

Could we see whether subjects who have been told they have diabetes show different BMI-pulse rate patterns than the subjects who haven't?

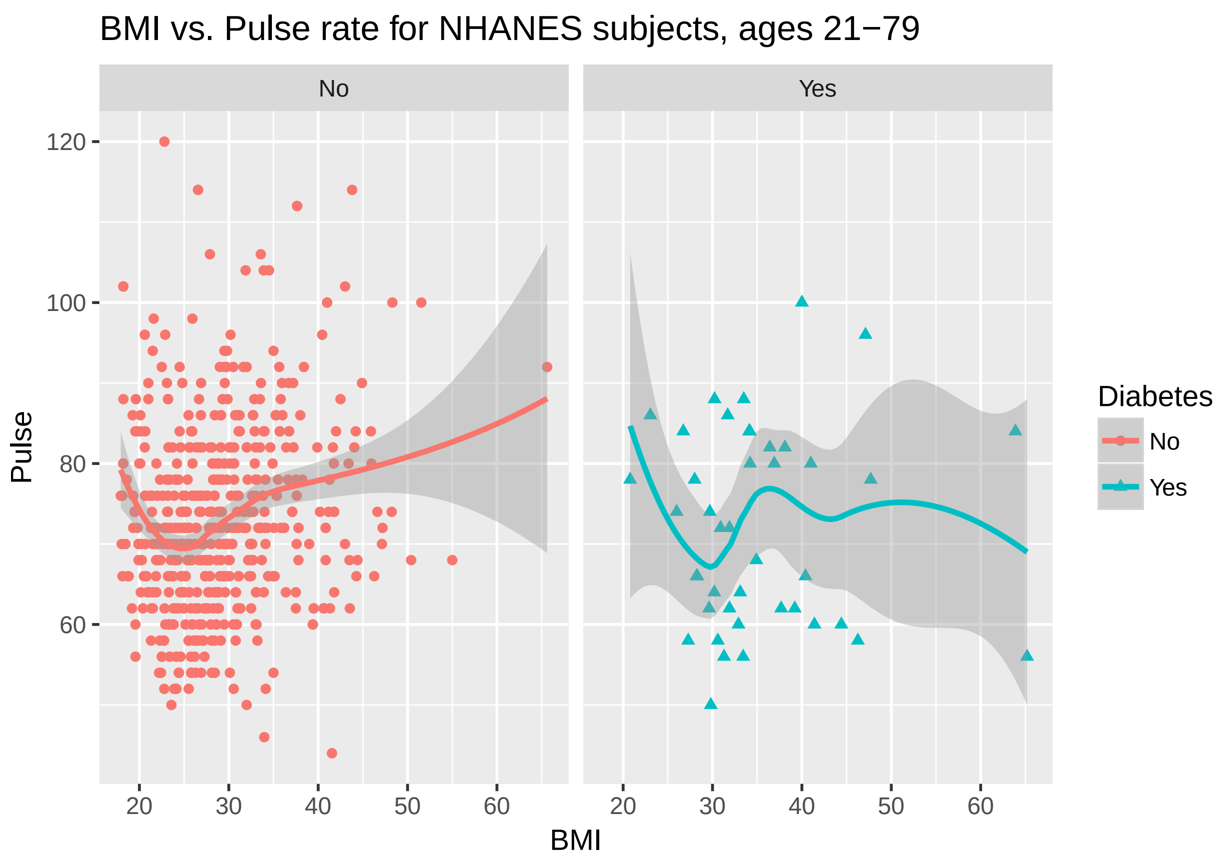
- Let's try doing this by changing the shape *and* the color of the points based on diabetes status.

```
ggplot(data = nh_data_2179,
       aes(x = BMI, y = Pulse,
           color = Diabetes, shape = Diabetes)) +
    geom_point() +
    geom_smooth(method = "loess") +
    labs(title = "BMI vs. Pulse rate for NHANES subjects, ages 21-79")
```



This plot might be easier to interpret if we faceted by Diabetes status, as well.

```
ggplot(data = nh_data_2179,
       aes(x = BMI, y = Pulse,
           color = Diabetes, shape = Diabetes)) +
    geom_point() +
    geom_smooth(method = "loess") +
    labs(title = "BMI vs. Pulse rate for NHANES subjects, ages 21-79") +
    facet_wrap(~ Diabetes)
```



3.9 General Health Status

Here's a Table of the General Health Status results. This is a self-reported rating of each subject's health on a five point scale (Excellent, Very Good, Good, Fair, Poor.)

```
nh_data_2179 %>%
    select(HealthGen) %>%
    table()
```

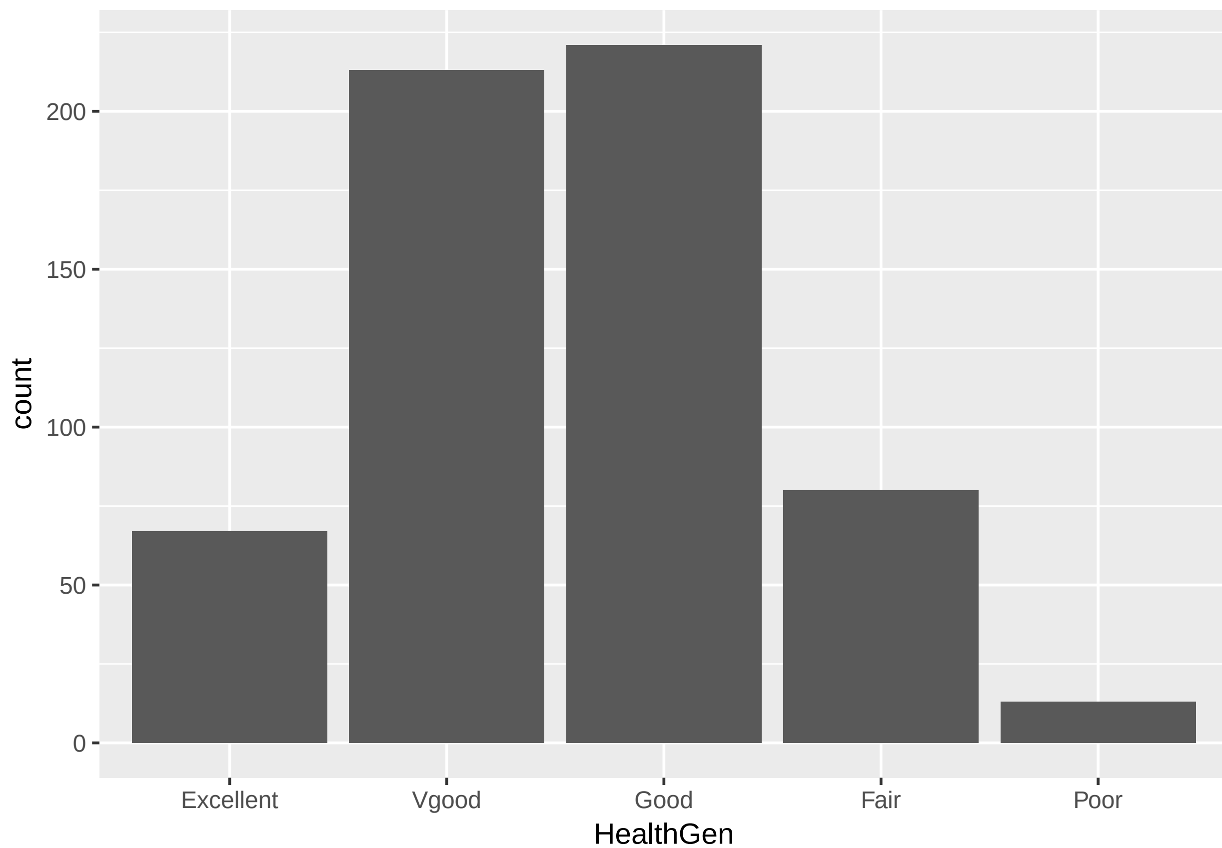
```
.
Excellent     Vgood      Good      Fair      Poor
       67       213       221        80        13
```

The HealthGen data are categorical, which means that summarizing them with averages isn't as appealing as looking at percentages, proportions and rates.

3.9.1 Bar Chart for Categorical Data

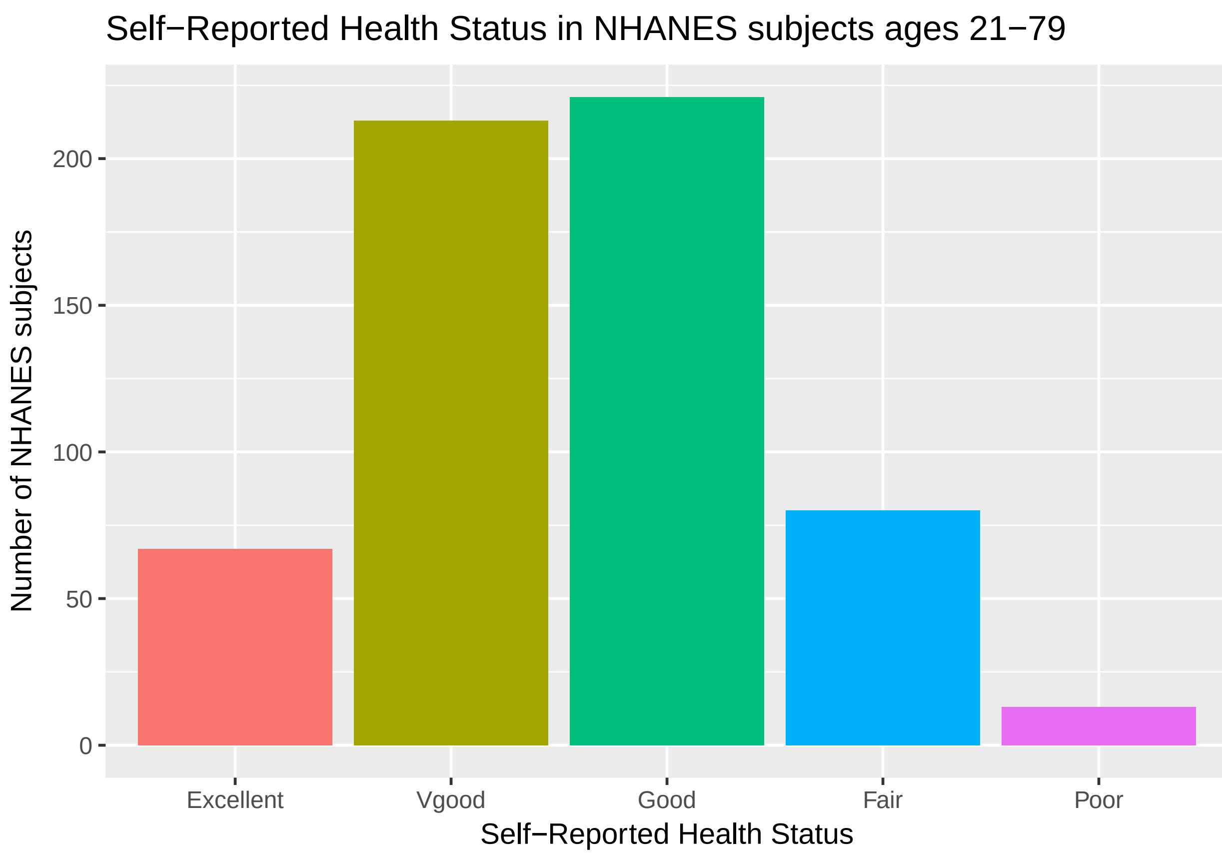
Usually, a **bar chart** is the best choice for a graphing a variable made up of categories.

```
ggplot(data = nh_data_2179, aes(x = HealthGen)) +
    geom_bar()
```



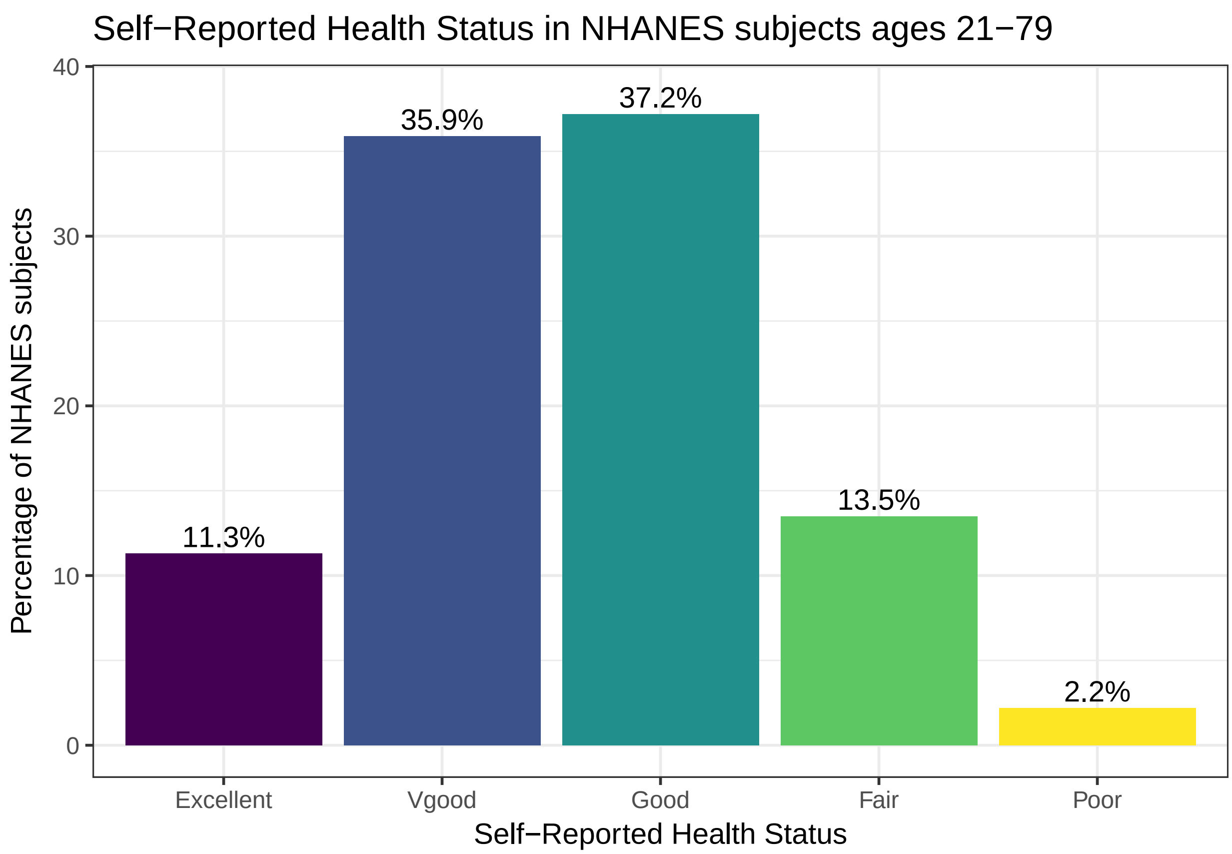
There are lots of things we can do to make this plot fancier.

```
ggplot(data = nh_data_2179, aes(x = HealthGen, fill = HealthGen)) +
    geom_bar() +
    guides(fill = FALSE) +
    labs(x = "Self-Reported Health Status",
         y = "Number of NHANES subjects",
         title = "Self-Reported Health Status in NHANES subjects ages 21-79")
```



Or, we can really go crazy...

```
nh_data_2179 %>%
    count(HealthGen) %>%
    ungroup() %>%
    mutate(pct = round(prop.table(n) * 100, 1)) %>%
    ggplot(aes(x = HealthGen, y = pct, fill = HealthGen)) +
    geom_bar(stat = "identity", position = "dodge") +
    scale_fill_viridis(discrete = TRUE) +
    guides(fill = FALSE) +
    geom_text(aes(y = pct + 1,     # nudge above top of bar
                  label = paste0(pct, '%')),  # prettify
              position = position_dodge(width = .9),
              size = 4) +
    labs(x = "Self-Reported Health Status",
         y = "Percentage of NHANES subjects",
         title = "Self-Reported Health Status in NHANES subjects ages 21-79") +
    theme_bw()
```



3.9.2 Working with Tables

We can add a marginal total, and compare subjects by Gender, as follows…

```
nh_data_2179 %>%
    select(Gender, HealthGen) %>%
    table() %>%
    addmargins()
```

```
        HealthGen
Gender   Excellent Vgood Good Fair Poor Sum
  female        34   107  107   34    8 290
  male          33   106  114   46    5 304
  Sum           67   213  221   80   13 594
```

If we like, we can make this look a little more polished with the `knitr::kable` function…

```
nh_data_2179 %>%
    select(Gender, HealthGen) %>%
    table() %>%
    addmargins() %>%
    knitr::kable()
```

	Excellent	Vgood	Good	Fair	Poor	Sum
female	34	107	107	34	8	290
male	33	106	114	46	5	304
Sum	67	213	221	80	13	594

If we want the proportions of patients within each Gender that fall in each HealthGen category (the row percentages), we can get them, too.

```
nh_data_2179 %>%
    select(Gender, HealthGen) %>%
    table() %>%
    prop.table(.,1) %>%
    knitr::kable()
```

	Excellent	Vgood	Good	Fair	Poor
female	0.1172414	0.3689655	0.3689655	0.1172414	0.0275862
male	0.1085526	0.3486842	0.3750000	0.1513158	0.0164474

To make this a little easier to use, we might consider rounding.

```
nh_data_2179 %>%
    select(Gender, HealthGen) %>%
    table() %>%
    prop.table(.,1) %>%
    round(.,2) %>%
    knitr::kable()
```

	Excellent	Vgood	Good	Fair	Poor
female	0.12	0.37	0.37	0.12	0.03
male	0.11	0.35	0.38	0.15	0.02

Another possibility would be to show the percentages, rather than the proportions (which requires multiplying the proportion by 100.) Note the strange "*" function, which is needed to convince R to multiply each entry by 100 here.

```
nh_data_2179 %>%
    select(Gender, HealthGen) %>%
    table() %>%
    prop.table(.,1) %>%
    "*"(100) %>%
    round(.,2)  %>%
    knitr::kable()
```

	Excellent	Vgood	Good	Fair	Poor
female	11.72	36.90	36.9	11.72	2.76
male	10.86	34.87	37.5	15.13	1.64

And, if we wanted the column percentages, to determine which gender had the higher rate of each HealthGen status level, we can get that by changing the prop.table to calculate 2 (column) proportions, rather than 1 (rows.)

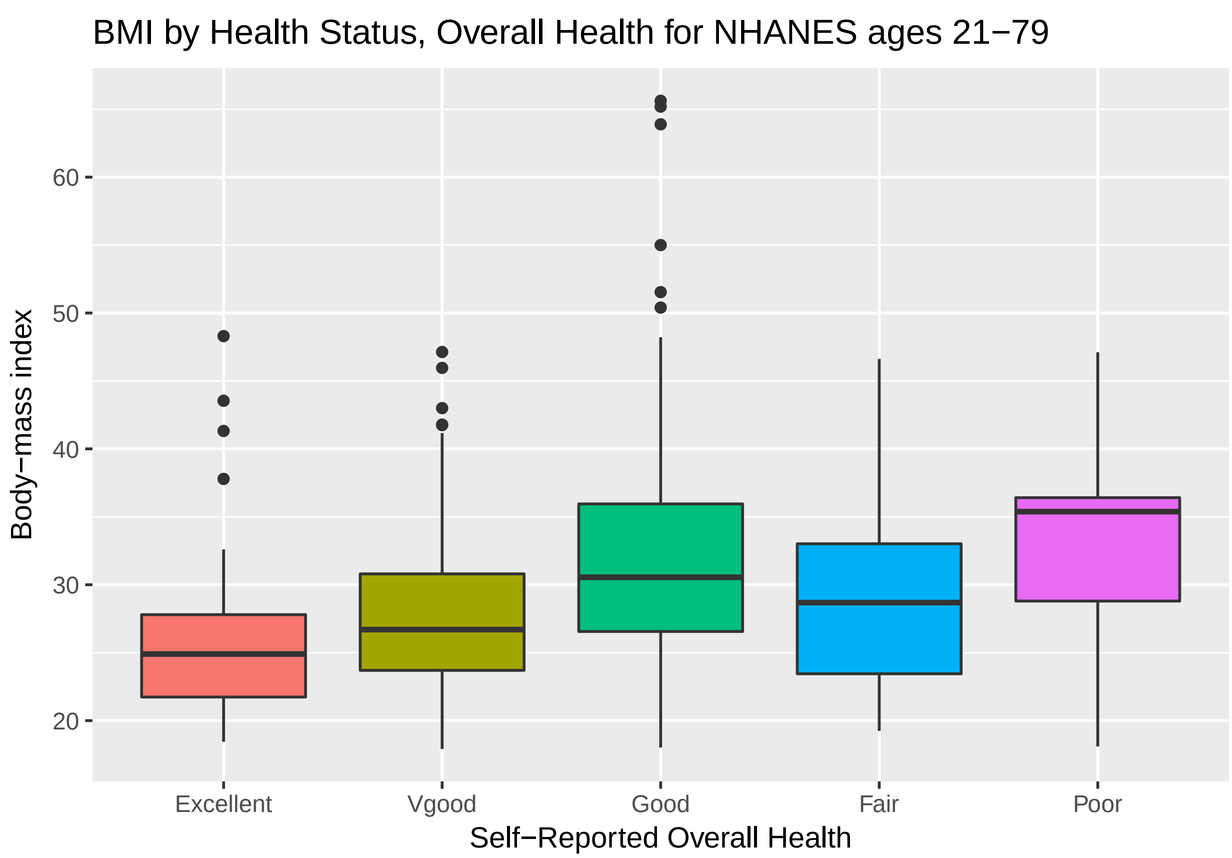
```
nh_data_2179 %>%
    select(Gender, HealthGen) %>%
    table() %>%
    prop.table(.,2) %>%
    "*"(100) %>%
    round(.,2)  %>%
    knitr::kable()
```

	Excellent	Vgood	Good	Fair	Poor
female	50.75	50.23	48.42	42.5	61.54
male	49.25	49.77	51.58	57.5	38.46

3.9.3 BMI by General Health Status

Let's consider now the relationship between self-reported overall health and body-mass index.

```
ggplot(data = nh_data_2179, aes(x = HealthGen, y = BMI, fill = HealthGen)) +
    geom_boxplot() +
    labs(title = "BMI by Health Status, Overall Health for NHANES ages 21-79",
         y = "Body-mass index", x = "Self-Reported Overall Health") +
    guides(fill = FALSE)
```



We can see that not too many people self-identify with the "Poor" health category.

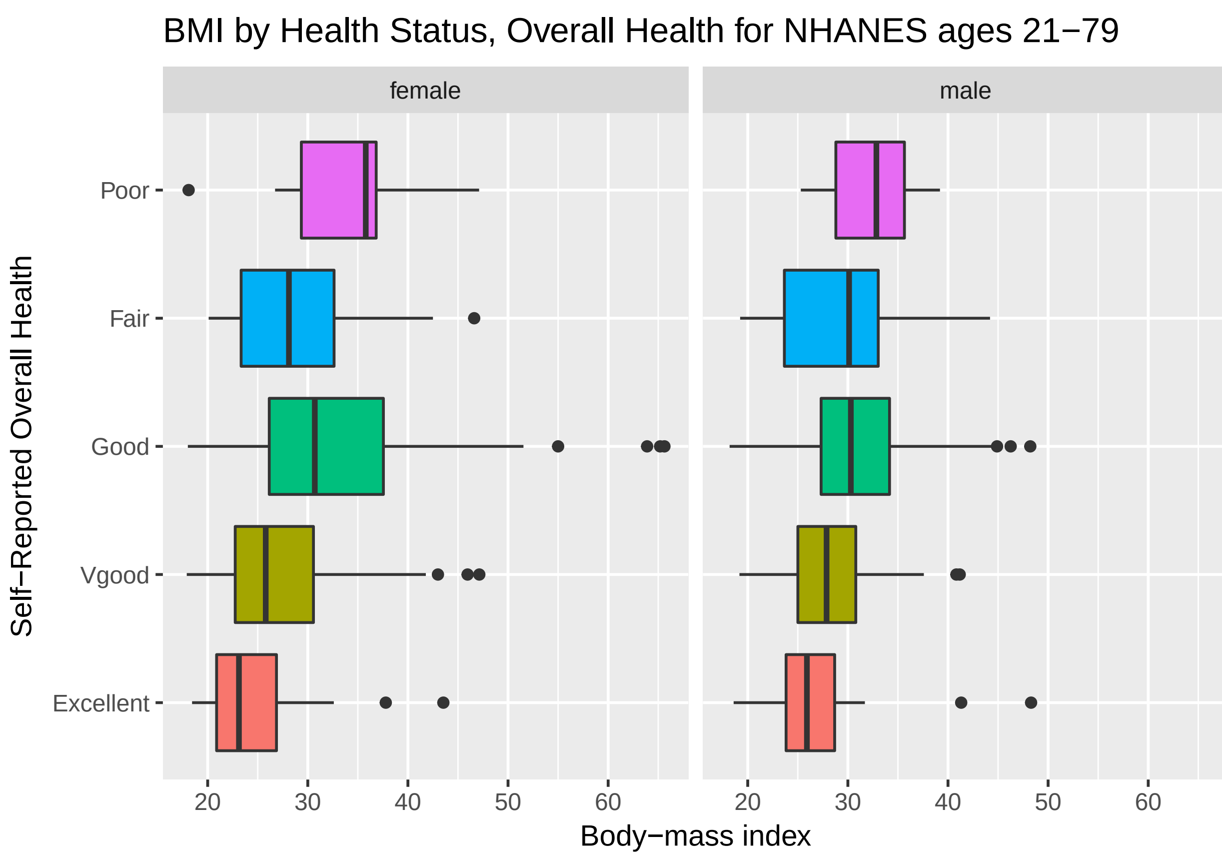
```
nh_data_2179 %>%
    group_by(HealthGen) %>%
    summarize(count = n(), mean(BMI), median(BMI)) %>%
    knitr::kable()
```

HealthGen	count	mean(BMI)	median(BMI)
Excellent	67	25.70060	24.900
Vgood	213	27.55878	26.700
Good	221	32.00321	30.550
Fair	80	29.28663	28.685
Poor	13	33.08154	35.380

3.9.4 BMI by Gender and General Health Status

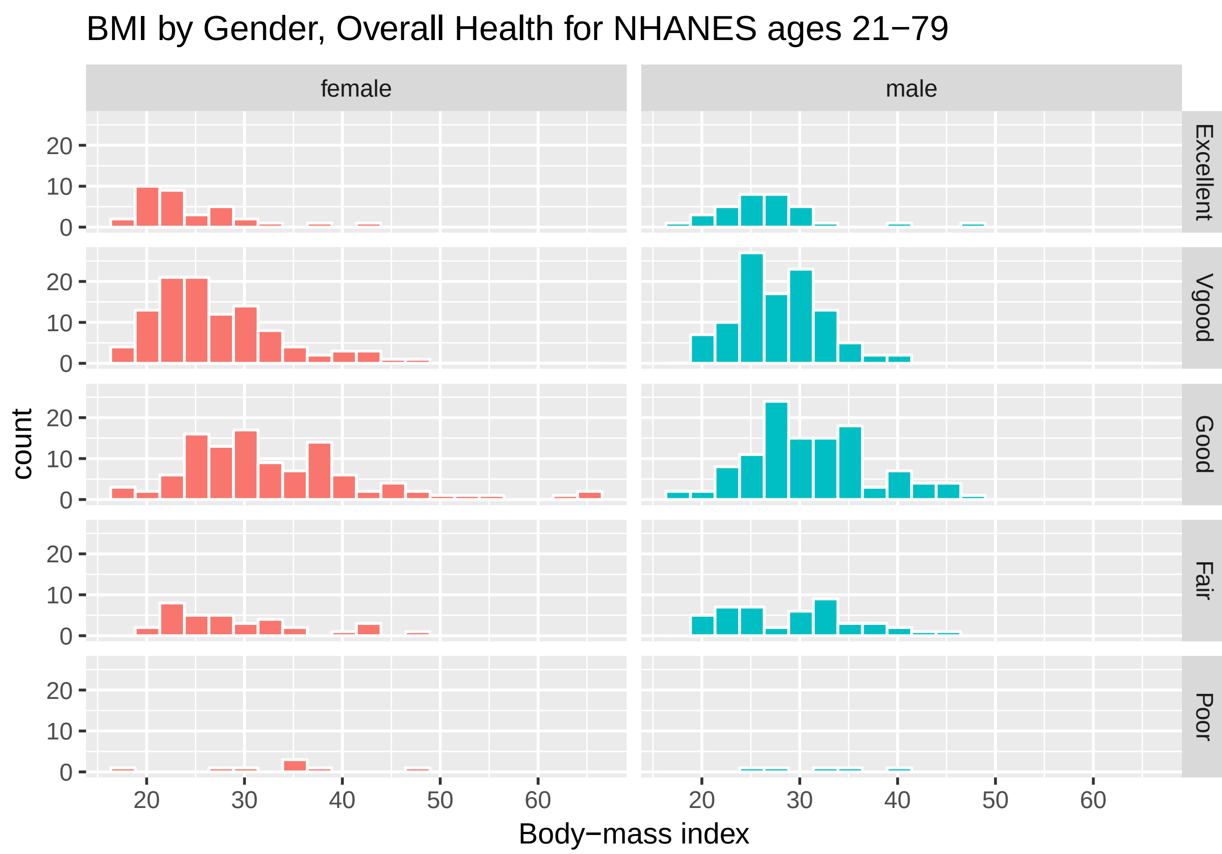
We'll start with two panels of boxplots to try to understand the relationships between BMI, General Health Status and Gender. Note the use of `coord_flip` to rotate the graph 90 degrees.

```
ggplot(data = nh_data_2179, aes(x = HealthGen, y = BMI, fill = HealthGen)) +
    geom_boxplot() +
    labs(title = "BMI by Health Status, Overall Health for NHANES ages 21-79",
         y = "Body-mass index", x = "Self-Reported Overall Health") +
    guides(fill = FALSE) +
    facet_wrap(~ Gender) +
    coord_flip()
```



Here's a plot of faceted histograms, which might be used to address similar questions.

```
ggplot(data = nh_data_2179, aes(x = BMI, fill = Gender)) +
    geom_histogram(color = "white", bins = 20) +
    labs(title = "BMI by Gender, Overall Health for NHANES ages 21-79",
         x = "Body-mass index") +
    guides(fill = FALSE) +
    facet_grid(HealthGen ~ Gender)
```



3.10 Conclusions

This is just a small piece of the toolbox for visualizations that we'll create in this class. Many additional tools are on the way, but the main idea won't change. Using the `ggplot2` package, we can accomplish several critical tasks in creating a visualization, including:

- Identifying (and labeling) the axes and titles
- Identifying a type of `geom` to use, like a point, bar or histogram
- Changing fill, color, shape, size to facilitate comparisons
- Building "small multiples" of plots with faceting

Good data visualizations make it easy to see the data, and `ggplot2`'s tools make it relatively difficult to make a really bad graph.

Chapter 4

Data Structures and Types of Variables

4.1 Data require structure and context

Descriptive statistics are concerned with the presentation, organization and summary of data, as suggested in Norman and Streiner (2014). This includes various methods of organizing and graphing data to get an idea of what those data can tell us.

As Vittinghoff et al. (2012) suggest, the nature of the measurement determines how best to describe it statistically, and the main distinction is between **numerical** and **categorical** variables. Even this is a little tricky - plenty of data can have values that look like numerical values, but are just numerals serving as labels.

As Bock, Velleman, and De Veaux (2004) point out, the truly critical notion, of course, is that data values, no matter what kind, are useless without their contexts. The Five W's (Who, What [and in what units], When, Where, Why, and often How) are just as useful for establishing the context of data as they are in journalism. If you can't answer Who and What, in particular, you don't have any useful information.

In general, each row of a data frame corresponds to an individual (respondent, experimental unit, record, or observation) about whom some characteristics are gathered in columns (and these characteristics may be called variables, factors or data elements.) Every column / variable should have a name that indicates *what* it is measuring, and every row / observation should have a name that indicates *who* is being measured.

4.2 A New NHANES Adult Sample

In previous work, we spent some time with a sample from the National Health and Nutrition Examination. Now, by changing the value of the `set.seed` function which determines the starting place for the random sampling, and changing some other specifications, we'll generate a new sample describing 500 adult subjects who completed the 2011-12 version of the survey when they were between the ages of 21 and 64.

Note also that what is listed in the NHANES data frame as `Gender` should be more correctly referred to as `sex`. `Sex` is a biological feature of an individual, while `Gender` is a social construct. This is an important distinction, so I'll change the name of the variable. I'm also changing the names of three other variables, to create `Race`, `SBP` and `DBP`.

```
library(NHANES) # load the NHANES package/library of functions, data

nh_temp <- NHANES %>%
```

```
    filter(SurveyYr == "2011_12") %>%
    filter(Age >= 21 & Age < 65) %>%
    mutate(Sex = Gender, Race = Race3, SBP = BPSysAve, DBP = BPDiaAve) %>%
    select(ID, Sex, Age, Race, Education, BMI, SBP, DBP, Pulse, PhysActive, Smoke100, SleepTrouble, Heal

set.seed(431002)
# use set.seed to ensure that we all get the same random sample

nh_adults <- sample_n(nh_temp, size = 500)

nh_adults
```

```
# A tibble: 500 x 13
      ID    Sex   Age  Race    Education   BMI   SBP   DBP Pulse
   <int> <fctr> <int> <fctr>      <fctr> <dbl> <int> <int> <int>
 1 64427   male    37  White College Grad  36.5   111    72    56
 2 63788 female    40  White  High School  18.2   115    74   102
 3 66874 female    31  White Some College  27.2    95    52    98
 4 69734   male    26  White College Grad  20.6   137    75    74
 5 70409   male    44  White  High School  29.2   112    71    62
 6 68961 female    64  White College Grad  24.2   123    70    80
 7 62616 female    37  Asian    8th Grade  19.3   109    73    82
 8 70130   male    42  Black  High School  31.2   119    71    62
 9 71218   male    33  White College Grad  27.7   110    67    68
10 69181 female    37  White    8th Grade  25.0   114    74    82
# ... with 490 more rows, and 4 more variables: PhysActive <fctr>,
#   Smoke100 <fctr>, SleepTrouble <fctr>, HealthGen <fctr>
```

The data consists of 500 rows (observations) on 13 variables (columns). Essentially, we have 13 pieces of information on each of 500 adult NHANES subjects who were included in the 2011-12 panel.

4.2.1 Summarizing the Data's Structure

We can identify the number of rows and columns in a data frame or tibble with the `dim` function.

```
dim(nh_adults)
```

```
[1] 500  13
```

The `str` function provides a lot of information about the structure of a data frame or tibble.

```
str(nh_adults)
```

```
Classes 'tbl_df', 'tbl' and 'data.frame':   500 obs. of  13 variables:
 $ ID          : int  64427 63788 66874 69734 70409 68961 62616 70130 71218 69181 ...
 $ Sex         : Factor w/ 2 levels "female","male": 2 1 1 2 2 1 1 2 2 1 ...
 $ Age         : int  37 40 31 26 44 64 37 42 33 37 ...
 $ Race        : Factor w/ 6 levels "Asian","Black",..: 5 5 5 5 5 5 1 2 5 5 ...
 $ Education   : Factor w/ 5 levels "8th Grade","9 - 11th Grade",..: 5 3 4 5 3 5 1 3 5 1 ...
 $ BMI         : num  36.5 18.2 27.2 20.6 29.2 24.2 19.3 31.2 27.7 25 ...
 $ SBP         : int  111 115 95 137 112 123 109 119 110 114 ...
 $ DBP         : int  72 74 52 75 71 70 73 71 67 74 ...
 $ Pulse       : int  56 102 98 74 62 80 82 62 68 82 ...
 $ PhysActive  : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 1 1 2 2 ...
 $ Smoke100    : Factor w/ 2 levels "No","Yes": 1 2 1 1 2 2 1 1 1 2 ...
```

```
 $ SleepTrouble: Factor w/ 2 levels "No","Yes": 1 2 1 1 1 1 1 1 1 2 ...
 $ HealthGen   : Factor w/ 5 levels "Excellent","Vgood",..: 2 3 3 1 3 2 3 3 3 2 ...
```

To see the first few observations, use **head**, and to see the last few, try **tail**…

```r
tail(nh_adults, 5) # shows the last five observations in the data set
```

```
# A tibble: 5 x 13
     ID    Sex   Age    Race      Education   BMI   SBP   DBP Pulse
  <int> <fctr> <int>  <fctr>         <fctr> <dbl> <int> <int> <int>
1 69692   male    50   Black 9 - 11th Grade  22.7   132    82    60
2 66472   male    61   White   Some College  41.3   141    77    62
3 71456   male    21 Mexican 9 - 11th Grade  26.7   113    66    78
4 71420 female    54 Mexican 9 - 11th Grade  32.5   126    69    68
5 63617   male    29   White   College Grad  23.2   105    72    76
# ... with 4 more variables: PhysActive <fctr>, Smoke100 <fctr>,
#   SleepTrouble <fctr>, HealthGen <fctr>
```

4.2.2 What are the variables?

The variables we have collected are described in the brief table below[1].

Variable	Description	Sample Values
ID	a numerical code identifying the subject	64427, 63788
Sex	sex of subject (2 levels)	male, female
Age	age (years) at screening of subject	37, 40
Race	reported race of subject (6 levels)	White, Asian
Education	educational level of subject (5 levels)	College Grad, High School
BMI	body-mass index, in kg/m^2	36.5, 18.2
SBP	systolic blood pressure in mm Hg	111, 115
DBP	diastolic blood pressure in mm Hg	72, 74
Pulse	60 second pulse rate in beats per minute	56, 102
PhysActive	Moderate or vigorous-intensity sports?	Yes, No
Smoke100	Smoked at least 100 cigarettes lifetime?	Yes, No
SleepTrouble	Told a doctor they have trouble sleeping?	Yes, No
HealthGen	Self-report general health rating (5 lev.)	Vgood, Good

The levels for the multi-categorical variables are:

- **Race**: Mexican, Hispanic, White, Black, Asian, or Other.
- **Education**: 8th Grade, 9 - 11th Grade, High School, Some College, or College Grad.
- **HealthGen**: Excellent, Vgood, Good, Fair or Poor.

4.3 Types of Variables

4.3.1 Quantitative Variables

Variables recorded in numbers that we use as numbers are called **quantitative**. Familiar examples include incomes, heights, weights, ages, distances, times, and counts. All quantitative variables have measurement

[1] Descriptions are adapted from the ?NHANES help file. Remember that what NHANES lists as Gender is captured here as Sex, and similarly Race3, BPSysAve and BPDiaAve from NHANES are here listed as Race, SBP and DBP.

units, which tell you how the quantitative variable was measured. Without units (like miles per hour, angstroms, yen or degrees Celsius) the values of a quantitative variable have no meaning.

- It does little good to be promised a salary of 80,000 a year if you don't know whether it will be paid in Euros, dollars, yen or Estonian kroon.
- You might be surprised to see someone whose age is 72 listed in a database on childhood diseases until you find out that age is measured in months.
- Often just seeking the units can reveal a variable whose definition is challenging - just how do we measure "friendliness", or "success," for example.
- Quantitative variables may also be classified by whether they are **continuous** or can only take on a **discrete** set of values. Continuous data may take on any value, within a defined range. Suppose we are measuring height. While height is really continuous, our measuring stick usually only lets us measure with a certain degree of precision. If our measurements are only trustworthy to the nearest centimeter with the ruler we have, we might describe them as discrete measures. But we could always get a more precise ruler. The measurement divisions we make in moving from a continuous concept to a discrete measurement are usually fairly arbitrary. Another way to think of this, if you enjoy music, is that, as suggested in Norman and Streiner (2014), a piano is a *discrete* instrument, but a violin is a *continuous* one, enabling finer distinctions between notes than the piano is capable of making. Sometimes the distinction between continuous and discrete is important, but usually, it's not.
  - The `nh_adults` data includes several quantitative variables, specifically Age, BMI, SBP, DBP and Pulse.
  - We know these are quantitative because they have units: Age in years, BMI in kg/m^2, the BP measurements in mm Hg, and Pulse in beats per minute.
  - Depending on the context, we would likely treat most of these as *discrete* given that are measurements are fairly crude (this is certainly true for Age, measured in years) although BMI is probably *continuous* in most settings, even though it is a function of two other measures (Height and Weight) which are rounded off to integer numbers of centimeters and kilograms, respectively.
- It is also possible to separate out quantitative variables into **ratio** variables or **interval** variables. An interval variable has equal distances between values, but the zero point is arbitrary. A ratio variable has equal intervals between values, and a meaningful zero point. For example, weight is an example of a ratio variable, while IQ is an example of an interval variable. We all know what zero weight is. An intelligence score like IQ is a different matter. We say that the average IQ is 100, but that's only by convention. We could just as easily have decided to add 400 to every IQ value and make the average 500 instead. Because IQ's intervals are equal, the difference between and IQ of 70 and an IQ of 80 is the same as the difference between 120 and 130. However, an IQ of 100 is not twice as high as an IQ of 50. The point is that if the zero point is artificial and moveable, then the differences between numbers are meaningful but the ratios between them are not. On the other hand, most lab test values are ratio variables, as are physical characteristics like height and weight. A person who weighs 100 kg is twice as heavy as one who weighs 50 kg; even when we convert kg to pounds, this is still true. For the most part, we can treat and analyze interval or ratio variables the same way.
  - Each of the quantitative variables in our `nh_adults` data can be thought of as ratio variables.
- Quantitative variables lend themselves to many of the summaries we will discuss, like means, quantiles, and our various measures of spread, like the standard deviation or inter-quartile range. They also have at least a chance to follow the Normal distribution.

4.3.2 Qualitative (Categorical) Variables

Qualitative or categorical variables consist of names of categories. These names may be numerical, but the numbers (or names) are simply codes to identify the groups or categories into which the individuals are divided. Categorical variables with two categories, like yes or no, up or down, or, more generally, 1 and 0, are called **binary** variables. Those with more than two-categories are sometimes called **multi-categorical** variables.

- When the categories included in a variable are merely names, and come in no particular order, we sometimes call them **nominal** variables. The most important summary of such a variable is usually a table of frequencies, and the mode becomes an important single summary, while the mean and median are essentially useless.
  - In the nh_adults data, Race is clearly a nominal variable with multiple unordered categories.
- The alternative categorical variable (where order matters) is called **ordinal**, and includes variables that are sometimes thought of as falling right in between quantitative and qualitative variables.
  - Examples of ordinal multi-categorical variables in the `nh_adults` data include the Education and HealthGen variables.
  - Answers to questions like "How is your overall physical health?" with available responses Excellent, Very Good, Good, Fair or Poor, which are often coded as 1-5, certainly provide a perceived *order*, but a group of people with average health status 4 (Very Good) is not necessarily twice as healthy as a group with average health status of 2 (Fair).
- Sometimes we treat the values from ordinal variables as sufficiently scaled to permit us to use quantitative approaches like means, quantiles, and standard deviations to summarize and model the results, and at other times, we'll treat ordinal variables as if they were nominal, with tables and percentages our primary tools.
- Note that all binary variables may be treated as ordinal, or nominal.
  - Binary variables in the `nh_adults` data include Sex, PhysActive, Smoke100, SleepTrouble. Each can be thought of as either ordinal or nominal.

Lots of variables may be treated as either quantitative or qualitative, depending on how we use them. For instance, we usually think of age as a quantitative variable, but if we simply use age to make the distinction between "child" and "adult" then we are using it to describe categorical information. Just because your variable's values are numbers, don't assume that the information provided is quantitative.

Chapter 5

Summarizing Quantitative Variables

Most numerical summaries that might be new to you are applied most appropriately to quantitative variables. The measures that will interest us relate to:

- the **center** of our distribution,
- the **spread** of our distribution, and
- the **shape** of our distribution.

5.1 The `summary` function for Quantitative data

R provides a small sampling of numerical summaries with the `summary` function, for instance.

```
nh_adults %>%
  select(Age, BMI, SBP, DBP, Pulse) %>%
  summary()
```

```
      Age             BMI             SBP             DBP        
 Min.   :21.0   Min.   :17.80   Min.   : 84.0   Min.   : 19.00  
 1st Qu.:31.0   1st Qu.:24.20   1st Qu.:109.0   1st Qu.: 65.00  
 Median :42.0   Median :27.70   Median :118.0   Median : 72.00  
 Mean   :42.1   Mean   :28.73   Mean   :118.6   Mean   : 72.25  
 3rd Qu.:53.0   3rd Qu.:32.10   3rd Qu.:127.0   3rd Qu.: 79.00  
 Max.   :64.0   Max.   :69.00   Max.   :202.0   Max.   :105.00  
                NA's   :3       NA's   :15      NA's   :15      
     Pulse       
 Min.   : 46.00  
 1st Qu.: 64.00  
 Median : 72.00  
 Mean   : 72.96  
 3rd Qu.: 80.00  
 Max.   :120.00  
 NA's   :15      
```

This basic summary includes a set of five **quantiles**[1], plus the sample's **mean**.

- `Min.` = the **minimum** value for each variable, so, for example, the youngest subject's Age was 21.
- `1st Qu.` = the **first quartile** (25^{th} percentile) for each variable - for example, 25% of the subjects were Age 31 or younger.

[1]The quantiles (sometimes referred to as percentiles) can also be summarized with a boxplot.

- `Median` = the **median** (50th percentile) - half of the subjects were Age 42 or younger.
- `Mean` = the **mean**, usually what one means by an *average* - the sum of the Ages divided by 500 is 42.1,
- `3rd Qu.` = the **third quartile** (75th percentile) - 25% of the subjects were Age 53 or older.
- `Max.` = the **maximum** value for each variable, so the oldest subject was Age 64.

The summary also specifies the number of missing values for each variable. Here, we are missing 3 of the BMI values, for example.

5.2 Measuring the Center of a Distribution

5.2.1 The Mean and The Median

The **mean** and **median** are the most commonly used measures of the center of a distribution for a quantitative variable. The median is the more generally useful value, as it is relevant even if the data have a shape that is not symmetric. We might also collect the **sum** of the observations, and the **count** of the number of observations, usually symbolized with n.

For variables without missing values, like `Age`, this is pretty straightforward.

```
nh_adults %>%
    summarize(n = n(), Mean = mean(Age), Median = median(Age), Sum = sum(Age))
```

```
# A tibble: 1 x 4
      n   Mean Median   Sum
  <int>  <dbl>  <dbl> <int>
1   500 42.102     42 21051
```

And again, the Mean is just the Sum (21051), divided by the number of non-missing values of Age (500), or 42.102.

The Median is the middle value when the data are sorted in order. When we have an odd number of values, this is sufficient. When we have an even number, as in this case, we take the mean of the two middle values. We could sort and list all 500 Ages, if we wanted to do so.

```
nh_adults %>% select(Age) %>%
    arrange(Age)
```

```
# A tibble: 500 x 1
     Age
   <int>
 1    21
 2    21
 3    21
 4    21
 5    21
 6    21
 7    21
 8    21
 9    21
10    21
# ... with 490 more rows
```

But this data set figures we don't want to output more than 10 observations to a table like this.

If we really want to see all of the data, we can use `View(nh_adults)` to get a spreadsheet-style presentation, or use the `sort` command…

```
sort(nh_adults$Age)
```

```
  [1] 21 21 21 21 21 21 21 21 21 21 21 21 22 22 22 22 22 22 22 22 22 22 23
 [24] 23 23 23 23 23 23 23 23 23 23 23 24 24 24 24 24 24 24 24 24 24 24 24
 [47] 24 25 25 25 25 25 25 25 25 25 25 25 25 26 26 26 26 26 26 26 26 26 26
 [70] 26 26 27 27 27 27 27 27 27 27 27 27 27 27 27 28 28 28 28 28 28 28 28
 [93] 28 28 28 28 28 28 28 28 29 29 29 29 29 29 29 29 29 29 29 30 30 30 30
[116] 30 30 30 30 30 30 30 30 31 31 31 31 31 31 31 31 31 31 31 31 32 32 32
[139] 32 32 32 32 32 32 32 32 33 33 33 33 33 33 33 33 33 33 33 34 34 34 34
[162] 34 34 34 34 35 35 35 35 36 36 36 36 36 36 36 36 36 37 37 37 37 37 37
[185] 37 37 37 37 37 37 37 37 37 37 38 38 38 38 38 38 38 38 38 38 39 39 39
[208] 39 39 39 39 39 39 39 39 39 39 40 40 40 40 40 40 40 40 40 40 41 41 41
[231] 41 41 41 41 41 41 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 43
[254] 43 43 43 43 43 43 43 43 43 43 43 44 44 44 44 44 44 44 44 44 44 44 45
[277] 45 45 45 45 45 45 45 46 46 46 46 46 46 46 46 46 46 46 46 46 47 47 47
[300] 47 47 47 47 47 47 48 48 48 48 48 48 48 48 48 48 48 49 49 49 49 49 49
[323] 49 49 49 49 49 49 49 49 49 50 50 50 50 50 50 50 50 50 50 50 50 50 50
[346] 50 50 50 50 51 51 51 51 51 51 51 51 51 51 51 51 52 52 52 52 52 52 52
[369] 52 52 52 53 53 53 53 53 53 53 53 53 53 53 53 53 54 54 54 54 54 54 54
[392] 54 54 54 54 55 55 55 55 55 55 56 56 56 56 56 56 56 56 56 56 56 56 56
[415] 56 56 56 56 56 56 57 57 57 57 57 57 57 57 57 57 57 57 57 58 58 58 58
[438] 58 58 58 58 58 58 58 59 59 59 59 59 59 59 59 59 59 59 60 60 60 60 60
[461] 60 60 60 60 60 60 60 61 61 61 61 61 61 61 61 61 61 61 61 62 62 62 62
[484] 62 62 62 63 63 63 63 63 64 64 64 64 64 64 64 64 64
```

Again, to find the median, we would take the mean of the middle two observations in this sorted data set. That would be the 250^{th} and 251^{st} largest Ages.

```
sort(nh_adults$Age)[250:251]
```

```
[1] 42 42
```

5.2.2 Dealing with Missingness

When calculating a mean, you may be tempted to try something like this...

```
nh_adults %>%
    summarize(mean(Pulse), median(Pulse))
```

```
# A tibble: 1 x 2
  `mean(Pulse)` `median(Pulse)`
          <dbl>           <int>
1            NA              NA
```

This fails because we have some missing values in the Pulse data. We can address this by either omitting the data with missing values before we run the summarize function, or tell the mean and median summary functions to remove missing values[2].

```
nh_adults %>%
    filter(complete.cases(Pulse)) %>%
    summarize(count = n(), mean(Pulse), median(Pulse))
```

```
# A tibble: 1 x 3
  count `mean(Pulse)` `median(Pulse)`
  <int>         <dbl>           <int>
```

[2] We could also use `!is.na` in place of `complete.cases` to accomplish the same thing.

```
1   485       72.9567              72
```

Or, we could tell the summary functions themselves to remove NA values.

```
nh_adults %>%
    summarize(mean(Pulse, na.rm=TRUE), median(Pulse, na.rm=TRUE))
```

```
# A tibble: 1 x 2
  `mean(Pulse, na.rm = TRUE)` `median(Pulse, na.rm = TRUE)`
                        <dbl>                         <int>
1                     72.9567                            72
```

While we eventually discuss the importance of **imputation** when dealing with missing data, this doesn't apply to providing descriptive summaries of actual, observed values.

5.2.3 The Mode of a Quantitative Variable

One other less common measure of the center of a quantitative variable's distribution is its most frequently observed value, referred to as the **mode**. This measure is only appropriate for discrete variables, be they quantitative or categorical. To find the mode, we usually tabulate the data, and then sort by the counts of the numbers of observations.

```
nh_adults %>%
    group_by(Age) %>%
    summarize(count = n()) %>%
    arrange(desc(count))
```

```
# A tibble: 44 x 2
     Age count
   <int> <int>
 1    56    19
 2    50    18
 3    28    16
 4    37    16
 5    42    16
 6    49    15
 7    24    13
 8    27    13
 9    39    13
10    46    13
# ... with 34 more rows
```

Note the use of three different "verbs" in our function there - for more explanation of this strategy, visit Grolemund and Wickham (2017).

As an alternative, the `modeest` package's `mfv` function calculates the sample mode (or most frequent value).[3]

5.3 Measuring the Spread of a Distribution

Statistics is all about variation, so spread or dispersion is an important fundamental concept in statistics. Measures of spread like the inter-quartile range and range (maximum - minimum) can help us understand and compare data sets. If the values in the data are close to the center, the spread will be small. If many of the values in the data are scattered far away from the center, the spread will be large.

[3]See the documentation for the `modeest` package's `mlv` function to look at other definitions of the mode.

5.3.1 The Range and the Interquartile Range (IQR)

The **range** of a quantitative variable is sometimes interpreted as the difference between the maximum and the minimum, even though R presents the actual minimum and maximum values when you ask for a range…

```
nh_adults %>%
    select(Age) %>%
    range()
```

```
[1] 21 64
```

And, for a variable with missing values, we can use…

```
nh_adults %>%
    select(BMI) %>%
    range(., na.rm=TRUE)
```

```
[1] 17.8 69.0
```

A more interesting and useful statistic is the **inter-quartile range**, or IQR, which is the range of the middle half of the distribution, calculated by subtracting the 25$^{\text{th}}$ percentile value from the 75$^{\text{th}}$ percentile value.

```
nh_adults %>%
    summarize(IQR(Age), quantile(Age, 0.25), quantile(Age, 0.75))
```

```
# A tibble: 1 x 3
  `IQR(Age)` `quantile(Age, 0.25)` `quantile(Age, 0.75)`
       <dbl>                 <dbl>                 <dbl>
1         22                    31                    53
```

We can calculate the range and IQR nicely from the summary information on quantiles, of course:

```
nh_adults %>%
    select(Age, BMI, SBP, DBP, Pulse) %>%
    summary()
```

```
      Age             BMI             SBP             DBP        
 Min.   :21.0   Min.   :17.80   Min.   : 84.0   Min.   : 19.00  
 1st Qu.:31.0   1st Qu.:24.20   1st Qu.:109.0   1st Qu.: 65.00  
 Median :42.0   Median :27.70   Median :118.0   Median : 72.00  
 Mean   :42.1   Mean   :28.73   Mean   :118.6   Mean   : 72.25  
 3rd Qu.:53.0   3rd Qu.:32.10   3rd Qu.:127.0   3rd Qu.: 79.00  
 Max.   :64.0   Max.   :69.00   Max.   :202.0   Max.   :105.00  
                NA's   :3       NA's   :15      NA's   :15      
     Pulse       
 Min.   : 46.00  
 1st Qu.: 64.00  
 Median : 72.00  
 Mean   : 72.96  
 3rd Qu.: 80.00  
 Max.   :120.00  
 NA's   :15      
```

5.3.2 The Variance and the Standard Deviation

The IQR is always a reasonable summary of spread, just as the median is always a reasonable summary of the center of a distribution. Yet, most people are inclined to summarize a batch of data using two numbers: the

mean and the **standard deviation**. This is really only a sensible thing to do if you are willing to assume the data follow a Normal distribution: a bell-shaped, symmetric distribution without substantial outliers.

But **most data do not (even approximately) follow a Normal distribution**. Summarizing by the median and quartiles (25th and 75th percentiles) is much more robust, explaining R's emphasis on them.

5.3.3 Obtaining the Variance and Standard Deviation in R

Here are the variances of the quantitative variables in the `nh_adults` data. Note the need to include `na.rm = TRUE` to deal with the missing values in some variables.

```
nh_adults %>%
    select(Age, BMI, SBP, DBP, Pulse) %>%
    summarize_all(var, na.rm = TRUE)
```

```
# A tibble: 1 x 5
      Age      BMI      SBP      DBP    Pulse
    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
1 157.178 42.09176 234.1718 117.3219 131.6613
```

And here are the standard deviations of those same variables.

```
nh_adults %>%
    select(Age, BMI, SBP, DBP, Pulse) %>%
    summarize_all(sd, na.rm = TRUE)
```

```
# A tibble: 1 x 5
       Age      BMI      SBP      DBP    Pulse
     <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
1 12.53706 6.487816 15.30267 10.83152 11.47438
```

5.3.4 Defining the Variance and Standard Deviation

Bock, Velleman, and De Veaux (2004) have lots of useful thoughts here, which are lightly edited here.

In thinking about spread, we might consider how far each data value is from the mean. Such a difference is called a *deviation*. We could just average the deviations, but the positive and negative differences always cancel out, leaving an average deviation of zero, so that's not helpful. Instead, we *square* each deviation to obtain non-negative values, and to emphasize larger differences. When we add up these squared deviations and find their mean (almost), this yields the **variance**.

$$\text{Variance} = s^2 = \frac{\Sigma(y - \bar{y})^2}{n-1}$$

Why almost? It would be the mean of the squared deviations only if we divided the sum by n, but instead we divide by $n-1$ because doing so produces an estimate of the true (population) variance that is *unbiased*[4]. If you're looking for a more intuitive explanation, this Stack Exchange link awaits your attention.

- To return to the original units of measurement, we take the square root of s^2, and instead work with s, the **standard deviation**.

$$\text{Standard Deviation} = s = \sqrt{\frac{\Sigma(y - \bar{y})^2}{n-1}}$$

[4] When we divide by n-1 as we calculate the sample variance, the average of the sample variances for all possible samples is equal to the population variance. If we instead divided by n, the average sample variance across all possible samples would be a little smaller than the population variance.

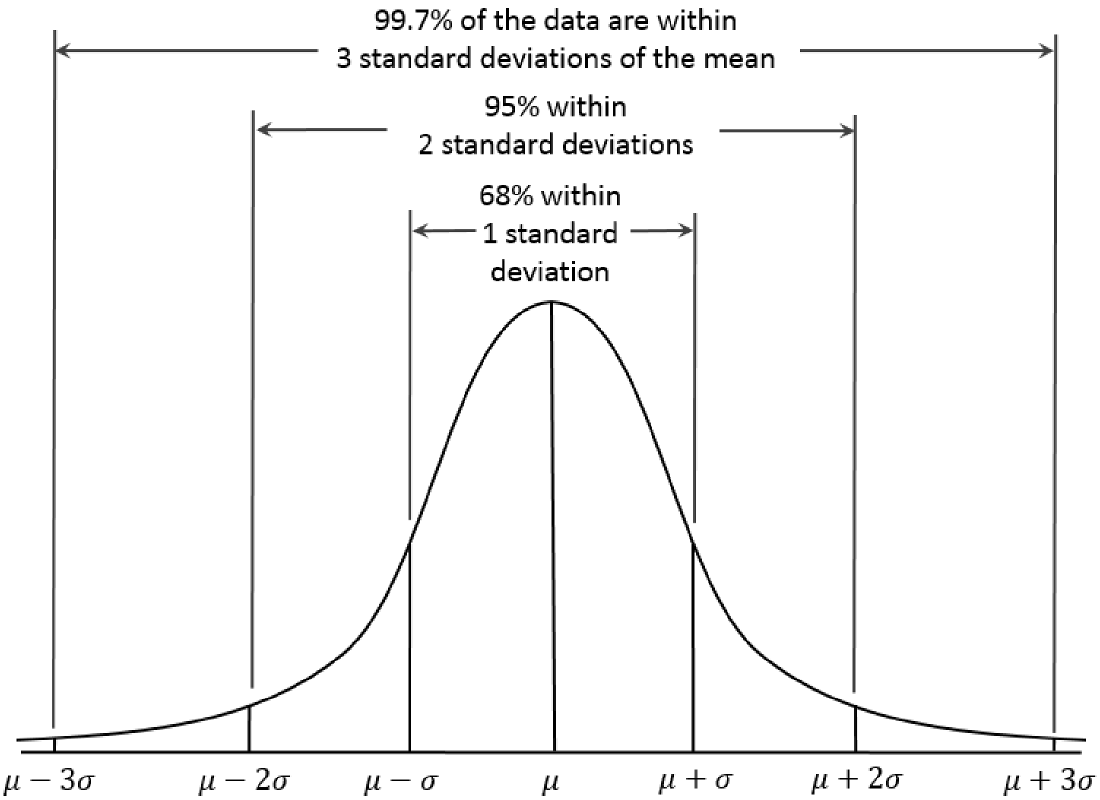


Figure 5.1: The Normal Distribution and the Empirical Rule

5.3.5 Empirical Rule Interpretation of the Standard Deviation

For a set of measurements that follow a Normal distribution, the interval:

- Mean ± Standard Deviation contains approximately 68% of the measurements;
- Mean ± 2(Standard Deviation) contains approximately 95% of the measurements;
- Mean ± 3(Standard Deviation) contains approximately all (99.7%) of the measurements.

We often refer to the population or process mean of a distribution with μ and the standard deviation with σ, leading to the Figure below.

But if the data are not from an approximately Normal distribution, then this Empirical Rule is less helpful.

5.3.6 Chebyshev's Inequality: One Interpretation of the Standard Deviation

Chebyshev's Inequality tells us that for any distribution, regardless of its relationship to a Normal distribution, no more than $1/\mathrm{k}^2$ of the distribution's values can lie more than k standard deviations from the mean. This implies, for instance, that for **any** distribution, at least 75% of the values must lie within two standard deviations of the mean, and at least 89% must lie within three standard deviations of the mean.

Again, most data sets do not follow a Normal distribution. We'll return to this notion soon. But first, let's try to draw some pictures that let us get a better understanding of the distribution of our data.

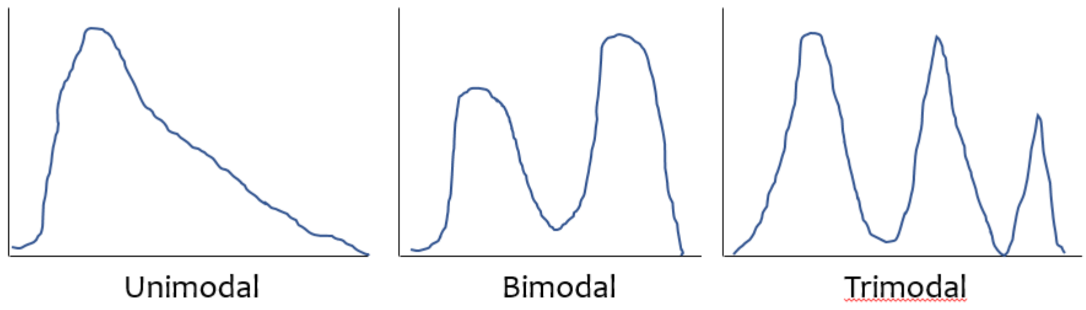


Figure 5.2: Unimodal and Multimodal Sketches

5.4 Measuring the Shape of a Distribution

When considering the shape of a distribution, one is often interested in three key points.

- The number of modes in the distribution, which I always assess through plotting the data.
- The **skewness**, or symmetry that is present, which I typically assess by looking at a plot of the distribution of the data, but if required to, will summarize with a non-parametric measure of **skewness**.
- The **kurtosis**, or heavy-tailedness (outlier-proneness) that is present, usually in comparison to a Normal distribution. Again, this is something I nearly inevitably assess graphically, but there are measures.

A Normal distribution has a single mode, is symmetric and, naturally, is neither heavy-tailed or light-tailed as compared to a Normal distribution (we call this mesokurtic).

5.4.1 Multimodal vs. Unimodal distributions

A unimodal distribution, on some level, is straightforward. It is a distribution with a single mode, or "peak" in the distribution. Such a distribution may be skewed or symmetric, light-tailed or heavy-tailed. We usually describe as multimodal distributions like the two on the right below, which have multiple local maxima, even though they have just a single global maximum peak.

Truly multimodal distributions are usually described that way in terms of shape. For unimodal distributions, skewness and kurtosis become useful ideas.

5.4.2 Skew

Whether or not a distribution is approximately symmetric is an important consideration in describing its shape. Graphical assessments are always most useful in this setting, particularly for unimodal data. My favorite measure of skew, or skewness if the data have a single mode, is:

$$skew_1 = \frac{\text{mean} - \text{median}}{\text{standard deviation}}$$

- Symmetric distributions generally show values of $skew_1$ near zero. If the distribution is actually symmetric, the mean should be equal to the median.
- Distributions with $skew_1$ values above 0.2 in absolute value generally indicate meaningful skew.
- Positive skew (mean > median if the data are unimodal) is also referred to as *right skew*.
- Negative skew (mean < median if the data are unimodal) is referred to as *left skew*.

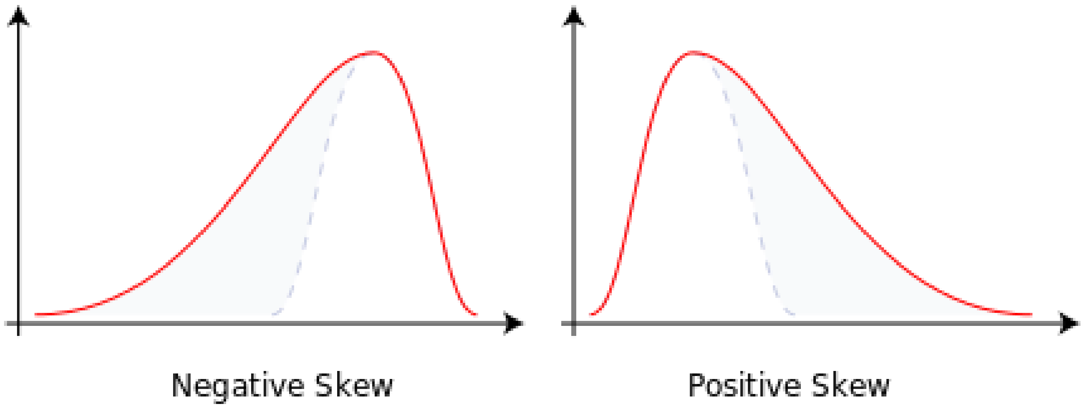


Figure 5.3: Negative (Left) Skew and Positive (Right) Skew

5.4.3 Kurtosis

When we have a unimodal distribution that is symmetric, we will often be interested in the behavior of the tails of the distribution, as compared to a Normal distribution with the same mean and standard deviation. High values of kurtosis measures (and there are several) indicate data which has extreme outliers, or is heavy-tailed.

- A mesokurtic distribution has similar tail behavior to what we would expect from a Normal distribution.
- A leptokurtic distribution is a thinner distribution, with lighter tails (fewer observations far from the center) than we'd expect from a Normal distribution.
- A platykurtic distribution is a flatter distribution, with heavier tails (more observations far from the center) than we'd expect from a Normal distribution.

Graphical tools are in most cases the best way to identify issues related to kurtosis.

5.5 More Detailed Numerical Summaries for Quantitative Variables

5.5.1 `favstats` in the `mosaic` package

The `favstats` function adds the standard deviation, and counts of overall and missing observations to our usual `summary` for a continuous variable. Let's look at systolic blood pressure, because we haven't yet.

```
mosaic::favstats(~ SBP, data = nh_adults)
```

```
 min  Q1 median  Q3 max     mean       sd   n missing
  84 109    118 127 202 118.5918 15.30267 485      15
```

We could, of course, duplicate these results with a rather lengthy set of `summarize` pieces...

```
nh_adults %>%
    filter(complete.cases(SBP)) %>%
    summarize(min = min(SBP), Q1 = quantile(SBP, 0.25), median = median(SBP),
              Q3 = quantile(SBP, 0.75), max = max(SBP),
              mean = mean(SBP), sd = sd(SBP), n = n(), missing = sum(is.na(SBP)))
```

```
# A tibble: 1 x 9
    min    Q1 median    Q3   max     mean       sd     n missing
```

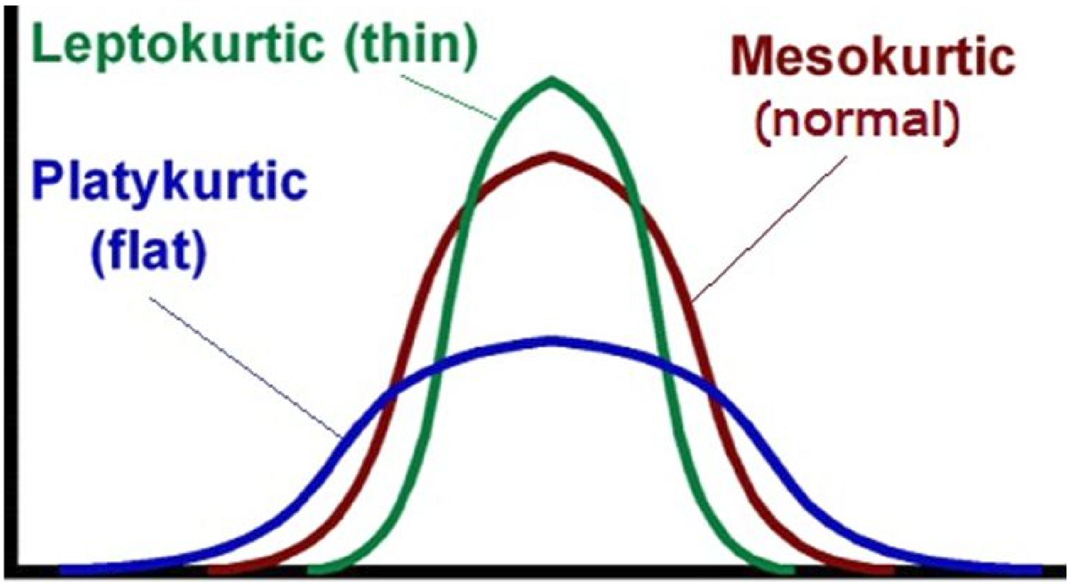


Figure 5.4: The Impact of Kurtosis

```
  <dbl> <dbl>  <int> <dbl> <dbl>    <dbl>    <dbl> <int>    <int>
1    84   109    118   127   202 118.5918 15.30267   485        0
```

The somewhat unusual structure of `favstats` (complete with an easy to forget `~`) is actually helpful. It allows you to look at some interesting grouping approaches, like this:

```
mosaic::favstats(SBP ~ Education, data = nh_adults)
```

```
       Education min  Q1 median     Q3 max     mean       sd   n missing
1      8th Grade  95 109    122 126.00 147 119.0952 14.06735  21       3
2 9 - 11th Grade 100 111    115 126.00 152 118.4386 11.96277  57       0
3    High School  89 109    120 128.75 202 121.3077 19.71835  78       3
4   Some College  85 110    118 128.00 163 119.0268 14.60514 149       4
5   College Grad  84 108    116 124.00 172 117.0444 14.72611 180       5
```

Of course, we could accomplish the same comparison with `dplyr` commands, too, but the `favstats` approach has much to offer.

```
nh_adults %>%
    filter(complete.cases(SBP, Education)) %>%
    group_by(Education) %>%
    summarize(min = min(SBP), Q1 = quantile(SBP, 0.25), median = median(SBP),
              Q3 = quantile(SBP, 0.75), max = max(SBP),
              mean = mean(SBP), sd = sd(SBP), n = n(), missing = sum(is.na(SBP)))
```

```
# A tibble: 5 x 10
       Education   min    Q1 median     Q3   max     mean       sd     n
          <fctr> <dbl> <dbl>  <dbl>  <dbl> <dbl>    <dbl>    <dbl> <int>
1      8th Grade    95   109    122 126.00   147 119.0952 14.06735    21
2 9 - 11th Grade   100   111    115 126.00   152 118.4386 11.96277    57
3    High School    89   109    120 128.75   202 121.3077 19.71835    78
4   Some College    85   110    118 128.00   163 119.0268 14.60514   149
```

```
5   College Grad    84    108    116 124.00    172 117.0444 14.72611   180
# ... with 1 more variables: missing <int>
```

5.5.2 describe in the psych package

The `psych` package has a more detailed list of numerical summaries for quantitative variables that looks us look at a group of observations at once.

```
psych::describe(nh_adults %>% select(Age, BMI, SBP, DBP, Pulse))
```

```
      vars   n   mean    sd median trimmed   mad  min max range  skew
Age      1 500  42.10 12.54   42.0   42.11 16.31 21.0  64  43.0 -0.03
BMI      2 497  28.73  6.49   27.7   28.15  5.78 17.8  69  51.2  1.33
SBP      3 485 118.59 15.30  118.0  117.79 13.34 84.0 202 118.0  1.00
DBP      4 485  72.25 10.83   72.0   72.11 10.38 19.0 105  86.0 -0.05
Pulse    5 485  72.96 11.47   72.0   72.52 11.86 46.0 120  74.0  0.46
      kurtosis   se
Age      -1.23 0.56
BMI       4.15 0.29
SBP       3.44 0.69
DBP       1.07 0.49
Pulse     0.45 0.52
```

The additional statistics presented here are:

- `trimmed` = a trimmed mean (by default in this function, this removes the top and bottom 10% from the data, then computes the mean of the remaining values - the middle 80% of the full data set.)
- `mad` = the median absolute deviation (from the median), which can be used in a manner similar to the standard deviation or IQR to measure spread.
  - If the data are $Y_1, Y_2, ..., Y_n$, then the `mad` is defined as $median(|Y_i - median(Y_i)|)$.
  - To find the `mad` for a set of numbers, find the median, subtract the median from each value and find the absolute value of that difference, and then find the median of those absolute differences.
  - For non-normal data with a skewed shape but tails well approximated by the Normal, the `mad` is likely to be a better (more robust) estimate of the spread than is the standard deviation.
- a measure of `skew`, which refers to how much asymmetry is present in the shape of the distribution. The measure is not the same as the *nonparametric skew* measure that we will usually prefer. The [Wikipedia page on skewness][https://en.wikipedia.org/wiki/Skewness] is very detailed.
- a measure of `kurtosis`, which refers to how outlier-prone, or heavy-tailed the the shape of the distribution is, mainly as compared to a Normal distribution.
- `se` = the standard error of the sample mean, equal to the sample sd divided by the square root of the sample size.

5.5.3 describe in the Hmisc package

```
Hmisc::describe(nh_adults %>% select(Age, BMI, SBP, DBP, Pulse))
```

```
nh_adults %>% select(Age, BMI, SBP, DBP, Pulse)

 5  Variables      500  Observations
---------------------------------------------------------------------------
Age
       n  missing distinct     Info     Mean      Gmd      .05      .10
     500        0       44    0.999     42.1    14.48       23       25
     .25      .50      .75      .90      .95
```

```
     31       42       53       59       61

lowest : 21 22 23 24 25, highest: 60 61 62 63 64
---------------------------------------------------------------------------
BMI
       n  missing distinct     Info     Mean      Gmd      .05      .10
     497        3      203        1    28.73    6.947    19.90    22.00
     .25      .50      .75      .90      .95
   24.20    27.70    32.10    36.54    40.82

lowest : 17.8 18.0 18.1 18.2 18.4, highest: 47.6 48.6 48.8 62.8 69.0
---------------------------------------------------------------------------
SBP
       n  missing distinct     Info     Mean      Gmd      .05      .10
     485       15       71    0.999    118.6    16.51       96      101
     .25      .50      .75      .90      .95
     109      118      127      137      143

lowest :  84  85  86  89  91, highest: 163 167 168 172 202
---------------------------------------------------------------------------
DBP
       n  missing distinct     Info     Mean      Gmd      .05      .10
     485       15       57    0.999    72.25    12.04       56       59
     .25      .50      .75      .90      .95
      65       72       79       86       90

lowest :  19  41  45  47  49, highest: 100 101 102 103 105
---------------------------------------------------------------------------
Pulse
       n  missing distinct     Info     Mean      Gmd      .05      .10
     485       15       31    0.997    72.96    12.81       56       60
     .25      .50      .75      .90      .95
      64       72       80       88       92

lowest :  46  48  50  52  54, highest:  98 100 102 108 120
---------------------------------------------------------------------------
```

The `Hmisc` package's version of `describe` for a distribution of data presents three new ideas, in addition to a more comprehensive list of quartiles (the 5$^{\text{th}}$, 10$^{\text{th}}$, 25$^{\text{th}}$, 50$^{\text{th}}$, 75$^{\text{th}}$, 90$^{\text{th}}$ and 95^th are shown) and the lowest and highest few observations. These are:

- `distinct` - the number of different values observed in the data.
- `Info` - a measure of how "continuous" the variable is, related to how many "ties" there are in the data, with Info taking a higher value (closer to its maximum of one) if the data are more continuous.
- `Gmd` - the Gini mean disfference - a robust measure of spread that is calculated as the mean absolute difference between any pairs of observations. Larger values of Gmd indicate more spread-out distributions.

Chapter 6

Summarizing Categorical Variables

Summarizing categorical variables numerically is mostly about building tables, and calculating percentages or proportions. We'll save our discussion of modeling categorical data for later. Recall that in the `nh_adults` data set we built previously, we had the following categorical variables. The number of levels indicates the number of possible categories for each categorical variable.

Variable	Description	Levels	Type
Sex	sex of subject	2	binary
Race	subject's race	6	nominal
Education	subject's educational level	5	ordinal
PhysActive	Participates in sports?	2	binary
Smoke100	Smoked 100+ cigarettes?	2	binary
SleepTrouble	Trouble sleeping?	2	binary
HealthGen	Self-report health	5	ordinal

6.1 The `summary` function for Categorical data

When R recognizes a variable as categorical, it stores it as a *factor*. Such variables get special treatment from the `summary` function, in particular a table of available values (so long as there aren't too many.)

```
nh_adults %>%
  select(Sex, Race, Education, PhysActive, Smoke100, SleepTrouble, HealthGen) %>%
  summary()
```

```
     Sex             Race              Education     PhysActive Smoke100
 female:253   Asian   : 29   8th Grade      : 24   No :225    No :289
 male  :247   Black   : 57   9 - 11th Grade: 57   Yes:275    Yes:211
              Hispanic: 39   High School    : 81
              Mexican : 43   Some College   :153
              White   :322   College Grad   :185
              Other   : 10
 SleepTrouble     HealthGen
 No :362      Excellent: 51
 Yes:138      Vgood    :153
              Good     :172
              Fair     : 71
              Poor     :  7
```

```
              NA's     : 46
```

6.2 Tables to describe One Categorical Variable

Suppose we build a table to describe the `HealthGen` distribution.

```
nh_adults %>%
    select(HealthGen) %>%
    table(., useNA = "ifany")
```

```
.
Excellent     Vgood      Good      Fair      Poor      <NA>
       51       153       172        71         7        46
```

What if we want to add a total count?

```
nh_adults %>%
    select(HealthGen) %>%
    table(., useNA = "ifany") %>%
    addmargins()
```

```
.
Excellent     Vgood      Good      Fair      Poor      <NA>       Sum
       51       153       172        71         7        46       500
```

What if we want to leave out the missing responses?

```
nh_adults %>%
    select(HealthGen) %>%
    table(., useNA = "no") %>%
    addmargins()
```

```
.
Excellent     Vgood      Good      Fair      Poor       Sum
       51       153       172        71         7       454
```

Let's put the missing values back in, but now calculate proportions instead. Since the total will just be 1.0, we'll leave that out.

```
nh_adults %>%
    select(HealthGen) %>%
    table(., useNA = "ifany") %>%
    prop.table()
```

```
.
Excellent     Vgood      Good      Fair      Poor      <NA>
    0.102     0.306     0.344     0.142     0.014     0.092
```

Now, we'll calculate percentages by multiplying the proportions by 100.

```
nh_adults %>%
    select(HealthGen) %>%
    table(., useNA = "ifany") %>%
    prop.table() %>%
    "*"(100)
```

```
.
Excellent     Vgood      Good      Fair      Poor      <NA>
```

```
    10.2      30.6      34.4      14.2       1.4       9.2
```

6.3 The Mode of a Categorical Variable

A common measure applied to a categorical variable is to identify the mode, the most frequently observed value. To find the mode for variables with lots of categories (so that the `summary` may not be sufficient), we usually tabulate the data, and then sort by the counts of the numbers of observations, as we did with discrete quantitative variables.

```
nh_adults %>%
    group_by(HealthGen) %>%
    summarize(count = n()) %>%
    arrange(desc(count))
```

```
# A tibble: 6 x 2
  HealthGen count
     <fctr> <int>
1      Good   172
2     Vgood   153
3      Fair    71
4 Excellent    51
5        NA    46
6      Poor     7
```

6.4 describe in the Hmisc package

```
Hmisc::describe(nh_adults %>%
                    select(Sex, Race, Education, PhysActive,
                           Smoke100, SleepTrouble, HealthGen))
```

```
nh_adults %>% select(Sex, Race, Education, PhysActive, Smoke100, SleepTrouble, HealthGen)

 7  Variables      500  Observations
---------------------------------------------------------------------------
Sex
       n  missing distinct
     500        0        2

Value      female   male
Frequency     253    247
Proportion  0.506  0.494
---------------------------------------------------------------------------
Race
       n  missing distinct
     500        0        6

Value         Asian    Black Hispanic  Mexican    White    Other
Frequency        29       57       39       43      322       10
Proportion    0.058    0.114    0.078    0.086    0.644    0.020
---------------------------------------------------------------------------
Education
```

```
       n  missing distinct 
     500        0        5 

Value           8th Grade 9 - 11th Grade    High School   Some College
Frequency              24             57             81            153
Proportion          0.048          0.114          0.162          0.306
                                                                      
Value        College Grad
Frequency             185
Proportion          0.370
---------------------------------------------------------------------------
PhysActive 
       n  missing distinct 
     500        0        2 
                      
Value        No  Yes
Frequency   225  275
Proportion 0.45 0.55
---------------------------------------------------------------------------
Smoke100 
       n  missing distinct 
     500        0        2 
                      
Value         No   Yes
Frequency    289   211
Proportion 0.578 0.422
---------------------------------------------------------------------------
SleepTrouble 
       n  missing distinct 
     500        0        2 
                      
Value         No   Yes
Frequency    362   138
Proportion 0.724 0.276
---------------------------------------------------------------------------
HealthGen 
       n  missing distinct 
     454       46        5 
                                                        
Value      Excellent     Vgood      Good      Fair      Poor
Frequency         51       153       172        71         7
Proportion     0.112     0.337     0.379     0.156     0.015
---------------------------------------------------------------------------
```

6.5 Cross-Tabulations

It is very common for us to want to describe the association of one categorical variable with another. For instance, is there a relationship between Education and SleepTrouble in these data?

```r
nh_adults %>%
    select(Education, SleepTrouble) %>%
    table() %>%
    addmargins()
```

```
                SleepTrouble
Education        No Yes Sum
  8th Grade      15   9  24
  9 - 11th Grade 40  17  57
  High School    67  14  81
  Some College  107  46 153
  College Grad  133  52 185
  Sum           362 138 500
```

To get row percentages, we can use:

```
nh_adults %>%
    select(Education, SleepTrouble) %>%
    table() %>%
    prop.table(., 1) %>%
    "*"(100)
```

```
                SleepTrouble
Education              No      Yes
  8th Grade      62.50000 37.50000
  9 - 11th Grade 70.17544 29.82456
  High School    82.71605 17.28395
  Some College   69.93464 30.06536
  College Grad   71.89189 28.10811
```

For column percentages, we use 2 instead of 1 in the `prop.table` function. Here, we'll also round off to two decimal places:

```
nh_adults %>%
    select(Education, SleepTrouble) %>%
    table() %>%
    prop.table(., 2) %>%
    "*"(100) %>%
    round(.,2)
```

```
                SleepTrouble
Education           No   Yes
  8th Grade       4.14  6.52
  9 - 11th Grade 11.05 12.32
  High School    18.51 10.14
  Some College   29.56 33.33
  College Grad   36.74 37.68
```

Here's another approach, to look at the cross-classification of Race and HealthGen:

```
xtabs(~ Race + HealthGen, data = nh_adults)
```

```
          HealthGen
Race        Excellent Vgood Good Fair Poor
  Asian             4     7    9    2    1
  Black             7    11   16   11    2
  Hispanic          1     9   18    8    0
  Mexican           5     6   12   16    1
  White            34   115  115   32    3
  Other             0     5    2    2    0
```

6.5.1 Cross-Classifying Three Categorical Variables

Suppose we are interested in `Smoke100` and its relationship to `PhysActive` and `SleepTrouble`.

```
xtabs(~ Smoke100 + PhysActive + SleepTrouble, data = nh_adults)
```

```
, , SleepTrouble = No

        PhysActive
Smoke100  No Yes
     No   99 135
     Yes  62  66

, , SleepTrouble = Yes

        PhysActive
Smoke100  No Yes
     No   26  29
     Yes  38  45
```

We can also build a **flat** version of this table, as follows:

```
ftable(Smoke100 ~ PhysActive + SleepTrouble, data = nh_adults)
```

```
                        Smoke100  No Yes
PhysActive SleepTrouble                 
No         No                     99  62
           Yes                    26  38
Yes        No                    135  66
           Yes                    29  45
```

And we can do this with `dplyr` functions, as well, for example...

```
nh_adults %>%
    select(Smoke100, PhysActive, SleepTrouble) %>%
    table()
```

```
, , SleepTrouble = No

        PhysActive
Smoke100  No Yes
     No   99 135
     Yes  62  66

, , SleepTrouble = Yes

        PhysActive
Smoke100  No Yes
     No   26  29
     Yes  38  45
```

Baumer, Benjamin S., Daniel T. Kaplan, and Nicholas J. Horton. 2017. *Modern Data Science with R*. Boca Raton, FL: CRC Press. https://mdsr-book.github.io/.

Bock, David E., Paul F. Velleman, and Richard D. De Veaux. 2004. *Stats: Modelling the World*. Boston MA: Pearson Addison-Wesley.

Cetinkaya-Rundel, Mine. 2017. "Teaching Data Science to New useRs." bit.ly/user2017.

Fox, John, and Sanford Weisberg. 2011. *An R Companion to Applied Regression*. Second. Thousand Oaks CA: Sage. http://socserv.socsci.mcmaster.ca/jfox/Books/Companion.

Gelman, Andrew, and Jennifer Hill. 2007. *Data Analysis Using Regression and Multilevel-Hierarchical Models*. New York: Cambridge University Press. http://www.stat.columbia.edu/~gelman/arm/.

Gelman, Andrew, and Deborah Nolan. 2017. *Teaching Statistics: A Bag of Tricks*. Second. Oxford, UK: Oxford University Press.

Grolemund, Garrett, and Hadley Wickham. 2017. *R for Data Science*. O'Reilly. http://r4ds.had.co.nz/.

Harrell, Frank E., and James C. Slaughter. 2017. *Biostatistics for Biomedical Research*. Vanderbilt University School of Medicine. biostat.mc.vanderbilt.edu/ClinStat.

Ismay, Chester, and Albert Y. Kim. 2017. *ModernDive: An Introduction to Statistical and Data Sciences via R*. http://moderndive.com/.

Norman, Geoffrey R., and David L. Streiner. 2014. *Biostatistics: The Bare Essentials*. Fourth. People's Medical Publishing House.

Vittinghoff, Eric, David V. Glidden, Stephen C. Shiboski, and Charles E. McCulloch. 2012. *Regression Methods in Biostatistics: Linear, Logistic, Survival, and Repeated Measures Models*. Second. Springer-Verlag, Inc. http://www.biostat.ucsf.edu/vgsm/.